CROSS-SEC 2026

The First International Conference on Cross-Domain Security in Distributed, Intelligent and Critical Systems

ISBN: 978-1-68558-442-9

April 19 - 23, 2026

Lisbon, Portugal

CROSS-SEC 2026 Editors

Andreas Aßmuth, Kiel University of Applied Sciences, Germany

CROSS-SEC 2026

Forward

The First International Conference on Cross-Domain Security in Distributed, Intelligent and Critical Systems (CROSS-SEC 2026), held on April 19 – 23, 2026, initiated a series of events addressing security as it applies to various domains.

CROSS-SEC aims to establish a focused, technically grounded, and interdisciplinary security conference within the framework of the ComputationWorld Congress. It is intended as a successor to our previous special tracks on cloud and security, with a broader and cleaner alignment toward IT security across domains. The conference will address security, trust, and resilience in distributed, AI-enhanced, and mission-critical systems – including cloud, edge, IoT/OT (Internet of Things and Operational Technology), cryptography, forensics, and related areas.

This event attracted excellent contributions and active participation from all over the world. We were very pleased to receive top quality contributions.

We take here the opportunity to warmly thank all the members of the CROSS-SEC 2026 technical program committee, as well as the numerous reviewers. The creation of a high quality conference program would not have been possible without their involvement. We also kindly thank all the authors that dedicated much of their time and effort to contribute to CROSS-SEC 2026. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations and sponsors. We also gratefully thank the members of the CROSS-SEC 2026 organizing committee for their help in handling the logistics and for their work that made this professional meeting a success.

We hope CROSS-SEC 2026 was a successful international forum for the exchange of ideas and results between academia and industry that will promote further progress in the area of security. We also hope that Lisbon provided a pleasant environment during the conference and everyone saved some time to enjoy this beautiful city.

CROSS-SEC 2026 General Chair

Andreas Aßmuth, Kiel University of Applied Sciences, Germany

CROSS-SEC 2026 Steering Committee

Christoph P. Neumann, Ostbayerische Technische Hochschule Amberg-Weiden, German
Sebastian Fischer, Ostbayerische Technische Hochschule Regensburg, Germany
Aspen Olmsted, Wentworth Institute of Technology, Boston, USA
Steve Chan, Decision Engineering Analysis Laboratory, USA
Carla Merkle Westphall, UFSC, Brazil

CROSS-SEC 2026

Committee

CROSS-SEC 2026 General Chair

Andreas Aßmuth, Kiel University of Applied Sciences, Germany

CROSS-SEC 2026 Steering Committee

Christoph P. Neumann, Ostbayerische Technische Hochschule Amberg-Weiden, German
Sebastian Fischer, Ostbayerische Technische Hochschule Regensburg, Germany
Aspen Olmsted, Wentworth Institute of Technology, Boston, USA
Steve Chan, Decision Engineering Analysis Laboratory, USA
Carla Merkle Westphall, UFSC, Brazil
Manmeet Kaur Mahinderjit Singh, University Sains Malaysia, Malaysia
Hiroki Kuzuno, Kobe University, Japan
Mussa Ally Dida, Nelson Mandela African Institution of Science and Technology, Tanzania
Fred Kaggwa, Mbarara University of Science and Technology (MUST), Uganda

CROSS-SEC 2026 Technical Program Committee

Ali Abdelli, University of Monastir, Tunisia
Florian Adamsky, Hof University of Applied Sciences, Germany
Dewan Mohammed Abdul Ahad, University of North Carolina at Charlotte, USA
Liron Ahmeti, Ostbayerische Technische Hochschule Regensburg, Germany
Mussa Ally Dida, Nelson Mandela African Institution of Science and Technology, Tanzania
Halil Arslan, Sivas Cumhuriyet University, Turkey
Andreas Aßmuth, Kiel University of Applied Sciences, Germany
Tobias J. Bauer, Fraunhofer AISEC, Germany
Sebastian Berndt, Technische Hochschule Lübeck, Germany
Stefano Braghin, IBM Research, Dublin, Ireland
Elif Calik, University of Galway, Ireland
Steve Chan, Decision Engineering Analysis Laboratory, USA
Pitpimon Choorod, King Mongkut's University of Technology North Bangkok, Thailand
Tobias Eggendorfer, Technische Hochschule Ingolstadt, Germany
Ian Ferguson, Abertay University, UK
Sebastian Fischer, Ostbayerische Technische Hochschule - Regensburg, Germany
Richard Frank, Simon Fraser University, Canada
Philipp Fuxen, Ostbayerische Technische Hochschule Regensburg, Germany
Marcus Gelderie, Hochschule Aalen, Germany
Mathias Gerstner, Ostbayerische Technische Hochschule Regensburg, Germany
Julian Graf, Ostbayerische Technische Hochschule Regensburg, Germany
Oliver Hahm, Frankfurt University of Applied Sciences, Germany

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

A Note on the Post-Quantum Security of Identity-Based Encryption on Isogenous Pairing Groups

Malte Andersch, Cezary Pilaszewicz and Marian Margraf
Fachbereich Mathematik und Informatik
Freie Universität Berlin
Berlin, Germany
e-mail: {malte.andersch | cezary.pilaszewicz | marian.margraf}@fu-berlin.de

***Abstract*—The development of cryptographic schemes that remain secure in the post-quantum era is an urgent challenge, particularly in light of the growing ubiquity of low-power devices and the looming threat of quantum computing. Identity-Based Encryption (IBE) offers a compelling alternative to traditional Public Key Infrastructures by simplifying key management, but most classical IBE schemes rely on number-theoretic assumptions that are vulnerable to quantum attacks. In response, Koshiba and Takashima proposed a novel approach based on *Isogenous Pairing Groups* (IPGs), claiming partial quantum resistance. In this work, we critically examine their construction and security claims and investigate the security in standard security notions. We show that the proposed scheme, despite its theoretical elegance, reduces to the Elliptic Curve Discrete Logarithm Problem (ECDLP) on supersingular curves, which can be broken in polynomial time by quantum algorithms in the IND-ID-CPA setting and in subexponential time classically in the authors' security models. Our analysis reveals structural weaknesses inherent to the IPG framework, such as the use of explicit group elements in prime-order groups and exploitable isogeny homomorphisms, which undermine the post-quantum security of isogeny-based cryptography. These findings suggest that IPG-based constructions, which use more than a single point per pairing group, are unlikely to provide robust post-quantum security. In Koshiba and Takashima's proposed IBE system, we can directly observe this when paying regard to not only the IBE's Master Secret Key but also Users' Secret Keys.**

Keywords-Identity-Based Encryption; Isogeny; Post-Quantum Cryptography; Isogenous Pairing Groups.

I. Introduction

In an increasingly connected and digital world, ensuring robust security has become a fundamental requirement in the design of modern information systems. The growing integration of low-power, resource-constrained devices - alongside the accelerating development of quantum computing - poses significant new challenges. These developments highlight the urgent need for cryptographic schemes that are both secure and efficient across a wide range of deployment environments.

Conventional Public Key Infrastructures, though well-established and secure, involve considerable overhead due to the need for certificate issuance, distribution, and validation. *Identity-Based Encryption* (IBE) addresses these limitations by allowing a User's Identity - such as an email address or device ID - to act directly as their public key, thereby eliminating the need for certificates and simplifying key management.

In such systems, a central authority known as the *Trusted Entity* (or *Private Key Generator*, PKG) holds a *Master Secret Key* (MSK) and uses it to derive Secret Keys for users based on their identities. This streamlined design is particularly attractive in applications like secure messaging, mobile platforms, hierarchical authorization systems, and direct device-to-device encryption. However, the centralized nature of IBE introduces a significant trade-off: the PKG can compute any User's Secret Key, which creates a single point of trust and potential vulnerability if the authority is compromised.

The foundational idea of IBE was first proposed by Shamir in 1985 [1], and later realized in a concrete pairing-based construction by Boneh and Franklin in 2001 [2]. Since then, many IBE schemes have emerged, relying on a variety of computational hardness assumptions ranging from number-theoretic problems to constructions based on lattices.

However, the arrival of quantum computing presents a critical threat to the security of some of these classical approaches. Quantum algorithms - most notably Shor's algorithm - can efficiently solve problems like factoring and discrete logarithms, which underpin the security of most traditional public key and IBE schemes. This renders them vulnerable in the face of a quantum-capable adversary.

In response, there has been a major shift toward the development of *Post-Quantum Cryptographic* (PQC) primitives - schemes designed to remain secure against both classical and quantum attacks. Designing IBE systems that are quantum-resistant remains a pressing challenge in this broader effort. Among the many emerging schemes, Koshiba and Takashima proposed a new framework for partially quantum secure systems called Isogenous Pairing Groups (IPGs) [3] alongside an IBE using these IPGs. We find their idea to be generally appealing and elegant. However, there are some serious doubts about the validity of their security claims in a more practical scenario than described by their new security models, especially when considering the security of the IBE's Users' Secret Keys. Thus, we aim to investigate the weaknesses of the proposed scheme and analyze whether its security goals can be upheld in more standard notions of security.

This paper is organized as follows: First, Section II introduces the necessary mathematical foundations used in this paper. Section III provides a general overview of IBE alongside the most common security definitions and assumptions. In Section IV, we present the IBE scheme proposed by Koshiba and Takashima, outline their security claims, and argue for a more thorough analysis in different security models than the

 ISBN: 978-1-68558-442-9

original authors' propositions. Finally, Section V highlights our attacks against the previously introduced scheme in the Chosen Plaintext Attack for IBE (ID-CPA) setting and the original authors' Payload-Hiding against Quantum adversaries (PH-PQ) setting and performs a complexity analysis of the presented attacks.

II. Preliminaries

For this paper, we consider elliptic curves in the following manner:

Definition 2.1. *Let E be an elliptic curve over the finite field $\mathbb{F}_{p^m}$ for p prime and an integer $m \geq 1$, given by a Weierstrass equation $E: y^2 = x^3 + ax + b, \quad a, b \in \mathbb{F}_{p^m}$. The set of $\mathbb{F}_{p^m}$-rational points on E is*

$$E(\mathbb{F}_{p^m}) = \{(x, y) \in \mathbb{F}_{p^m}^2 : y^2 = x^3 + ax + b\} \cup \{\mathcal{O}\},$$

where $\mathcal{O}$ denotes the point at infinity. Equipped with the chord-and-tangent addition law, $E(\mathbb{F}_{p^m})$ forms a finite abelian group with $\mathcal{O}$ as the neutral element.

A. Isogenies

Ever since the seminal work of Shamir introducing the notion of IBE in [1], elliptic curves have played a central role in the design of IBE systems, including prominent examples like the Boneh-Franklin scheme [2] and the Boneh-Boyen scheme [4]. These constructions largely depend on the hardness of problems related to scalar multiplication on elliptic curves and bilinear pairings. However, the advent of quantum computing threatens the foundational assumptions of these schemes, as quantum algorithms can solve the underlying Discrete Logarithm Problem (DLP) efficiently.

To address this, attention has shifted towards alternative cryptographic primitives which resist quantum attacks. One such promising candidate is the class of *isogenies* between elliptic curves. Although their use in cryptography is relatively recent, isogeny-based schemes have gained momentum due to their conjectured post-quantum security and the appealing advantage of compact key sizes, which is particularly beneficial in resource-constrained environments.

Definition 2.2. *Let K be a field and $\overline{K}$ its algebraic closure. Let E_1 and E_2 be elliptic curves over $\overline{K}$, with respective points at infinity $\mathcal{O}_{E_1}$ and $\mathcal{O}_{E_2}$. Then, an* isogeny *is a finite, non-constant morphism $\phi : E_1 \to E_2$ defined over $\overline{K}$ such that $\phi(\mathcal{O}_{E_1}) = \mathcal{O}_{E_2}$. If this morphism is also defined over K, we say that ϕ is an isogeny over K.*

While we do not delve into the mathematical theory underlying isogenies - see [5] for a thorough introduction - we do introduce the central computational problem that forms the security foundation of isogeny-based cryptography:

Problem 2.3 (General Isogeny Problem, [6])**.** *Given two elliptic curves E_1 and E_2 defined over a finite field K, with $\#E_1 = \#E_2$, where $\#E$ denotes the number of points on the curve E, find an isogeny $\phi : E_1 \to E_2$ defined over K.*

To date, Problem 2.3 is believed to be intractable even for quantum adversaries, particularly in the supersingular setting. In fact, its hardness is closely tied to the difficulty of computing the endomorphism ring of elliptic curves, a problem shown to be equivalent in certain cases [7].

B. Bilinear Pairings

As the original primitive used in the Boneh-Franklin IBE construction, *pairings* have gained a lot of attention not only in the world of IBEs but also in a new subbranch of cryptography, the *pairing-based cryptography*. For the main concept of Koshiba and Takashima's paper, the IPGs, we also need to briefly introduce pairings as their second component.

Definition 2.4. *Let G_1, G_2, G_T be cyclic groups. A* Pairing *is a map*

$$e : G_1 \times G_2 \to G_T,$$

satisfying the following properties:

- *e is bilinear: $e(aP, bQ) = e(P, Q)^{ab}$ for all $a, b \in \mathbb{Z}, (P, Q) \in G_1 \times G_2$,*
- *e is non-degenerate: $e(P, Q) \neq 1$ for some $(P, Q) \in G_1 \times G_2$,*
- *e is efficiently computable.*

For G_1, G_2, the additive notation is most commonly used, while we use multiplicative notation for G_T.

While the considerations work for any pairing, we will use the Weil pairing used by the original authors of [3], and thus we will mainly focus on this specific pairing's properties here.

Before exploring the Weil pairing itself, we first need to introduce the notion of *torsion subgroups*.

Definition 2.5. *Let $E(\mathbb{F}_{p^m})$ an elliptic curve, then $E(\overline{\mathbb{F}_{p^m}})$ is the group of points over the algebraic closure of $\mathbb{F}_{p^m}$. Let also $n \in \mathbb{N}_+$. Then the group*

$$E[n] = \{P \in E(\overline{\mathbb{F}_{p^m}}) \mid [n]P = \mathcal{O}\}$$

is called the n-torsion subgroup of E. The order of all points P in this group divides n.

Using these n-torsion subgroups, we can define the Weil pairing as follows:

Definition 2.6. *Let $E(\mathbb{F}_{p^m})$ be an elliptic curve with p prime and $n \in \mathbb{N}^+$ with $\gcd(n, p^m) = 1$. The Weil pairing is the map*

$$e : E[n] \times E[n] \to \overline{\mathbb{F}_{p^m}}.$$

Specifically, the Weil pairing maps to the set of all primitive n-th roots of unity over $\overline{\mathbb{F}_{p^m}}$.

Additionally, the Weil pairing also satisfies the following properties:

(*a*) *Identity*: $e(P, P) = 1$, for all $P \in E[n]$
(*b*) *Bilinearity*: as before
(*c*) *Alternation*: $e(P, Q) = e(Q, P)^{-1}$, for all $P, Q \in E[n]$
(*d*) *Non-degeneracy*: If $e(P_1, P_2) = 1$ for all $P_2 \in E[n]$, then $P_1 = \mathcal{O}$

(*e*) *Embedding Degree*: Let k be the smallest non-zero integer such that $p^{mk} \equiv 1 \mod n$. Then $E[n] \subseteq E(\mathbb{F}_{p^{mk}})$, and therefore $e(P_1, P_2) \in \mathbb{F}_{p^{mk}}$ for all $P_1, P_2 \in E[n]$

(*f*) *Endomorphism Interaction*: For all separable $\alpha \in End(E)$ and $S, T \in E[n]$ it holds

$$e(\alpha(S), \alpha(T)) = e(S, T)^{deg(\alpha)}.$$

To keep this paper concise, we will not delve into the construction and implementation of the Weil pairing and instead refer the interested reader to [8]. It is important to note that while e_n is commonly used to denote the Weil pairing for the n-torsion subgroup, we omit this index n and instead later on use the indices t to indicate the group the pairing belongs to.

C. Isogenous Pairing Groups

The notion of IPGs is first introduced by Koshiba and Takashima in [3]. Through the combination of isogenies and bilinear pairings, the authors aim to establish a framework for constructing IBEs, where the MSK is protected via quantum-safe isogenies and the individual messages are classically secure through pairing cryptography.

Before delving into the concrete definition of IBE schemes and the proposed IBE scheme using IPGs, we first formally define IPGs and outline the fundamental security assumptions.

Definition 2.7 (Isogenous Pairing Groups, Def. 4 in [3])**.** *IPGs are an array of length $t \geq 0$ of tuples $\left(\mathbb{G}_t, \hat{\mathbb{G}}_t, g_t, \hat{g}_t, e_t, \phi_t\right)$ together with a target group $\mathbb{G}_T$, where $\left(\mathbb{G}_t, \hat{\mathbb{G}}_t, e_t, \mathbb{G}_T\right)$ are asymmetric pairing groups of prime order q with pairings*

$$e_t : \mathbb{G}_t \times \hat{\mathbb{G}}_t \to \mathbb{G}_T,$$

$\hat{g}_t \in \hat{\mathbb{G}}_t$, isogeny $\phi_t : \mathbb{G}_0 \to \mathbb{G}_t$ and $g_t = \phi_t(g_0) \in \mathbb{G}_t$. For $t = 0$, the isogeny ϕ_0 is the identity function.

The IPGs' generator (Gen^{IPG}) generates a random instance as follows:

$$Gen^{\mathsf{IPG}}(1^\kappa, d) \to \Big(pk^{\mathsf{IPG}} := \Big((\mathbb{G}_t, \hat{\mathbb{G}}_t, g_t, \hat{g}_t, e_t)_{t\in[0,d]}, \mathbb{G}_T\Big), \; sk^{\mathsf{IPG}} := (\phi_t)_{t\in[d]}\Big).$$

The IPGs satisfy the ***compatibility*** *property:*

$$g_T := e_0(g_0, \hat{g}_0) = e_t(g_t, \hat{g}_t) = e_t(\phi_t(g_0), \hat{g}_t) \neq 1, g_T \in \mathbb{G}_T$$

for all $t \in [1, d]$, and we require $\mathbb{G}_t \neq \hat{\mathbb{G}}_t$ for all $t \in [0, d]$.

Based on these IPGs, we can then outline the necessary security definitions for the proposed IBE scheme. The most fundamental one of these is the *Isogeny Problem on IPG*, transferring the general isogeny problem into the setting of IPG.

Problem 2.8 (Isogeny Problem on IPG, Def. 5 in [3])**.** *Let*

$$\Big(pk^{\mathsf{IPG}} := \Big((\mathbb{G}_t, \hat{\mathbb{G}}_t, g_t, \hat{g}_t, e_t)_{t=0,1}, \mathbb{G}_T\Big), \; sk^{\mathsf{IPG}} := \phi_1\Big) \leftarrow Gen^{\mathsf{IPG}}(1^\kappa, 1).$$

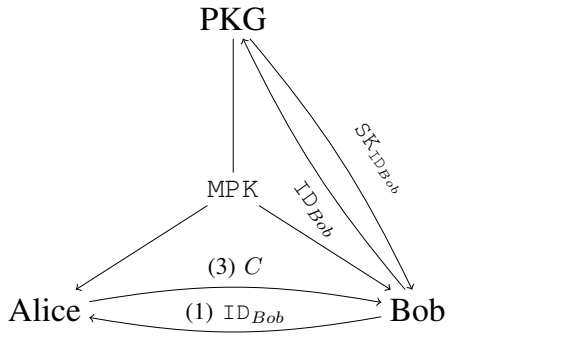


Figure 1. IBE Network Setup.

Given pk^{IPG}, find $sk^{\mathsf{IPG}} = \phi_1$.

Despite the additional inputs when compared to Problem 2.3 with $\mathbb{G}_0 \subseteq E_1(K)$, $\mathbb{G}_1 \subseteq E_2(K)$, Koshiba and Takashima claim Problem 2.8 to have no efficient quantum adversary.

III. Identity-Based Encryption

This section offers a general overview of the subject of IBE. We provide a definition of the necessary algorithms for IBE schemes and the IBE-related security games, as well as a brief discussion of the concept of pre-challenge quantum security in relation to the previously introduced IPGs.

A. Premise

The main challenge that IBE aims to overcome is the exchange of public keys, which is a fundamental part of traditional Public Key Infrastructures. The infrastructure of a typical IBE system is outlined in Figure 1. Here, Alice and Bob both retrieve the Master Public Key MPK from the PKG. When Alice wants to send an encrypted message to Bob, she uses Bob's User Identity $\mathtt{ID}_{Bob}$ and the MPK to encrypt the message M. Bob can retrieve his Secret Key $\mathtt{SK}_{\mathtt{ID}_{Bob}}$ from the PKG.

Definition 3.1. *An IBE scheme consists of four algorithms.*

- `setup`*: An algorithm that receives the security parameter as its input and creates the* `MPK` *and* `MSK`*.*
- `extract`*: An algorithm that uses the* `MPK`*,* `MSK` *and an Identity to generate the Secret Key corresponding to the Identity.*
- `encrypt`*: An algorithm that uses the* `MPK` *and an Identity to encrypt a message.*
- `decrypt`*: An algorithm that uses the* `MPK` *and a Secret Key corresponding to an Identity to decrypt a ciphertext into a message.*

B. Security Models on IBE

In the setting of IBE, classical security notions from Public Key Encryption (PKE) must be adapted to account for Identity-derived keys. The traditional definitions of Indistinguishability under adaptive Chosen Plaintext Attacks (IND-CPA) and

Chosen Ciphertext Attacks (IND-CCA) are extended to accommodate the fact that an adversary may obtain Secret Keys for multiple Identities $\texttt{ID}_1, \ldots, \texttt{ID}_n$, provided the challenge Identity remains hidden. Specifically, the attacker is allowed to query an extraction oracle $\mathcal{O}^{ext}(\texttt{ID})$ to obtain User's Secret Keys for arbitrary identities, except for the challenge Identity. We formalize this new CPA security for IBE (ID-CPA) via the following game-based definition:

Definition 3.2. *For an IBE scheme* $\Pi = (\mathtt{setup}, \mathtt{extract}, \mathtt{encrypt}, \mathtt{decrypt})$ *and a Probabilistic Polynomial-Time (PPT) adversary* $\mathcal{A}$*, the Indistinguishability under adaptive Chosen Plaintext Attack for IBE (IND-ID-CPA) game is defined as follows:*

Algorithm 1 $\text{IND-ID}^{cpa}_{\mathcal{A},\Pi}(\kappa)$ game

Require: 1^κ the security parameter
1: $\texttt{extIDs} \coloneqq \emptyset$
2: $\texttt{MPK}, \texttt{MSK} \leftarrow \Pi.\texttt{setup}(1^\kappa)$
3: $\texttt{ID}^*, M_0, M_1 \leftarrow \mathcal{A}^{\mathcal{O}^{ext}(\cdot)}(ask, 1^\kappa, \texttt{MPK})$
4: $b \xleftarrow{\$} \{0,1\}$
5: $C_{\texttt{ID}^*} \leftarrow \Pi.\texttt{encrypt}(\texttt{MPK}, \texttt{ID}^*, M_b)$
6: $b' \leftarrow \mathcal{A}^{\mathcal{O}^{ext}(\cdot)}(guess, \texttt{MPK}, \texttt{ID}^*, C_{\texttt{ID}^*})$
7: **if** $\texttt{ID}^* \in \texttt{extIDs}$ **then**
8: **return** $\perp$
9: **end if**
10: **return** 1 iff $b' == b$

Figure 2. IND-ID-CPA game for IBE schemes.

For completeness, we also need to define the extraction oracle:

Algorithm 2 $\mathcal{O}^{ext}_{\Pi}(\texttt{ID})$

Require: ID the Identity to extract the Secret Key for
1: $\texttt{extIDs} \coloneqq \texttt{extIDs} \cup \{\texttt{ID}\}$
2: $\texttt{SK}_{\texttt{ID}} \leftarrow \Pi.\texttt{extract}(\texttt{MPK}, \texttt{MSK}, \texttt{ID})$
3: **return** $\texttt{SK}_{\texttt{ID}}$

Figure 3. Extraction Oracle for IBE security games.

We say that an IBE scheme has indistinguishable encryption under adaptive Chosen Plaintext Attack if, for any PPT adversary $\mathcal{A}$*, it holds that* $\mathcal{A}$*'s advantage* $Adv(\mathcal{A}) =: \varepsilon$ *in the success probability, defined as* $\Pr[\textit{IND-ID}^{cpa}_{\mathcal{A},\Pi}(\kappa) = 1] = \frac{1}{2} + \varepsilon$*, is negligible in* κ*.*

This definition can be extended to cover the IBE's Indistinguishability against Chosen Ciphertext Attacks (IND-ID-CCA) by providing the adversary with a decryption oracle, which, like the extraction oracle, rejects any query on the challenge Identity and ciphertext by returning $\perp$.

In addition to Indistinguishability-based (IND) notions, we consider a weaker but still meaningful security property: *One-wayness* (OW) of IBE schemes.

Definition 3.3. *For an IBE scheme* $\Pi = (\mathtt{setup}, \mathtt{extract}, \mathtt{encrypt}, \mathtt{decrypt})$ *and a PPT adversary* $\mathcal{A}$*, the One-wayness under adaptive Chosen Plaintext Attack for IBE (OW-ID-CPA) game is defined as follows:*

Algorithm 3 $\text{OW-ID}^{cpa}_{\mathcal{A},\Pi}(\kappa)$ game

Require: 1^κ the security parameter
1: $\texttt{extIDs} \coloneqq \emptyset$
2: $\texttt{MPK}, \texttt{MSK} \leftarrow \Pi.\texttt{setup}(1^\kappa)$
3: $\texttt{ID}^* \leftarrow \mathcal{A}^{\mathcal{O}^{ext}(\cdot)}(ask, 1^\kappa, \texttt{MPK})$
4: $M \xleftarrow{\$} \mathcal{M}$
5: $C_{\texttt{ID}^*} \leftarrow \Pi.\texttt{encrypt}(\texttt{MPK}, \texttt{ID}^*, M)$
6: $M' \leftarrow \mathcal{A}^{\mathcal{O}^{ext}(\cdot)}(guess, \texttt{MPK}, \texttt{ID}^*, C_{\texttt{ID}^*})$
7: **if** $\texttt{ID}^* \in \texttt{extIDs}$ **then**
8: **return** $\perp$
9: **end if**
10: **return** 1 iff $M' == M$

Figure 4. OW-ID-CPA game for IBE schemes.

The extraction oracle is defined identically to the extraction oracle in the Indistinguishability game; see Algorithm 2.

We say that an IBE scheme provides One-wayness under adaptive Chosen Plaintext Attack if, for any PPT adversary $\mathcal{A}$*, it holds that* $\mathcal{A}$*'s advantage* $Adv(\mathcal{A}) =: \varepsilon$ *in the success probability, defined as* $\Pr[\textit{OW-ID}^{cpa}_{\mathcal{A},\Pi}(\kappa) = 1] = \varepsilon$*, is negligible in* κ*.*

Similar to the Indistinguishability scenario, the One-wayness can be extended to cover the CCA setting by incorporating a decryption oracle. While Indistinguishability is the de facto security standard for PKE schemes, the notion of One-wayness is especially relevant in our context. Any encryption scheme that satisfies One-wayness under CPA can be adapted to provide Indistinguishability under CCA via the Fujisaki-Okamoto transformation (FO-transform) [9]. This means that establishing One-wayness is often sufficient for constructing practically secure IBE schemes.

Naturally, Indistinguishability implies One-wayness: if an adversary can recover the plaintext from a ciphertext (as in the One-wayness game), they can also distinguish which message was encrypted (as in the IND game). In fact, a successful OW-ID-CPA adversary can be used as a subroutine in the IND-ID-CPA game by running it on the challenge ciphertext and comparing its output against the two original messages to guess the correct bit.

C. Pre-Challenge Quantum Security

Intuitively, the pre-challenge quantum security is aimed at protecting the MSK from quantum adversaries instead of the actual ciphertexts. This prevents an adversary from gaining classical access to the decryption of all ciphertexts via a single attack on the scheme with a quantum computer. Hence, allowing a quantum adversary to only decrypt ciphertexts one by one and thereby speculating upon the slow speed of introduction of powerful quantum computers. The cost of a

single attack is assumed to be high enough to mitigate the usefulness of early quantum machines against the scheme.

Using this assumption combined with the Decisional Bilinear Diffie-Hellman (DBDH) assumption [10], one obtains the following problem definition, used to prove the security of the proposed IBE scheme.

Problem 3.4 (qIsog-DBDH, Def. 7 in [3])**.** *Let $\mathcal{A} \coloneqq (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary, where $\mathcal{A}_1$ is modeled as a polynomial-time quantum adversary and $\mathcal{A}_2$ as a classical PPT machine. Let*

$$\Big(pk^{\mathsf{IPG}} \coloneqq \Big((\mathbb{G}_t, \hat{\mathbb{G}}_t, g_t, \hat{g}_t, e_t)_{t=0,1}, \mathbb{G}_T\Big), \\ sk^{\mathsf{IPG}} \coloneqq \phi_1\Big) \leftarrow Gen^{\mathsf{IPG}}(1^\kappa, 1)$$

be generated by the IPG setup algorithm. Then:

- *$\mathcal{A}_1$ receives pk_{IPG} and outputs a state **state** $\leftarrow \mathcal{A}_1(pk_{\mathsf{IPG}})$.*
- *Next, random elements $\alpha, \beta, \delta \xleftarrow{\$} \mathbb{F}_q$ are sampled.*
- *$\mathcal{A}_2$ receives $\mathcal{X}_b$ for a uniformly chosen bit $b \xleftarrow{\$} \{0,1\}$, where $\mathcal{X}_0, \mathcal{X}_1$ are defined as the following tuples*

$$\mathcal{X}_0 \coloneqq \Big(\textbf{\textit{state}},\ g_0^\alpha,\ \hat{g}_1^\beta,\ g_T^{\alpha\beta}\Big), \\ \mathcal{X}_1 \coloneqq \Big(\textbf{\textit{state}},\ g_0^\alpha,\ \hat{g}_1^\beta,\ g_T^\delta\Big),$$

  with $g_T = e_0(g_0, \hat{g}_0) = e_1(\phi_1(g_0), \hat{g}_1) = e_1(g_1, \hat{g}_1)$.
- *$\mathcal{A}_2$ outputs a guess bit $b' \in \{0,1\}$.*

If $b = b'$, then the adversary $\mathcal{A} \coloneqq (\mathcal{A}_1, \mathcal{A}_2)$ wins*. The qIsog-DBDH assumption then states that for any adversary $\mathcal{A}$ as defined above, $\mathcal{A}$'s advantage for solving the qIsog-DBDH problem is negligible in κ.*

Koshiba and Takashima reduce Problem 3.4 to Problem 2.8, to show that a cryptosystem that is provably secure under the qIsog-DBDH assumption has no efficient adversary.

Applying the idea of pre-challenge quantum security to the established ID-CPA and ID-CCA notions of security, we replace the ask phase of the adversary $\mathcal{A}$ in the games defined in Algorithm 1 and Algorithm 3 with the quantum algorithm $\mathcal{A}_1$ and the $guess$ phases with the classical algorithm $\mathcal{A}_2$. We still assume all queries to the oracles in both phases to be only classical queries.

IV. IBE USING IPG

Using the notion of IPGs from Section II-C, Koshiba and Takashima introduce an IBE scheme that is supposedly resistant to pre-challenge quantum adversaries. In this section, we want to briefly introduce the scheme itself, alongside the original authors' security claims.

A. Scheme Description

Figure 5 contains the pseudocode for the implementation of the IBE on IPGs scheme. As before, we denote with $\mathbb{G}_t$ the cyclic symmetric pairing groups of prime order q exponential in κ on supersingular elliptic curves over a finite field $\mathbb{F}_{p^m}$.

Anonymous IBE using IPG approach

$\texttt{setup}\,(1^\kappa) \to (\texttt{MPK}, \texttt{MSK})$:
1) Generate IPG parameters:
$$\Big(\text{pk}^{\mathsf{IPG}} \coloneqq \Big((\mathbb{G}_t, \hat{\mathbb{G}}_t, g_t, \hat{g}_t, e_t)_{t=0,1}, \mathbb{G}_T\Big), \\ \text{sk}^{\mathsf{IPG}} \coloneqq \phi_1\Big) \leftarrow \text{Gen}^{\mathsf{IPG}}(1^\kappa, 1).$$
2) Generate a random hash function $H : \mathbb{F}_q \to \mathbb{G}_0$ with $\mathbb{F}_q$ being the space of all possible Identities
3) Output $\texttt{MPK} \coloneqq (\text{pk}^{\mathsf{IPG}}, H)$, $\texttt{MSK} \coloneqq \text{sk}^{\mathsf{IPG}}$

$\texttt{extract}\,(\texttt{MPK}, \texttt{MSK}, \texttt{ID}) \to \texttt{SK}_{\texttt{ID}}$:
1) Calculate $h_0 \coloneqq H(\texttt{ID}) \in \mathbb{G}_0$ and $h_1 \coloneqq \phi_1(h_0)$
2) Output $\texttt{SK}_{\texttt{ID}} \coloneqq h_1$

$\texttt{encrypt}\,(\texttt{MPK}, \texttt{ID}, M) \to C_{\texttt{ID}}$:
1) Calculate $h_0 \coloneqq H(\texttt{ID})$
2) Generate a random exponent $\zeta \xleftarrow{\$} \mathbb{F}_q^\times$
3) Calculate $C \coloneqq \hat{g}_1^\zeta$ and $z \coloneqq e_0(h_0, \hat{g}_0)^\zeta$
4) Encrypt the plaintext as $C_T \coloneqq z \cdot M$
5) Output $C_{\texttt{ID}} \coloneqq (C, C_T)$

$\texttt{decrypt}\,(\texttt{MPK}, \texttt{SK}_{\texttt{ID}}, C_{\texttt{ID}'}) \to M$:
1) Calculate $z' \coloneqq e_1(h_1, C)$
2) Obtain the plaintext as $M' \coloneqq C_T \cdot (z')^{-1}$
3) Output $M \coloneqq M'$

Figure 5. Anonymous IBE against Pre-Challenge Quantum Adversaries proposed in [3].

The correctness of the algorithm for $\texttt{ID} == \texttt{ID}'$ follows from the IPG's compatibility property in Definition 2.7:

$$z' = e_1(h_1, C) = e_1(\phi_1(h_0), \hat{g}_1^\zeta) = e_1(\phi_1(h_0), \hat{g}_1)^\zeta \\ = e_0(h_0, \hat{g}_0)^\zeta = z.$$

The scheme in Figure 5 allows for the encryption of arbitrary messages $M \in \mathbb{F}_q$, thus the encryption of multiple bits. Meanwhile, the Secret Key size for each $\texttt{ID}$ is just a single element of $\mathbb{G}_0$, the ciphertext is the size of two elements of $\mathbb{G}_0$, and the public parameters are precisely the parameters for two elliptic curves. Compared to the Boneh-Franklin IBE, this is simply a doubling in size for the public parameters, while all other parameter sizes are equally large, giving an impression of practicality for the scheme.

B. The Original Authors' Claims

Koshiba and Takashima provide two security claims for the construction in Figure 5.

Claim 4.1 (Theorem 1. in [3])**.** *The proposed IBE scheme is Payload-Hiding against Quantum adversaries (PH-PQ) under Chosen Plaintext Attack under the qIsog-DBDH assumption in the quantum Random Oracle Model.*

While the PH-PQ security closely resembles the standard IND-ID-CPA security, there is a small distinction, in that the adversary must not query the challenge Identity to the mapping hash function H (implemented as a quantum accessible random oracle) in the pre-challenge phase. We present the definition of PH-PQ security in the appendix of the full version of this paper [11].

Besides Claim 4.1, Koshiba and Takashima also present another claim on a security notion they call Anonymous-ID (A-ID) security.

Claim 4.2 (Theorem 2. in [3])**.** *The proposed IBE scheme is A-ID secure against pre-challenge quantum adversaries under the qIsog-DBDH assumption in the quantum Random Oracle Model.*

Although this notion of security has little importance in literature for IBE schemes, we also want to provide its definition in the appendix of the full version of this paper [11].

Both these new notions can be put into perspective when introducing another notion of security, namely One-way Payload-Hiding against Quantum adversaries under Chosen Plaintext Attack (OW-PH-PQ-CPA); see Definition 4.3.

Definition 4.3. *For an IBE scheme* $\Pi = (\texttt{setup}, \texttt{extract}, \texttt{encrypt}, \texttt{decrypt})$ *and a PPT adversary* $\mathcal{A}$*, the OW-PH-PQ-CPA game is defined as follows:*

Algorithm 4 OW-PH-PQ$^{cpa}_{\mathcal{A},\Pi}(\kappa)$ game

Require: 1^κ the security parameter
1: $\texttt{extIDs} \coloneqq \emptyset$
2: $\texttt{MPK}, \texttt{MSK} \leftarrow \Pi.\texttt{setup}(1^\kappa)$
3: $\texttt{ID}^* \leftarrow \mathcal{A}^{\mathcal{O}^{ext}(\cdot)}(ask, 1^\kappa, \texttt{MPK})$
4: $M^* \xleftarrow{\$} \mathbb{F}_q$
5: $C_{\texttt{ID}^*} \leftarrow \Pi.\texttt{encrypt}(\texttt{MPK}, \texttt{ID}^*, M^*)$
6: $M \leftarrow \mathcal{A}^{\mathcal{O}^{ext}(\cdot)}(guess, \texttt{MPK}, \texttt{ID}^*, C_{\texttt{ID}^*})$
7: **if** $\texttt{ID}^* \in \texttt{extIDs}$ **then**
8: **return** $\perp$
9: **end if**
10: **return** 1 iff $M == M^*$

Figure 6. OW-PH-PQ-CPA game for IBE schemes.

During the pre-challenge phase, the adversary is not allowed to query the challenge Identity to the hash function H, implemented as a quantum-accessible random oracle. The extraction oracle is defined identically to the extraction oracle in the Indistinguishability game (see Algorithm 2) and can only be queried classically.

We say that an IBE scheme is OW-PH-PQ secure under adaptive Chosen Plaintext Attack if, for any PPT adversary $\mathcal{A}$, it holds that $\mathcal{A}$'s advantage $Adv(\mathcal{A}) =: \varepsilon$ in the success probability, defined as $\Pr[\textit{OW-PH-PQ}^{cpa}_{\mathcal{A},\Pi}(\kappa) = 1] = \varepsilon$, is negligible in κ.

Based on the game defined in Definition 4.3, we present the following lemmas:

Lemma 4.4. *Let* $\Pi = (\texttt{setup}, \texttt{extract}, \texttt{encrypt}, \texttt{decrypt})$ *be a Payload-Hiding against Quantum adversaries IBE scheme, i.e., for every adversary $\mathcal{A}$ in the PH-PQ$^{cpa}_{\mathcal{A},\Pi}(\kappa)$ game, it holds that $Adv(\mathcal{A}) \leq negl(\kappa)$ for some negligible function $negl(\cdot)$. Then it must hold for every adversary $\mathcal{A}'$ in the OW-PH-PQ$^{cpa}_{\mathcal{A}',\Pi}(\kappa)$ game, that $Adv(\mathcal{A}') \leq negl'(\kappa)$, for some negligible function $negl'(\cdot)$.*

Lemma 4.5. *Let* $\Pi = (\texttt{setup}, \texttt{extract}, \texttt{encrypt}, \texttt{decrypt})$ *be an A-ID secure IBE scheme, i.e., for every adversary $\mathcal{A}$ in the A-ID$^{cpa}_{\mathcal{A},\Pi}(\kappa)$ game, it holds that $Adv(\mathcal{A}) \leq negl(\kappa)$ for some negligible function $negl(\cdot)$. Then it must hold for every adversary $\mathcal{A}'$ in the OW-PH-PQ$^{cpa}_{\mathcal{A}',\Pi}(\kappa)$ game that $Adv(\mathcal{A}') \leq negl'(\kappa)$, for some negligible function $negl'(\cdot)$.*

We prove Lemma 4.4 and Lemma 4.5 in the appendix of the full version of this paper [11].

Using Lemma 4.4 and Lemma 4.5, it becomes clear that any successful attack in the OW-PH-PQ game can directly be transformed into an attack on the claims by Koshiba and Takashima.

Furthermore, it is clear that these security notions are directly implied by the standard OW-ID-CPA security notions. Any attacker breaking OW-PH-PQ security can also be directly used to break OW-ID-CPA security. The only difference lies in the restricted oracle accessibility for the challenge Identity in the PH-PQ scenario, which, when lifted, does not affect the attacker's success probability.

We argue that the original authors' models, by restricting the calculation of the challenge Identity's public key, do not provide full practicality, as the direct calculation of a User's Public Key from their Identity is one of the main aspects of IBE. In practice, the restriction posed upon the hash function mapping the identities to public keys cannot be implemented.

Furthermore, while Koshiba and Takashima describe their intuition of pre-challenge quantum security as the adversary being only able to decrypt each ciphertext one-by-one, they simply reduce this to protecting the MSK and neglect the post-quantum security of the Users' Secret Keys. For these reasons, we want to extend their security analysis to the ID-CPA model in the pre-challenge quantum adversary setting in order to shed light on these aspects of the scheme's security.

V. Proposed Attacks against the IBE on IPGs

This section will give a brief introduction to the *Discrete Logarithm Problem* (DLP) on elliptic curves before outlining different attacks in a pre-challenge quantum setting and a classical setting.

A. Discrete Logarithm Problem on Elliptic Curves

In cryptography, one-way trapdoor functions form the basis for modern asymmetric ciphers. While it remains unclear whether those actually exist, several mathematically hard problems have been proposed as the foundation for such functions. One of these problems is the DLP.

Problem 5.1 (Discrete Logarithm Problem)**.** *Let $\mathbb{G}$ be a group, and let $x, y \in \mathbb{G}$ be elements such that y is in the subgroup generated by x. The DLP is the problem of determining an integer such that $x^d = y$.*

When working with elliptic curves, the group of points on the curve is typically denoted as additive and the d-fold scalar multiplication of a point P is written as $[d]P$. We can then reformulate this problem for the group of points on elliptic curves.

Problem 5.2 (Elliptic Curve Discrete Logarithm Problem)**.** *Let $E(\mathbb{F}_{p^m})$ be an elliptic curve, and let $P, Q \in E(\mathbb{F}_{p^m})$ be points such that Q is in the subgroup generated by P. The* Elliptic Curve Discrete Logarithm Problem *(ECDLP) asks to find an integer d satisfying $Q = [d]P$.*

To this day, the fastest known algorithms to solve the ECDLP classically are collision algorithms such as Pollard's Rho [12] boasting exponential runtimes ($\sqrt{p}$ for a curve group of order q in the case of Pollard's Rho). As no efficient classical attacks exist, the ECDLP forms the basis for many modern cryptographic algorithms, such as ECDSA [13] and ECDH [14] (based on a closely related hard problem).

B. Quantum Attack against the Discrete Logarithm Problem

The ECDLP can be solved efficiently on a quantum computer in the same way as was shown by Shor [15] for the DLP in multiplicative groups of cyclic groups of prime order. More precisely, the reason is that the ECDLP reduces to an instance of the hidden subgroup problem for finite abelian groups. Given $E(\mathbb{F}_{p^m})$ and $P, Q \in E(\mathbb{F}_{p^m})$ such that $Q = [k]P$ for some $k \in \mathbb{Z}$, and letting r be the order of P, the following function hides the subgroup generated by $(\overline{k}, \overline{-1}) \in \mathbb{Z}_r \times \mathbb{Z}_r$.

$$\begin{aligned} f\colon \mathbb{Z}_r \times \mathbb{Z}_r &\to E(\mathbb{F}_{p^m}) \\ (\overline{x}, \overline{y}) &\mapsto [x]P + [y]Q \end{aligned}$$

As shown in [16], there is a quantum algorithm that finds a generator of the hidden subgroup with probability at least $1 - \frac{1}{r}$ making $\mathcal{O}(\log(r))$ calls to f, along with efficiently computable quantum operations and classical post-processing that runs in time $\mathcal{O}(\log^2(r))$. The function f is efficiently computable by the standard double-and-add technique, ensuring that the overall procedure is. For an optimized way to implement f, see [17]. Moreover, from any generator found by the algorithm, k can be efficiently recovered.

C. Proposed Quantum Attack

We present our attack in the pre-challenge quantum OW-ID-CPA setting (see Definition 3.3), as this not only breaks the scheme's pre-challenge quantum IND-ID-CPA security but also highlights that the FO-transform is not directly applicable to this scheme. Therefore, there exists no simple way of achieving chosen-ciphertext security for the proposed scheme (see Section A for the scheme's CCA security).

For our attack, we aim to exploit the homomorphism property of the secret isogeny $\phi_1 : \mathbb{G}_0 \to \mathbb{G}_1$. As per the definition of homomorphism, it must hold for any $g \in \mathbb{G}_0, a \in \mathbb{Z}$ that $\phi_1(a \cdot g) = a \cdot \phi_1(g)$. Intuitively, the attack consists of two steps:

1) select a random Identity $\texttt{ID}^*$ as the challenge Identity and generate its hash $h^* = H(\texttt{ID}^*)$,
2) solve the ECDLP using Shor's algorithm for some x, such that $h^* = [x]g_0$.

This might fail if $h^* \notin \langle g_0 \rangle$, however, since $\mathbb{G}_0$ is cyclic of prime order, this can only occur if g_0 is the neutral element of $\mathbb{G}_0$. As g_0 is drawn at random from all $q \in 2^{O(n)}$ elements in $\mathbb{G}_0$ during the IPGs generation, this only happens with negligible probability. In Remark 5.3, we give an intuition on handling this case. Once we have obtained x, it must now also hold that if $h^* = [x]g_0$ then

$$\phi_1(h^*) = \phi_1([x]g_0) = [x]\phi_1(g_0) = [x]g_1.$$

We thus simply multiply the public g_1 by x and obtain the Secret Key $\texttt{SK}_{\texttt{ID}^*}$ for $\texttt{ID}^*$ without having queried it to the extraction oracle. Thus, we can select $\texttt{ID}^*$ as the challenge Identity and decipher any ciphertext encrypted under $\texttt{ID}^*$ classically in the post-challenge phase.

Formally, we construct a pre-challenge quantum attacker in the OW-ID-CPA game against the scheme, as shown in the following two algorithms. We assume Shor's algorithm for the ECDLP to be available as a subroutine $ECDLPShor(\cdot)$. We further assume the point $g_0 \neq \mathcal{O}_0$, i.e., the point g_0 from the $\texttt{MPK}$ is not the neutral element of $\mathbb{G}_0$.

Algorithm 5 $\mathcal{A}_1$

Require: $1^\kappa, \texttt{MPK}, \mathcal{O}^{ext}(\cdot)$
1: $\texttt{ID}^* \xleftarrow{\$} \mathbb{F}_q$
2: $h^* := H(\texttt{ID}^*)$
3: $d \leftarrow ECDPLShor(\mathbb{G}_0, g_0, h^*)$
4: $\texttt{SK}_{\texttt{ID}} := g_1$
5: $\texttt{SK}_{\texttt{ID}^*} := [d]\texttt{SK}_{\texttt{ID}}$
6: **return** $\texttt{ID}^*$

Figure 7. Pre-Challenge Quantum Adversary $\mathcal{A}_1$.

Remark 5.3. *We want to highlight that the attack is still very much possible if $g_0 = \mathcal{O}_0$: instead of using g_0 as the basis for the ECDLP, we instead generate another random $\texttt{ID}$ and issue a key extraction query for this Identity. We can then use $g_0 := H(\texttt{ID})$ and $g_1 \leftarrow \mathcal{O}^{ext}(\texttt{ID})$ for the attack.*

The classical part of the adversary obtains the Secret Key $\texttt{SK}_{\texttt{ID}^*}$ from this first algorithm. The complete algorithm is defined in Algorithm 6.

It should be evident that this adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ succeeds with probability $\Pr[\text{OW-ID}^{cpa}_{\Pi,\mathcal{A}}(\kappa) = 1] = 1 - \frac{1}{q}$. This stems from the fact that solving the ECDLP using Shor's algorithm for the Hidden Subgroup Problem succeeds with probability $1 - \frac{1}{ord(h^*)}$ (see Section V-B) and $ord(h^*) = q$ since $\mathbb{G}_0$ is cyclic of prime order q (see Section II-C). By the correctness assumption, it must hold that

Algorithm 6 $\mathcal{A}_2$

Require: $\texttt{MPK}, \mathcal{O}^{ext}(\cdot), \texttt{ID}^*, C_{\texttt{ID}^*}$
1: $M \leftarrow \Pi.\texttt{decrypt}(\texttt{MPK}, \texttt{SK}_{\texttt{ID}^*}, C_{\texttt{ID}^*})$
2: **return** M

Figure 8. Post-Challenge Classical Adversary $\mathcal{A}_2$.

$\Pi.\texttt{decrypt}(\texttt{MPK}, \texttt{SK}_{\texttt{ID}}, \Pi.\texttt{encrypt}(\texttt{MPK}, \texttt{ID}, M)) = M$ with probability 1. Runtime-wise, we expect the hash function, random selection, and scalar multiplication of curve points to run in constant time $\tilde{O}(1)$ and Shor's algorithm to run in time $O(\log^2(q))$.

It is also noteworthy that we managed to reconstruct a key for an arbitrary Identity without even possessing a single key-pair ourselves, and thus without querying the extraction oracle.

We acknowledge that this attack cannot be executed in the authors' PH-PQ security model. Nonetheless, this attack shows, that when provided access to the hash function H, as would be in a practical setting, a pre-challenge quantum adversary could reconstruct any User's Secret Key and thus decrypt any ciphertext addressed to them efficiently with a classical algorithm.

D. Proposed Classical Attack

Besides the quantum algorithm breaking the scheme's OW-ID-CPA pre-challenge quantum security in polynomial time, it is also possible to construct a classical adversary attacking the scheme's OW-PH-PQ (see Definition 4.3) security in subexponential time and therefore challenging both claims presented in Section IV-B. The idea here is generally the same as for the previous attack: solve the ECDLP, reconstruct the User's Secret Key for the challenge Identity, and use this key to decipher the challenge ciphertext. While the ECDLP is usually considered to be difficult to solve using classical means, we can exploit another property of the proposed scheme.

When working with isogenies as trapdoor homomorphisms, one usually uses supersingular elliptic curves, as these provide a more complex, expander-like isogeny graph, making it quantum-hard to navigate the graph and find secret isogenies. Furthermore, the endomorphism ring for supersingular elliptic curves forms a non-commutative quaternion algebra, which is also quantum-hard to compute. It is proven that finding an isogeny between two elliptic curves is as hard as computing their endomorphism ring [7]. Hence, the authors also propose using supersingular elliptic curves.

Now, as we have shown that an attack against this scheme is basically an attack against the ECDLP, we are facing another property of supersingular elliptic curves: their embedding degree is always $k \leq 6$, making them prone to the MOV attack [18]: Menezes et al. present a method of reducing the ECDLP for supersingular elliptic curves to the DLP in $\mathbb{F}_{p^{2k}}$. In this setting, we are no longer restricted to the use of collision algorithms (Pollard's Rho), which boast exponential runtime-complexities, but can instead use subexponential time algorithms such as the Index Calculus method [19]. Because of such attacks as the one presented in the following, NIST also recommends against the usage of supersingular elliptic curves (or curves with a small embedding degree in general) for classical ECDLP-based cryptography [20].

The attack itself follows the same scheme as the quantum attack in Section V-C. However, as we cannot query the challenge Identity in the pre-challenge phase for PH-PQ security, we instead only select a random challenge Identity in the first phase.

Algorithm 7 $\mathcal{A}_1$

Require: $1^\kappa, \texttt{MPK}, \mathcal{O}^{ext}(\cdot)$
1: $\texttt{ID}^* \xleftarrow{\$} \mathbb{F}_\text{q}$
2: **return** $\texttt{ID}^*$

Figure 9. Pre-Challenge Classical Adversary $\mathcal{A}_1$.

In the second phase, we then perform the same scheme of attack as in Algorithm 5.

Algorithm 8 $\mathcal{A}_2$

Require: $\texttt{MPK}, \mathcal{O}^{ext}(\cdot), \texttt{ID}^*, C_{\texttt{ID}^*}$
1: $h^* \coloneqq H(\texttt{ID}^*)$
2: $d \leftarrow \text{MOV-DLP}(\mathbb{G}_0, g_0, h^*)$
3: $\texttt{SK}_{\texttt{ID}} \coloneqq g_1$
4: $\texttt{SK}_{\texttt{ID}*} \coloneqq [d]\texttt{SK}_{\texttt{ID}}$
5: $M \leftarrow \Pi.\texttt{decrypt}(\texttt{MPK}, \texttt{SK}_{\texttt{ID}^*}, C_{\texttt{ID}^*})$
6: **return** M

Figure 10. Post-Challenge Classical Adversary $\mathcal{A}_2$.

To solve the ECDLP, instead of Shor's quantum algorithm for the ECDLP, we instead apply the MOV-reduction, described in Algorithm 9.

Algorithm 9 MOV-DLP

Require: $\mathbb{G}_0$ of order q on curve E_0, two points $g_o, h^* \in \mathbb{G}_0$
1: Determine k, c from [18, Table 1.]
2: $g' \xleftarrow{\$} E_0\left(\mathbb{F}_{p^{2k}}\right)$
3: $g \coloneqq \frac{c \cdot ord(g')}{q} \cdot g'$
4: $\alpha \coloneqq e_q(h^*, g)$
5: $\beta \coloneqq e_q(g_0, g)$
6: $x \leftarrow \text{NFS-DL}(\alpha, \beta)$
7: **return** x

Figure 11. DLP Algorithm using the MOV-Reduction.

Here, e_q denotes the Weil pairing on the q-torsion group of E_0. First, the ECDLP is reduced to the DLP in $\mathbb{F}_{p^{2k}}$ through the use of the Weil pairing. We then use the Number Field Sieve for Discrete Logarithms (NFS-DL) described in [21] from the Index Calculus family as a subroutine to solve the DLP in $\mathbb{F}_{p^{2k}}$. We expect the runtime of all operations except the NFS-DL to be in $\tilde{O}(1)$, while the Index Calculus boasts a

complexity of $L_{p^{2k}}\left(1/3, \sqrt[3]{64/9}\right)$. Since $q \in O(p)$, we can also write this as $L_{q^{2k}}\left(1/3, \sqrt[3]{64/9}\right)$, which is subexponential in the size of $\mathbb{G}_0$ for small values of k, such as those for supersingular elliptic curves.

The algorithm has a success probability of $\Pr[\text{OW-ID}^{cpa}_{\Pi,\mathcal{A}}(\kappa) = 1] = 1 - \frac{1}{q}$. The algorithm always succeeds unless $ord(\alpha) < q$ and the probability $\Pr[ord(\alpha) = q] = \frac{\varphi(q)}{q}$ since there are $\varphi(q) = q - 1$ many elements of order q in $\mathbb{F}_{p^{2k}}$ and there are q cosets of $\langle g_0 \rangle$ in $E[q]$.

VI. Conclusion

In this paper, we provided detailed pre-challenge quantum attacks in both the OW-ID-CPA setting and the OW-PH-PQ setting on the IBE scheme based on IPGs proposed by Koshiba and Takashima. By leveraging the fact that the scheme's security reduces to the ECDLP on supersingular elliptic curves, we were able to break the scheme in polynomial time $O(\kappa^2)$ using quantum algorithms in the OW-ID-CPA game and subexponential time $L_{p^{2k}}[1/3, \sqrt[3]{64/9}], k \leq 6$ classically in the OW-PH-PQ game. We argue that IPGs contain an inherent structural weakness in any (pre-challenge) quantum scenario: the pairings used in IPGs require the use of explicit group elements and cannot be directly applied to curves as a whole. However, while isogenies between supersingular elliptic curves provide quantum secure properties for cryptography, mapping these explicit points between the isogenous groups using isogenies does not necessarily. Since the groups are cyclic of prime order, all the elements are related to one another through the elliptic curve discrete logarithm, as all elements, apart from the neutral element, are generators of $\mathbb{G}_t$. Meanwhile, through the homomorphism property of isogenies between elliptic curves, and since $ord(\mathbb{G}_t) = q$ for all $t \in [0, d]$, all the IPGs must also be isomorphic to one another. This means any adversary capable of solving the ECDLP can determine the relation of two public points $\phi_t(P), \phi_t(Q) \in \mathbb{G}_t$ (here we consider ϕ_0 to be the identity function) and thus also the relation of their origin points' P, Q images under the isogenies ϕ'_t in every pairing group $\mathbb{G}_{t'}$ isogenous to $\mathbb{G}_t$. Hence, if a single secret isogenous point to a public point is known, every secret point in the same group for every public point can be calculated as well using Shor's algorithm on a quantum computer. Classically, the required use of supersingular elliptic curves for the isogenies also weakens the scheme's resistance against an attack on the discrete logarithm in these groups significantly.

References

[1] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes," in *Advances in Cryptology*, G. R. Blakley and D. Chaum, Eds., Berlin, Heidelberg: Springer, 1985, pp. 47–53, ISBN: 978-3-540-39568-3. DOI: 10.1007/3-540-39568-7_5.

[2] D. Boneh and M. Franklin, "Identity-Based Encryption from the Weil Pairing," in *Advances in Cryptology — CRYPTO 2001*, J. Kilian, Ed., Berlin, Heidelberg: Springer, 2001, pp. 213–229, ISBN: 978-3-540-44647-7. DOI: 10.1007/3-540-44647-8_13.

[3] T. Koshiba and K. Takashima, "Pairing Cryptography Meets Isogeny: A New Framework of Isogenous Pairing Groups," 2016, Accessed: Feb. 13, 2025. [Online]. Available: https://eprint.iacr.org/2016/1138, pre-published.

[4] D. Boneh and X. Boyen, "Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles," in *Advances in Cryptology - EUROCRYPT 2004*, C. Cachin and J. L. Camenisch, Eds., Berlin, Heidelberg: Springer, 2004, pp. 223–238, ISBN: 978-3-540-24676-3. DOI: 10.1007/978-3-540-24676-3_14.

[5] L. De Feo, "Mathematics of Isogeny Based Cryptography," Nov. 11, 2017, arXiv: 1711.04062 [cs], pre-published.

[6] S. D. Galbraith and F. Vercauteren, "Computational problems in supersingular elliptic curve isogenies," *Quantum Information Processing*, vol. 17, no. 10, p. 265, Oct. 2018, ISSN: 1570-0755, 1573-1332. DOI: 10.1007/s11128-018-2023-6.

[7] B. Wesolowski, "The supersingular isogeny path and endomorphism ring problems are equivalent," in *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, Feb. 2022, pp. 1100–1111. DOI: 10.1109/FOCS52979.2021.00109.

[8] J. H. Silverman, *The Arithmetic of Elliptic Curves* (Graduate Texts in Mathematics). New York, NY: Springer New York, 2009, vol. 106, ISBN: 978-0-387-09494-6. DOI: 10.1007/978-0-387-09494-6.

[9] E. Fujisaki and T. Okamoto, "Secure Integration of Asymmetric and Symmetric Encryption Schemes," *Journal of Cryptology*, vol. 26, no. 1, pp. 80–101, Jan. 1, 2013, ISSN: 1432-1378. DOI: 10.1007/s00145-011-9114-1.

[10] Y. Yacobi, "A Note on the Bilinear Diffie-Hellman Assumption," 2002, Accessed: Jul. 16, 2025. [Online]. Available: https://eprint.iacr.org/2002/113, pre-published.

[11] M. Andersch, C. Pilaszewicz, and M. Margraf, "A Note on the Post-Quantum Security of Identity-Based Encryption on Isogenous Pairing Groups," 2025, Accessed: Feb. 18, 2026. [Online]. Available: https://eprint.iacr.org/2025/1439, pre-published.

[12] J. M. Pollard, "A monte carlo method for factorization," *BIT Numerical Mathematics*, vol. 15, no. 3, pp. 331–334, Sep. 1, 1975, ISSN: 1572-9125. DOI: 10.1007/BF01933667.

[13] D. Johnson, A. Menezes, and S. Vanstone, "The Elliptic Curve Digital Signature Algorithm (ECDSA)," *International Journal of Information Security*, vol. 1, no. 1, pp. 36–63, Aug. 1, 2001, ISSN: 1615-5262. DOI: 10.1007/s102070100002.

[14] R. Haakegaard and J. Lang, "The Elliptic Curve Diffie-Hellman (ECDH)," [Online]. Available: http://koclab.cs.ucsb.edu/teaching/ecc/project/2015Projects/Haakegaard+Lang.pdf.

[15] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, Oct. 1997, ISSN: 0097-5397. DOI: 10.1137/S0097539795293172.

[16] S. Hallgren, A. Russell, and A. Ta-Shma, "Normal subgroup reconstruction and quantum computation using group representations," in *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, ser. STOC '00, New York, NY, USA: Association for Computing Machinery, May 1, 2000, pp. 627–635, ISBN: 978-1-58113-184-0. DOI: 10.1145/335305.335392.

[17] J. Proos and C. Zalka, "Shor's discrete logarithm quantum algorithm for elliptic curves," *Quantum Info. Comput.*, vol. 3, no. 4, pp. 317–344, Jul. 1, 2003, ISSN: 1533-7146.

[18] A. Menezes, S. Vanstone, and T. Okamoto, "Reducing elliptic curve logarithms to logarithms in a finite field," in *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing - STOC '91*, New Orleans, Louisiana, United States: ACM Press, 1991, pp. 80–89, ISBN: 978-0-89791-397-3. DOI: 10.1145/103418.103434.

[19] L. Adleman, "A subexponential algorithm for the discrete logarithm problem with applications to cryptography," in *20th Annual Symposium on Foundations of Computer Science (Sfcs 1979)*, Oct. 1979, pp. 55–60. DOI: 10.1109/SFCS.1979.2.

[20] L. Chen, D. Moody, A. Regenscheid, A. Robinson, and K. Randall, "Recommendations for Discrete Logarithm-based Cryptography: Elliptic Curve Domain Parameters," National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-186, Feb. 3, 2023, NIST SP 800–186. DOI: 10.6028/NIST.SP.800-186.

[21] R. Barbulescu, "Algorithms for discrete logarithm in finite fields," Ph.D. dissertation, Université de Lorraine, Dec. 5, 2013. Accessed: Jul. 30, 2025. [Online]. Available: https://hal.univ-lorraine.fr/tel-01750438.

APPENDIX

A. Another Note on CCA security

While the authors do not claim the scheme to be CCA secure, we still want to show a trivial attack breaking the cipher, when provided access to a decryption oracle. Notably, we choose to present this attack in the One-way ID-CCA setting, as breaking this weaker security notion also breaks the Indistinguishability by implication.

We construct the following classical attacker $\mathcal{A}$ in the OW-ID-CCA game:

Algorithm 10 $\mathcal{A}$, ask phase

Require: $1^\kappa, \mathtt{MPK}, \mathcal{O}^{ext}(\cdot), \mathcal{O}^{dec}(\cdot)$

1: $\mathtt{ID}^* \xleftarrow{\$} \mathbb{F}_q$
2: **return** $\mathtt{ID}^*$

Figure 12. Pre-Challenge Phase Algorithm for Adversary $\mathcal{A}$.

In the pre-challenge phase, we simply choose a random Identity to attack. The challenger will then select a challenge plaintext M^* uniformly at random, encrypt it and send the resulting challenge-ciphertext back to the adversary $\mathcal{A}$ in the post challenge phase described in Algorithm 11.

Assuming the hash function H to run in $O(1)$, the attack boasts a runtime-complexity of $\tilde{O}(1)$. The attack's correctness

Algorithm 11 $\mathcal{A}$, $guess$ phase

Require: $\mathtt{MPK}, \mathcal{O}^{ext}(\cdot), \mathcal{O}^{dec}(\cdot), C_{\mathtt{ID}^*} = (C^*, C_T^*)$

1: $h_0 \coloneqq H(\mathtt{ID}^*)$
2: $\overline{C} \coloneqq C^* \cdot \hat{g}_1$
3: $\overline{C}_T \coloneqq C_T^* \cdot e_0(h_0, \hat{g}_0)$
4: $\overline{M} \leftarrow \mathcal{O}^{dec}((\overline{C}, \overline{C}_T))$
5: **return** $\overline{M}$

Figure 13. Post-Challenge Phase Algorithm for Adversary $\mathcal{A}$.

can be proven by observing the following equality:

$$\begin{aligned}
\overline{M} &= \overline{C}_T \cdot (e_1(h_1, \overline{C}))^{-1} = \overline{C}_T \cdot (e_1(h_1, C \cdot \hat{g}_1))^{-1} \\
&= \overline{C}_T \cdot (e_1(h_1, \hat{g}_1^\zeta \cdot \hat{g}_1))^{-1} = \overline{C}_T \cdot (e_1(h_1, \hat{g}_1^{\zeta+1}))^{-1} \\
&= \overline{C}_T \cdot (e_1(h_1, \hat{g}_1)^{\zeta+1})^{-1} = \overline{C}_T \cdot (e_1(\phi_1(h_0), \hat{g}_1)^{\zeta+1})^{-1} \\
&= \overline{C}_T \cdot (e_0(h_0, \hat{g}_0)^{\zeta+1})^{-1} \\
&= C_T^* \cdot e_0(h_0, \hat{g}_0) \cdot (e_0(h_0, \hat{g}_0)^{\zeta+1})^{-1} \\
&= M^* \cdot e_0(h_0, \hat{g}_0)^\zeta \cdot e_0(h_0, \hat{g}_0) \cdot (e_0(h_0, \hat{g}_0)^{\zeta+1})^{-1} \\
&= M^* \cdot e_0(h_0, \hat{g}_0)^{\zeta+1} \cdot (e_0(h_0, \hat{g}_0)^{\zeta+1})^{-1} = M^*.
\end{aligned}$$

Furthermore, all the used variables, namely $h_0, \hat{g}_0, \hat{g}_1$, are publicly available. As we assume h_i and $\hat{g}_i$ to stem from distinct groups, the bilinear pairing will never degenerate to 1. We assume that $\hat{g}_1 \neq \hat{\mathcal{O}}_1$, where $\hat{\mathcal{O}}_1$ denotes the neutral element of $\hat{\mathbb{G}}_1$, as well as $e_0(h_0, \hat{g}_0) \neq 1$.in either case, the original scheme would break: the first would result in $\hat{g}_1^\zeta = \hat{g}_1$, therefore $e_1(h_1, C) = e_1(h_1, \hat{g}_1) = e_0(h_0, \hat{g}_0)$, which allows the calculation of z' and thus M without knowledge of ζ, while the latter would directly encode $C_T = M$ and therefore leak the plaintext. It must then always follow, that $(C^*, C_T^*) \neq (\overline{C}, \overline{C}_T)$; hence the decryption oracle never rejects the decryption query. This attack therefore succeeds with probability $\Pr[\text{OW-ID}^{cca}_{\Pi,\mathcal{A}}(1^\kappa) = 1] = 1$.

We acknowledge that ID-CCA-security could, in fact, be achieved by applying modifications such as the FO-transform [9] to the proposed scheme if it were OW-ID-CPA secure, as shown by Boneh and Franklin in [2]; a property which we dispute in Section V.

A Multi Method Framework for GNSS Anomaly Detection in Vehicular Systems Using NMEA Data

Mathias Gerstner*, Tobias Reichel†, Sebastian Fischer*, Rudolf Hackenberg*
*Dept. Informatics and Mathematics, OTH Regensburg
Regensburg, Germany
e-mail: {mathias.gerstner, | sebastian.fischer, | rudolf.hackenberg}@oth-regensburg.de
†Central Office for Information Technology in the Security Sector
Munich, Germany
e-mail: tobias.reichel@zitis.bund.de

***Abstract*—Intended interference with satellite signals, including jamming and spoofing, has become increasingly common and poses a growing threat to safety-critical applications. In the automotive domain, this development creates a strong demand for reliable and lightweight detection mechanisms that can be deployed on standard vehicle platforms. This paper presents an Intrusion Detection System (IDS) designed to detect such attacks on common Global Navigation Satellite System (GNSS) receivers in road vehicle environments. The proposed system operates as a software solution and processes live, recorded, or simulated navigation data streams in real time. It integrates threshold checks, rule-based consistency analysis, and a Machine Learning (ML) algorithm together with a score fusion method to generate anomaly scores without requiring specialized hardware or labeled attack data. The system is adjusted and validated using real-world driving data combined with synthetic attack scenarios. Experimental results demonstrate reliable detection of diverse attack patterns while maintaining a low false positive rate under normal driving conditions, showing the suitability of the approach for automotive GNSS security monitoring.**

Keywords-gnss; nmea; autonomous driving; intrusion detection system; spoofing.

I. Introduction

Global Navigation Satellite System (GNSS), such as Global Positioning System (GPS) or Galileo, are fundamental components of modern navigation and localization applications. They are widely used in domains like aviation, maritime transportation, and automated driving to provide position, velocity, and precise timing information [1, pp. 10–14]. However, the open and largely unencrypted nature of civilian GNSS signals makes them vulnerable to cyberattacks such as jamming and spoofing, where counterfeit signals mislead receivers into reporting false navigation data [2]. As discussed in previous work, manipulated GNSS data can have significant safety, security, and forensic implications, particularly when navigation information is blindly trusted by systems [3].

To address these threats, GNSS-specific Intrusion Detection Systems (IDSs) have been proposed to monitor navigation data streams and identify anomalies indicative of attacks or system malfunctions [4]. Existing research has often focused on maritime environments or cooperative multi receiver setups exploiting spatial diversity. While these approaches can provide robust detection capabilities, they often require additional hardware or infrastructure support that limits their area of application [5]. In future autonomous driving scenarios, it will be important to have lightweight, configurable IDSs suitable for standard GNSS receivers commonly deployed in road vehicles.

This paper presents a framework designed to detect anomalies in National Marine Electronics Association (NMEA)-based navigation data streams for road vehicles. The system supports multiple input sources, including real-time data from physical GNSS receivers, replayed historical recordings, and simulated data generated by simulations, such as IPG CarMaker [6]. This design enables reproducible testing of attack scenarios and facilitates integration into real-time applications.

Anomaly detection is performed using parallel analysis methods. A threshold module evaluates navigation parameters against fixed bounds obtained from Exploratory Data Analysis (EDA) of real-world driving data. A rule-based component checks physical and logical consistency constraints, such as acceleration limits, velocity changes, or implausible short term deviations. In addition, a Machine Learning (ML)-based detector uses a Local Outlier Factor (LOF) algorithm to identify outliers in the navigation parameters without requiring labeled attack data.

The outputs of the individual detection modules are combined by a score fusion mechanism that computes a total anomaly score reflecting the presence and severity of detected irregularities. This modular architecture allows for a systematic evaluation of individual detection strategies while remaining suitable for deployment on automotive systems with limited computational resources.

Beyond standalone operation, the proposed IDS is intended to serve as a component of a forensic framework. Specifically, it is planned to be integrated into a Forensic Incident Recorder (FIR), currently under development, which extends the functionality of the Data Storage System for Automated Driving (DSSAD) [7]. Based on the severity level reported by the IDS, the FIR selectively triggers the recording of additional sensor data, such as speedometer or camera information, to balance storage constraints with forensic requirements in the event of suspected GNSS behavior [8].

The remainder of this paper is structured as follows. Section II reviews related work on GNSS intrusion detection. Section III describes the collection and exploratory analysis

of navigation data. Synthetic attack generation is presented in Section IV, while Section V details the system architecture and implementation. Section VI discusses the evaluation methodology and results. Section VII shows the system's capabilities and limitations, and Section VIII summarizes the findings and outlines directions for future work.

II. Related Work

Due to the increasing relevance of reliable satellite-based navigation, a growing amount of research addresses the detection of attacks on GNSS systems. Existing approaches include rule-based techniques as well as ML methods. However, a substantial part of the proposed solutions focuses on maritime applications or depends on specialized hardware. In contrast, lightweight and modular IDS architectures suitable for road vehicles remain underrepresented in the literature.

Amro et al. [9] propose an anomaly detection system for NMEA messages that relies on rules identifying violations of physical limits and timing constraints. ML methods were initially considered but discarded due to insufficient performance caused by limited training data. In contrast, the approach presented in this paper extends similar rule-based concepts by incorporating an unsupervised ML layer and fusing all detector outputs.

Boudehenn et al. [10] developed an IDS that operates on a Raspberry Pi and processes NMEA data. Their system employs a ML model to detect spoofing and jamming attacks. While computationally efficient for specific attack classes, it is tailored to maritime scenarios and relies on a single detection mechanism, limiting its adaptability to dynamic road vehicle environments.

Spravil et al. [11] introduced the MAritime Nmea based Anomaly detection (MANA), which detects GPS spoofing through the fusion of multiple consistency checks. However, MANA relies exclusively on predefined thresholds and rules, whereas the proposed architecture extends this concept by integrating anomaly detection through ML as a third detector.

Lemieszewski [12] proposed a physical consistency checking approach that compares the speed reported by a GNSS receiver with the vehicle's speedometer. This rule-based method demonstrated successful spoofing detection in real-world driving experiments but may fail for attacks that closely mimic realistic vehicle dynamics. In contrast, the IDS in this paper operates solely on GNSS data, enabling deployment without dependencies on additional vehicle sensors.

The developed system addresses these limitations by emphasizing the use of real-world driving data for analysis and detector calibration, complemented by controlled synthetic manipulations. By processing GNSS data in real time and integrating multiple detection strategies, the IDS provides a practical solution for automotive GNSS security monitoring without much overhead.

III. Exploratory Data Analysis

An EDA was conducted to find the statistical properties of the GNSS data and to identify feature correlations relevant for anomaly detection. The analysis was based on 50 historical driving recordings collected on a fixed suburban route with a length of 71 km. The measurement drives were conducted on the road B16 in Germany during a time window between March and December 2025. The receiver used is the u-blox ZED-F9R with the active antenna u-blox ANN-MB-00-00. All drives were performed under normal operating conditions without intentional interference, providing a baseline of expected GNSS behavior. The resulting insights were used to derive threshold values, define detection rules, and select suitable input features for the ML detection component.

Correlation analysis of all captured, unaltered data points revealed several strong linear dependencies among some GNSS features. In particular, the Dilution of Precision (DOP) metrics showed strong positive correlations. Similarly, position uncertainty parameters, including the major and minor axes of the error ellipse, as well as latitude- and longitude-related standard deviations, were highly correlated with each other and with the DOP metrics. These results were to be expected, but must be taken into account when selecting parameters [13].

In contrast, dynamic motion features such as vehicle speed and course exhibited only weak correlations with signal quality parameters. This indicates that motion behavior provides independent information that is not directly influenced by the satellite constellation quality. Consequently, these features are suitable for detecting anomalies characterized by implausible dynamics, such as unrealistic accelerations or abrupt direction changes, even when signal quality indicators appear nominal.

Feature correlations play a crucial role in the selection of inputs for the ML detector. Highly correlated features introduce redundancy into the feature space and can negatively affect the performance of ML algorithms [14]. To mitigate this, only weakly or moderately correlated features were selected for the LOF algorithm, ensuring a balanced representation of independent behavioral and signal related characteristics.

In addition to correlation analysis, descriptive statistics of all driving data were collected for all GNSS parameters. Mean values, standard deviations, and percentiles were calculated to characterize normal operating ranges. These statistics form the base for threshold detection and help to reduce false positives caused by normal environmental variability. The thresholds used in the IDS are based on the data provided by the mentioned specific hardware. Therefore, they have to be adjusted if another GNSS receiver and antenna are used.

IV. NMEA Data Manipulation

To enable systematic evaluation of the IDS, an offline GNSS anomaly generation framework was developed that operates directly on recorded NMEA data stored in Comma Separated Values (CSV) format. The internal processing workflow, including EDA and model training, is illustrated in Figure 1. By manipulating existing recordings instead of generating fully synthetic navigation data, realistic temporal behavior and noise characteristics of real GNSS measurements are preserved.

In addition to offline manipulation, controlled spoofing experiments were conducted in a shielding tent to study NMEA

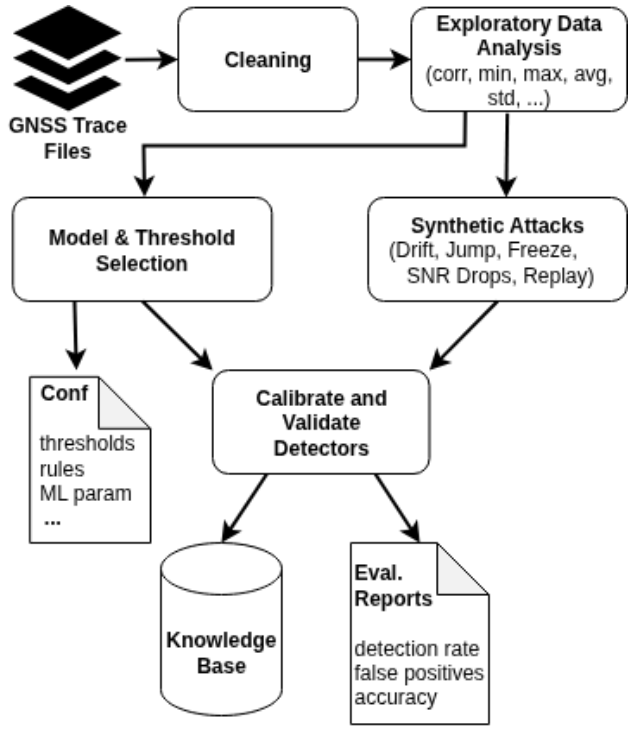


Figure 1. Overview of the EDA and model training process.

behavior under real spoofing conditions. Using a Software Defined Radio (SDR), counterfeit GNSS signals with incorrect satellite constellations were transmitted to a receiver. These experiments revealed effects such as temporary loss of valid navigation data, position freezes, and slow positional drifts. Shortly after the start of spoofing, NMEA sentences often contained empty or invalid fields before stable spoofed position and time information was created by the receiver. Since this behavior is comparatively easy to detect, offline manipulation assumes a near-ideal spoofing scenario with smooth transitions and no data gaps.

In the following, different effects of spoofing were applied to the NMEA data to emulate attacks and fault conditions. A position jump attack introduces a sudden spatial displacement by shifting all NMEA sentences containing geographic location by a constant, configurable offset starting at a defined time. The modified coordinates are converted back into NMEA sentences and the checksums are fitted to the new payload.

A drift attack models a drag-off spoofing scenario by gradually increasing the positional offset over a specified time interval. Each affected NMEA sentence is shifted by a time-dependent fraction of the maximum configured offset of 100 m, resulting in a smooth and continuous deviation from the true trajectory.

For a position freeze attack, all NMEA sentences containing position information are overwritten with a fixed location equal to the last valid position before the attack onset, while other fields, such as time, continue to evolve normally. This manipulation emulates a receiver that appears stationary despite ongoing movement.

Signal quality manipulation, as it may occur during jamming, is realized through Signal to Noise Ratio (SNR) degradation. Within a defined time window, NMEA GSV sentences are modified by identifying satellite SNR fields and scaling them by a configurable factor.

Finally, replay attacks are considered, representing a practical threat requiring minimal knowledge of receiver internals. In this scenario, authentic GNSS signals are captured using an SDR and later retransmitted. To increase realism and detection difficulty, an advanced replay strategy is assumed in which the captured signals are not simply replayed as captured. If the timestamps show an earlier time than the real time, such attacks are easy to detect because time cannot run backwards. Therefore, the timestamps in the replayed messages are set to the actual time, avoiding obvious temporal inconsistencies.

Together, these manipulation techniques provide a diverse set of attack scenarios that support evaluation of the detection capabilities of the GNSS IDS [15].

V. System Architecture

The developed IDS combines multiple complementary detection strategies to identify spoofing and jamming anomalies. Instead of relying on a single detection mechanism, the system integrates thresholds, rules, and ML approaches and aggregates their outputs in a score fusion module. An overview of the architecture is shown in Figure 2.

If NMEA data is produced by a simulation or a real GNSS receiver, it is usually embedded in other software. Therefore, a Message Queuing Telemetry Transport (MQTT) client is implemented to fetch this data from a broker, which acts as the interface between data producing components and the IDS itself. Reports or alerts generated by the IDS can also be sent back to other software via the MQTT interface.

In this study, we used GNSS trace files to train and validate the system's capabilities. Regardless of the NMEA data source, it is first stored in a queue for raw data. A processing module then extracts the relevant information from the NMEA records and places it into a separate queue for each detector. The detectors can then retrieve the current GNSS data from these queues and evaluate it using various methods. If a detector identifies an anomaly, it calculates a score depending on the severity of the anomaly. These scores are combined into a total anomaly score, which is mapped to a continuous severity scale ranging from 0 to 1, enabling graded alerting instead of binary decisions.

A. Threshold Detection

Threshold violation detection is the simplest and most computationally efficient component of the IDS. It relies on static upper or lower bounds derived from physical constraints and statistical properties observed during the EDA. Monitored parameters include geometry related quality indicators, position uncertainty estimates, satellite availability, SNR, and vehicle speed. The thresholds used in this work are summarized in

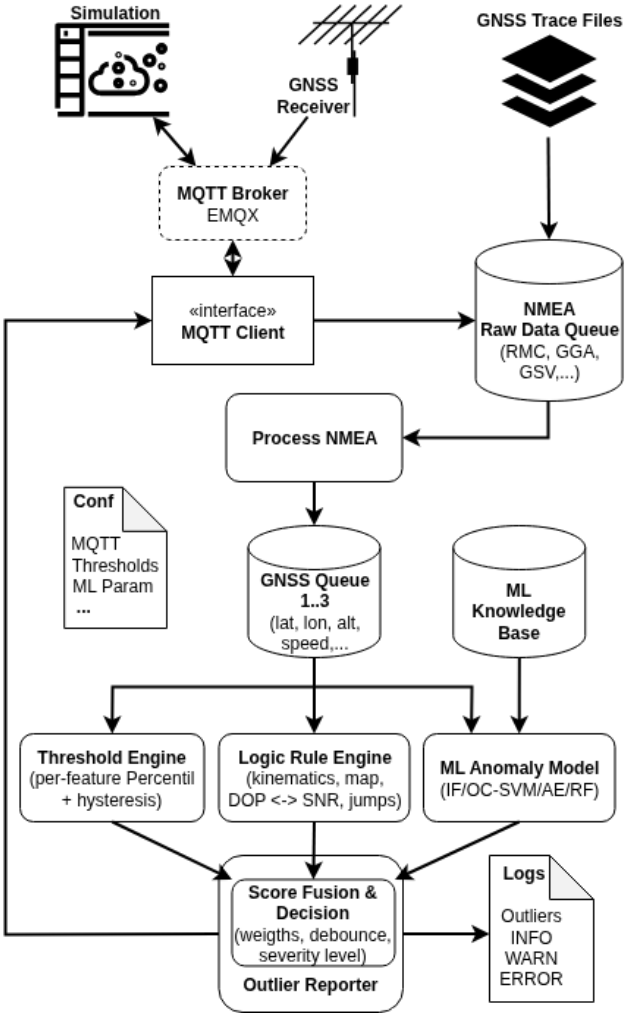


Figure 2. Internal System Architecture of the GNSS IDS.

Table I. The speed on the testing route is 100 km/h, so an alert limit of 120 km/h was chosen for an anomaly report.

For all metrics, additional overshoot thresholds are defined to capture extremely unrealistic values. The thresholds are defined by the extrema observed across all 50 drives, i.e., the maximum for some parameters and the minimum for others. Violations of these overshoot thresholds result in increased severity contributions, indicating a higher probability of malicious manipulation. Threshold checks are evaluated on a per epoch basis at 1 Hz, enabling immediate detection of very noticeable anomalies. While this approach lacks contextual awareness and may produce false positives in challenging environments such as urban canyons or tunnels, it provides an effective first line of defense.

B. Rule Detection

The rule-based detection module evaluates higher-level physical and logical consistency constraints within the GNSS data stream. Instead of analyzing individual parameters, it considers temporal and relational properties. The implemented rules are:

TABLE I. THRESHOLDS USED FOR ANOMALY DETECTION

Metric	Threshold	Overshoot Threshold
PDOP	> 2.03	> 22.54
HDOP	> 1.13	> 19.94
VDOP	> 1.71	> 22.01
RMS range	$> 45.0\,\mathrm{m}$	$> 135.0\,\mathrm{m}$
Std. major axis	$> 10.0\,\mathrm{m}$	$> 199.0\,\mathrm{m}$
Std. minor axis	$> 5.7\,\mathrm{m}$	$> 66.0\,\mathrm{m}$
Std. latitude	$> 4.0\,\mathrm{m}$	$> 109.0\,\mathrm{m}$
Std. longitude	$> 2.5\,\mathrm{m}$	$> 147.0\,\mathrm{m}$
Std. altitude	$> 6.7\,\mathrm{m}$	$> 500.0\,\mathrm{m}$
GPS satellites	< 9	$< 7 \vee > 17$
Mean SNR	$< 22.7\,\mathrm{dB}$	$< 5.0\,\mathrm{dB}$
SNR std. dev.	$< 6.2\,\mathrm{dB}$	$< 1.5\,\mathrm{dB}$
Max. speed	$> 120\,\mathrm{km/h}$	$> 160\,\mathrm{km/h}$

- Lack of incoming data for more than 5 seconds
- Acceleration (positive or negative) exceeding 14 m/s^2
- Position jump exceeding 80 m/s
- Yaw rate exceeding 60°/s (unrealistic steering maneuver)
- Jump between two epochs in the mean SNR over all satellites exceeding 12 dB

The threshold values were determined based on an EDA conducted over all 50 drives, which correspond to a total driving distance of 3,350 km.

This detector is effective at identifying structured inconsistencies that may not exceed individual thresholds but violate expected motion or signal behavior. However, like most deterministic approaches, it can be evaded by well-executed gradual attacks, such as slow drag-off spoofing, that remain within predefined limits over extended periods.

C. Machine Learning Detection

The third detector is based on ML. It uses a LOF algorithm that learns normal system behavior by analyzing density patterns in the parameters. The approach compares the current 20 seconds with the last 200 seconds and therefore does not need labeled attack data.

The detector operates on a rolling window of recent GNSS observations. Incoming epochs are evaluated at a frequency of 5 seconds. For each evaluation cycle, the most recent data is split into a training subset representing nominal behavior and a smaller test subset that is assessed for anomalies. All features are standardized based on the training data. This sliding window enables the detector to adapt to slowly varying environmental and operational conditions while remaining sensitive to short term deviations.

The selected feature set contains the delta of latitude and longitude (previous position to current location), speed, position DOP, major axis standard deviation, number of visible satellites, standard deviation of SNR, and the Root Mean Square (RMS) satellite range. Strongly correlated parameters are always represented by just one of them. By analyzing these parameters, the LOF detector captures dependencies that cannot be reliably addressed by threshold checks or fixed rules.

Anomalies are identified when test samples exhibit significantly lower local density compared to their neighborhood in the learned feature space [16]. To reduce sensitivity to minor fluctuations, only anomalies exceeding a predefined severity threshold of -5.0 are reported. Detected events are then sent to the fusion and reporting modules of the system.

The main limitation of this module is a moderate detection latency introduced by the window-based processing, which represents an acceptable trade-off for increased detection reliability.

D. Score Fusion

The final stage of the IDS architecture is a parallel operating score fusion module. Instead of binary decisions, the system aggregates, weights and normalizes the outputs of all detection layers. The fusion process produces a continuous total anomaly score, which is mapped to a severity value between 0 and 1. In the current implementation, an anomaly is reported when the total score exceeds 0.5, with all parameters configurable via a central configuration file.

This fused representation improves robustness against false positives from individual detectors and enables a balanced reaction strategy. Low severity anomalies trigger early warnings, while high confidence events supported by multiple detection layers escalate to critical alerts. For integration into the planned FIR, the continuous severity score will be mapped to discrete levels (LOW, MEDIUM, HIGH, and CRITICAL) to control severity dependent response and data recording behavior.

VI. Validation with Real World Data

The performance of the IDS is validated using real-world driving data. Two evaluation scenarios are considered: (i) detection performance under synthetic attack conditions and (ii) false positive behavior under nominal, attack-free operation.

A. Detection Performance

As described in Section IV, synthetic spoofing and jamming attacks are generated and injected into recorded NMEA files. These manipulated datasets are processed by the IDS to evaluate its ability to detect anomalous behavior.

Figure 3 shows the resulting anomaly scores for a representative drive containing different attacks. The IDS raises elevated anomaly scores during attack phases, while maintaining low scores during nominal operation outside the manipulated intervals.

A short detection latency is observed for slow drifts and location freeze attacks, as it is inherently difficult to distinguish between a legitimate vehicle stop and malicious manipulation. One false positive alert can be seen at the right border of the graphic, caused by the fast decrease of DOP values.

B. False Positive Analysis

In addition to detection capability, false positive behavior is a key performance metric for an IDS. To evaluate this aspect, the system was applied to a dataset containing 50 real-world driving runs on a rural road.

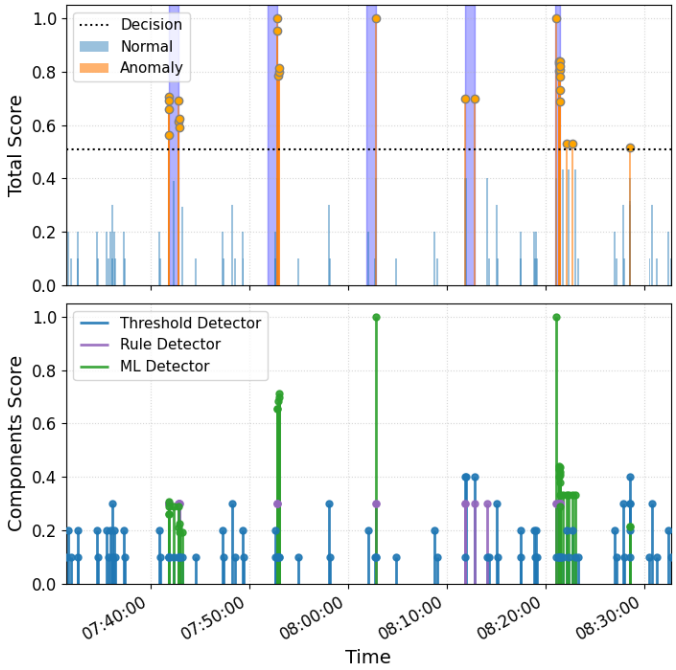


Figure 3. IDS anomaly total and per component scores during different spoofing and jamming attacks. Blue marks show the phases of attack. Orange marks show detected anomalies. From left to right the attacks are: position jump, slow drift attack, location freeze attack, jamming (SNR drop), replay attack.

Figure 4 presents the anomaly scores for the same route shown in Figure 3, but without injected attacks. The IDS maintains low anomaly scores throughout the drive, with only one single false alert observed, showing robustness during normal operation while remaining sensitive to abrupt inconsistencies.

For a systematic analysis, all driving runs were evaluated. Since the dataset contains no synthetic attacks, all detected anomaly events are treated as false positives. Consecutive anomaly alerts are counted as a single event until the system reports a normal state at least once. Each driving run comprises approximately 4000 GNSS epochs with 1 Hz sampling and a distance of 71 km.

TABLE II. Performance of the GNSS IDS evaluated on 3550 km of drives.

Metric	False positives per hour
Mean false positive rate	2.27
Median false positive rate	1.80
Maximum false positive rate	10.80

As shown in Table II, the system exhibits low false positive activity under normal conditions, with less than three events per hour on average. Analysis of runs with higher rates indicates that false positives are primarily caused by degradations in GNSS signal quality. The Table III contains the individual features and detectors that were most frequently involved in false-positive anomaly alerts. In total, 126 alerts were raised over all test drives. Within these, the threshold validation of the mean SNR was involved in 54 of them.

 ISBN: 978-1-68558-442-9

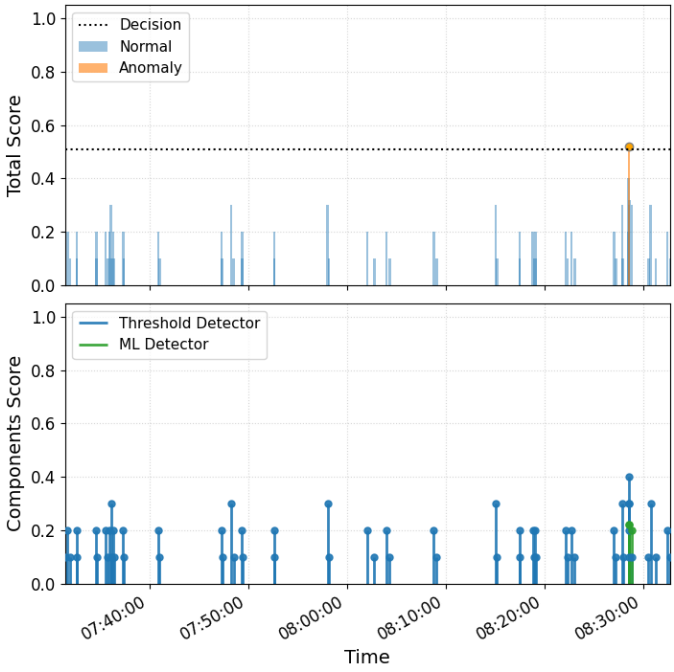


Figure 4. IDS anomaly scores under normal driving conditions without attacks.

TABLE III. MOST FREQUENT CAUSES OF FALSE POSITIVE EVENTS ACROSS ALL ATTACK FREE RUNS.

Detector and Feature	Occurrences
Threshold: mean SNR	54
LOF: RMS range	52
LOF: major axis position error	51
LOF: Position DOP	47
Threshold: RMS range	41

These results indicate that the system achieves a good balance between detection sensitivity and robustness against false alerts. While near perfect synthetic attacks are reliably detected, the false positive rate remains low under real-world driving conditions.

VII. DISCUSSION

Overall, the results show several practical challenges in the implementation of an IDS for navigation data. For example, rapid degradations of SNR or DOP values are not always associated with malicious interference. These effects frequently occur with specific road segments, such as loops, bridges, or forests, where signals are shadowed by obstacles. This behavior shows the importance of considering environmental influences when interpreting anomaly detections.

During the adjustment of the system's parameters, one task was the removal of the satellite SNR standard deviation from the rule-based detector. This feature is naturally correlated with the mean SNR which leads to simultaneous violations of both rules during signal degradations. As a result, these correlated reactions significantly increased the false positive rate.

By removing the standard deviation rule, the overall median false positive rate was reduced from 2.52 to 1.80 events per hour. However, this improvement came at the cost of reduced sensitivity in the jamming attack scenario. The overall severity score during this attack decreased from 1.0 to 0.7. This is still enough to raise an alert, but with a less clear statement. This illustrates the problematic trade-off between maximizing attack detection and maintaining an acceptable false positive rate. An alternative approach to handling SNR false positives could be the use of temporal gradient features. In particular, the rate of change of the SNR may provide a more discriminative indicator for abrupt interference events such as jamming while being less sensitive to naturally low but stable signal conditions.

The results further suggest that adding contextual information could help reduce false positives. For example, the use of digital maps with known influencing objects such as bridges, dense vegetation, or urban structures could allow the IDS to adapt its sensitivity locally. Such context-aware mechanisms could reduce alerts caused by predictable, non-malicious signal degradations without globally lowering detection rates.

It is also noteworthy that conventional performance metrics, such as the F1 score, were not applied in this study. This decision was made because it typically requires reliable ground truth for attacks, which is not available for real-world driving data. Instead, the false positive rate per fixed driving distance under normal conditions was chosen as the primary robustness metric. This approach provides more meaningful insight into system behavior in a real-world environment where attacks are rare and unknown, but too many false alerts can significantly limit practical usability.

VIII. CONCLUSION AND FUTURE WORK

The presented anomaly detection system for road vehicle environments operates on standard NMEA data. While the detection of anomalies works well, there are still a small number of false positives in data without attacks. Therefore, future work will focus on incorporating contextual information, such as additional vehicle dynamics, to verify plausible motion. Another promising concept is the integration of map and environmental data to further reduce these false positives and improve robustness against highly sophisticated spoofing attacks.

ACKNOWLEDGMENT

This work was supported by the project 'Digital Forensics in IT Systems (DiForIT)', funded by the German Federal Ministry for Economic Affairs and Climate Action (BMWK).

REFERENCES

[1] E. D. Kaplan and C. J. Hegarty, Eds., *Understanding GPS/GNSS: Principles and Applications*, 3rd ed. Boston, MA, USA: Artech House, 2017.

[2] T. E. Humphreys, B. M. Ledvina, M. L. Psiaki, B. W. O'Hanlon, and P. M. Kintner, "Assessing the spoofing threat: Development of a portable GPS civilian spoofer", in *Proc. ION GNSS Conf.*, 2008.

[3] T. Reichel et al., "A forensic analysis of GNSS spoofing attacks on autonomous vehicles", in *Proc. 16th Int. Conf. Cloud Computing, GRIDs, and Virtualization, Valencia, Spain*, Apr. 2025, pp. 32–39.

[4] A. Mohanty and G. Gao, "A survey of machine learning techniques for improving global navigation satellite systems", *EURASIP J. Adv. Signal Process.*, vol. 2024, no. 1, p. 73, 2024. DOI: 10.1186/s13634-024-01167-7.

[5] M. L. Psiaki and T. E. Humphreys, "GNSS spoofing and detection", *Proc. IEEE*, vol. 104, no. 6, pp. 1258–1270, Jun. 2016. DOI: 10.1109/JPROC.2016.2526658.

[6] IPG Automotive GmbH, *CarMaker – Vehicle dynamics simulation*, [Online]. Available: https://www.ipg-automotive.com/solutions/product-portfolio/carmaker. Accessed: Feb. 17, 2026.

[7] United Nations Economic Commission for Europe (UNECE), "Guidance on data storage system for automated driving (DSSAD) and EDR", UNECE WP.29 GRVA, Tech. Rep., 2025, [Online]. Available: https://unece.org/sites/default/files/2025-06/GRVA-22-23e.pdf. Accessed: Dec. 18, 2025.

[8] K. Dolos et al., "Forensic readiness for autonomous mobility: The forensic incident recorder and information system concept", *Forensic Sci. Int.: Digit. Investig.*, vol. 56, Art. no. 302044, Mar. 2026. DOI: 10.1016/j.fsidi.2026.302044.

[9] A. Amro, A. Oruc, V. Gkioulos, and S. Katsikas, "Navigation data anomaly analysis and detection", *Information*, vol. 13, no. 3, p. 104, 2022. DOI: 10.3390/info13030104.

[10] C. Boudehenn, O. Jacq, M. Lannuzel, J.-C. Cexus, and A. Boudraa, "Navigation anomaly detection: An added value for maritime cyber situational awareness", in *Proc. Int. Conf. Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, Jun. 2021, pp. 1–4. DOI: 10.1109/CyberSA52016.2021.9478189.

[11] J. Spravil, C. Hemminghaus, M. von Rechenberg, E. Padilla, and J. Bauer, "Detecting maritime GPS spoofing attacks based on NMEA sentence integrity monitoring", *J. Mar. Sci. Eng.*, vol. 11, no. 5, p. 928, 2023. DOI: 10.3390/jmse11050928.

[12] L. Lemieszewski, "Transport safety: GNSS spoofing detection using the single-antenna receiver and the speedometer of a vehicle", *Procedia Comput. Sci.*, vol. 207, pp. 3181–3188, 2022. DOI: 10.1016/j.procs.2022.09.375.

[13] W. Li et al., "GDOP and the CRB for positioning systems", *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E100-A, no. 2, pp. 733–737, 2017. DOI: 10.1587/transfun.E100.A.733.

[14] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection", *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, 2003.

[15] Spirent Federal, "GNSS signal spoofing: How to evaluate the risks to safety-critical and liability-critical systems", Spirent Federal, Tech. Rep. DWP0014, Issue 1-02, 2020, [Online]. Available: https://spirentfederal.com/wp-content/uploads/Federal-DWP0014-Issue-1-02-GNSS-Signal-Spoofing-min.pdf, Accessed: Feb. 17, 2026.

[16] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers", *SIGMOD Rec.*, vol. 29, no. 2, pp. 93–104, May 2000. DOI: 10.1145/335191.335388.

CVEs With a CVSS Score Greater Than or Equal to 9

Lena Sinterhauf [1,2], Andreas Aßmuth [2], and Roland Kaltefleiter [1]
[1] NetUSE AG, Kiel, Germany
e-mail: {lsi | rk}@netuse.de
[2] Kiel University of Applied Sciences, Kiel, Germany
e-mail: andreas.assmuth@haw-kiel.de

***Abstract*—Critical vulnerabilities with Common Vulnerability Scoring System scores of 9.0 or higher pose severe risks to organisations' information systems. Timely detection and remediation are essential to minimise economic and reputational damage from cyberattacks. This paper provides a thorough analysis of the identification and resolution timelines of such critical vulnerabilities. A mixed-methods approach is employed, integrating quantitative data from global vulnerability databases analysing 245,456 Common Vulnerabilities and Exposures records spanning from 2009 to 2024, of which 12.8 % were critical, with qualitative case studies of notable incidents. This methodical combination of quantitative and qualitative data sources enables the identification of patterns and delay factors in vulnerability management. The findings indicate significant delays in public disclosure and patch deployment, influenced by industry-specific factors, resource availability and organisational processes. The paper concludes with a series of actionable recommendations to improve the efficiency of vulnerability responses. Despite faster disclosure, the remediation gap for critical vulnerabilities remains a systemic risk, driven by organisational inertia and system complexity.**

Keywords-critical vulnerabilities; vulnerability detection time; vulnerability management; patch management.

I. Introduction

The increasing digitisation and interconnectivity of organisations and critical infrastructures has led to a growing threat landscape dominated by cyberattacks. Information security constitutes a central component within contemporary corporate strategies, with the objective of preserving the integrity, availability, and confidentiality of data. The standardised identification and documentation of security vulnerabilities is facilitated by the Common Vulnerabilities and Exposures (CVE) system [1]. The assessment of vulnerabilities is conducted through the utilisation of the Common Vulnerability Scoring System (CVSS), a method that employs a scale ranging from 0 (lowest) to 10 (highest) to evaluate the severity of the vulnerabilities identified [2]. Vulnerabilities that receive a score of 9.0 or higher are designated as critical, given the substantial risks they pose to the affected systems.

Despite the implementation of established security processes, the timely detection and remediation of critical vulnerabilities remain challenging. The failure to disclose vulnerabilities or the failure to deploy patches in a timely manner can expose organisations to significant risks of exploitation, resulting in financial losses, operational disruptions and reputational damage [3][4]. Notable incidents, such as the Log4Shell vulnerability, have underscored the pressing need for effective and efficient vulnerability management [5]–[7].

The present study investigates the speed and efficiency of identifying and remediating critical vulnerabilities, focusing on those with CVSS scores of 9.0 or above. It synthesises quantitative analyses of vulnerability data from global databases spanning 2009 to 2024 with qualitative case studies of major security incidents. The objective of this study is to identify the factors that contribute to delays in vulnerability response and to provide actionable recommendations for improving the resilience of IT infrastructures against critical security threats.

This study provides a long-term analysis of critical vulnerabilities (CVSS ≥ 9.0) across 245,456 CVE records over a period of 16 years (2009 to 2024). Contrary to previous studies, this research combines large-scale quantitative analysis with qualitative case studies of major incidents (Heartbleed, EternalBlue, and Log4Shell) to identify systemic delays in vulnerability remediation and examines sector-specific patch patterns across more than 20 industries, thus providing practical insights for the prioritisation of vulnerability remediation in the context of limited security resources.

The structure of this paper is as follows: Section II provides a comprehensive review of the extant literature on vulnerability management and scoring systems. Section III delineates the research methodology. The fourth section of this text presents the results of the data analyses and case studies. The subsequent section, Section V, discusses the implications of these findings. Finally, Section VI concludes with a summary and suggestions for future research.

II. Related work

Research on software vulnerabilities has addressed the subjects of detection, severity assessment, and remediation. CVSS a widely utilised numerical classification system for vulnerability criticality, which serves to guide the prioritisation of remediation efforts [2][8]. Service Level Agreements (SLAs) have been proposed as a means of defining remediation timelines. It is recommended that critical vulnerabilities be addressed within days to weeks [9][10].

Empirical studies have analysed vulnerability lifecycles, management frameworks, open-source processes, and metrics, such as mean time to remediate and disclosure-to-patch delays [11][12]. The extant literature consistently highlights challenges in the timely remediation of issues, which are influenced by system complexity, organisational readiness, and resource constraints.

Research focusing explicitly on critical vulnerabilities reports that, despite improvements in disclosure speed, patch

TABLE I. RELATED WORK ON CVE/CVSS ANALYSIS (2009 TO 2025)

Year	Title	Author(s)	Topics
2019	Practical patch management and mitigation	S. Alexiou	Patch SLAs, remediation timelines [10]
2022	Guide to enterprise patch management planning	M. Souppaya, K. Scarfone	MTTR metrics, enterprise patching [11]
2025	To patch or not to patch	J.R.C. Nurse	Patching motivations, organisational challenges [12]
2025	The secret life of CVEs	P. Przymus et al.	CVE lifecycle analysis [13]
2025	Out of sight, still at risk	P. Przymus et al.	Transitive vulnerabilities, Maven ecosystem [14]

deployment often lags behind, thereby extending exposure windows and exploitation risk [13][14]. Despite the existence of regulatory and sector-specific guidelines that advocate for faster responses, the efficacy of patching varies across sectors and vendors.

Key studies on vulnerability management and CVE lifecycles have been summarised in Table I. Whilst the extant literature provides insights into patching processes and vulnerability lifecycles, none offer a long-term (2009 to 2024) analysis focused on critical vulnerabilities (CVSS ≥ 9.0) across sectors, combined with qualitative case studies of high-impact incidents. The present study addresses this gap by integrating large-scale quantitative data with detailed case analyses to uncover patterns, delays, and factors unique to the highest severity vulnerabilities.

III. METHODS

The present study employs a mixed-methods approach in order to comprehensively analyse the identification and remediation processes of critical security vulnerabilities. The methodology integrates quantitative analysis of vulnerability data with qualitative case studies to gain both breadth and depth of understanding.

The quantitative component employs data from recognised vulnerability databases, namely the National Vulnerability Database (NVD) and the MITRE CVE database [15][16]. The data were obtained from the official NVD JSON 2.0 feeds (as of 28 May 2025) and MITRE CVE list downloads (as of 20 May 2025). The datasets were processed using Python scripts (e.g., *pandas*, *json*) to filter vulnerabilities with CVSS base scores of ≥ 9.0, to calculate temporal metrics (e.g., days from *reservation_date* to *published* and to *lastModifiedDate* as patch proxy), and to aggregate results by year, assignee (assigners), and sector.

The analysis encompasses $245{,}456$ CVE records registered between January 2009 and December 2024, of which $31{,}430$ (approx. $12.8\,\%$) were classified as critical with CVSS base scores of 9.0 or higher. Two primary temporal dimensions were examined: the duration from CVE reservation to public disclosure, and the duration from disclosure to patch availability (approximated using database modification timestamps). Statistical analysis was conducted to examine these timelines, identify trends and patterns over time, and reveal discrepancies across industry sectors and software categories.

The qualitative element of the study comprises in-depth case studies of notable security incidents, including the Heartbleed, EternalBlue, and Log4Shell vulnerabilities [6][17][18]. The case studies presented offer insights into the challenges encountered by organisations in the field of vulnerability management, including issues, such as delays, organisational factors, and best practices in mitigation.

The integration of quantitative and qualitative data facilitates cross-validation and more profound interpretation of results. Quantitative findings reveal statistical trends and potential delay factors, while qualitative analysis contextualises these findings within actual incident scenarios and management practices.

The process of data validation entailed several key steps. Firstly, database entries were subjected to rigorous cross-checking. Secondly, the results were meticulously compared with existing literature to ensure the reliability of the findings. Finally, consistency was maintained throughout all analysis phases. This integrated approach facilitates the development of pragmatic recommendations that are designed to enhance the efficiency and effectiveness of critical vulnerability management.

IV. RESULTS

The analysis encompasses vulnerability data from 2009 to 2024, with a particular emphasis on critical vulnerabilities that have a severity score of 9.0 or higher. The results of the study reveal three primary dimensions of interest: detection and publication timelines, patch availability delays, and organisational or sectoral variation in response times.

A. Detection and Publication Timelines

The total number of registered vulnerabilities increased substantially over the study period, particularly after 2016 when the CVE assignment process was expanded through the introduction of CVE Numbering Authorities (CNAs) [19][20], as illustrated in Figure 1.

In this context, “registrations” denote the initial allocation of a CVE ID by MITRE or designated CNAs upon internal vulnerability reporting, prior to public disclosure [21][22]. “Publications” refer to the subsequent public release of detailed

vulnerability information in databases, such as MITRE and the NVD, making it visible to the global security community [21]–[23].

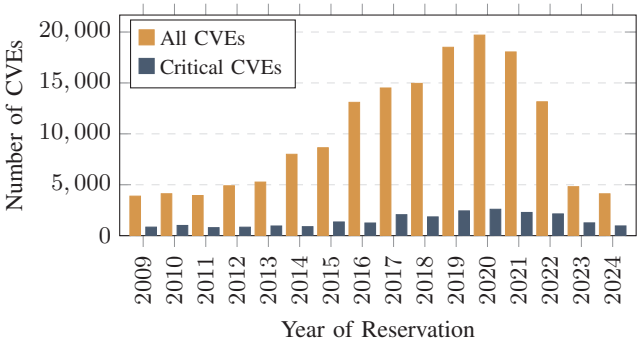


Figure 1. Annual distribution of CVE registrations (2009–2024). Orange bars represent total no. of CVEs, while blue bars show no. of critical vulnerabilities.

This phenomenon resulted in a marked increase in the publication of vulnerabilities in 2017 and again during the period of the pandemic caused by the virus known as SARS-CoV-2 (2020–2021), when working remotely and accelerated digitisation led to a greater number of exposed systems (cf. Figure 2).

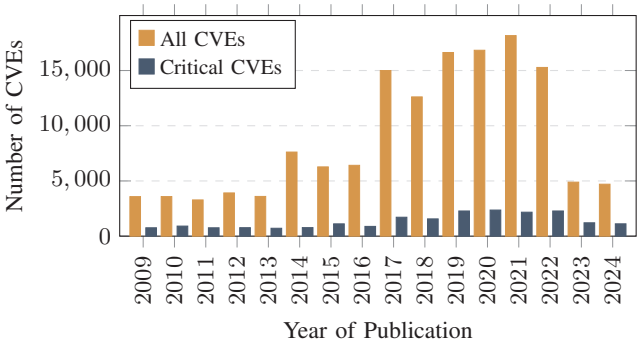


Figure 2. Annual distribution of CVE publications (2009–2024). Orange bars represent total no. of CVEs, while blue bars show no. of critical vulnerabilities.

The proportion of critical vulnerabilities remained relatively stable throughout the observation period at approximately 12.8 % of all registered CVEs, with a peak of 2,589 cases recorded in 2020. As illustrated in Figure 3, the average time from CVE reservation to public disclosure has decreased dramatically, from over 400 days in 2013 to approximately 33 days in 2024.

Notably, while critical vulnerabilities were published significantly faster than the overall average in earlier years (e.g., 57 vs. 105 days in 2009), this gap has virtually disappeared in recent years, indicating that systematic improvements in disclosure processes now benefit all vulnerability severity levels equally. This convergence represents a positive development in the CVE ecosystem. Nevertheless, the CVSS remains essential for severity classification and prioritisation, particularly when organisations face resource constraints or must process large

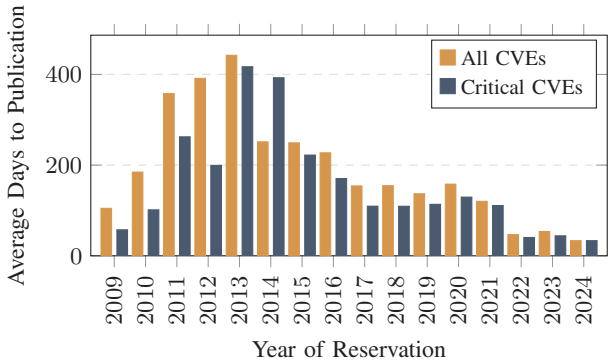


Figure 3. Average time from CVE reservation to public disclosure (2009–2024). Orange bars represent all CVEs, while blue bars show critical vulnerabilities.

numbers of vulnerabilities simultaneously, making it advisable to address critical issues first.

Our investigations also revealed that the time between registration and publication of a vulnerability varies considerably. While in some cases, assigners published registered CVEs on the same day (duration 0 days), in other cases it took up to several years. Of course, the reasons for these enormous differences in time are not apparent from the data available to us. In cases where the time span is very short, immediate publication is usually due to already known or simultaneously published vulnerabilities that were subsequently assigned a CVE ID. However, short time spans also indicate that some organisations appear to have particularly efficient processes in place, possibly automated disclosure procedures or internal Standard Operating Procedures (SOPs) that prioritise rapid publication. Long durations do not automatically mean that poor work was done in these cases. Reasons for this can also include complex coordination processes, late discovery of the actual impact, or subsequent publication of confidential vulnerabilities [23][24].

Notably, none of the assigners have an average publication time of 0 days for critical CVEs. This indicates that critical vulnerabilities are always subjected to at least a brief review before they are made public. Furthermore, it can be observed that the longest average delay for critical CVEs (approx. 850 days) is significantly shorter than for all CVEs (over 2,300 days). This suggests that critical vulnerabilities are generally processed and published more quickly, even when delays occur. At the same time, the data shows that the variance in duration for critical CVEs is lower, suggesting increased process standardisation or prioritisation.

B. Time to Patch Availability

Patch deployment analysis demonstrates that turnaround times are both longer and more variable than those observed in the publication phase. The mean time to release a patch for general vulnerabilities was approximately 1,732 days (median: 1,335 days), whereas for critical vulnerabilities it was around 2,024 days (median: 1,668 days).

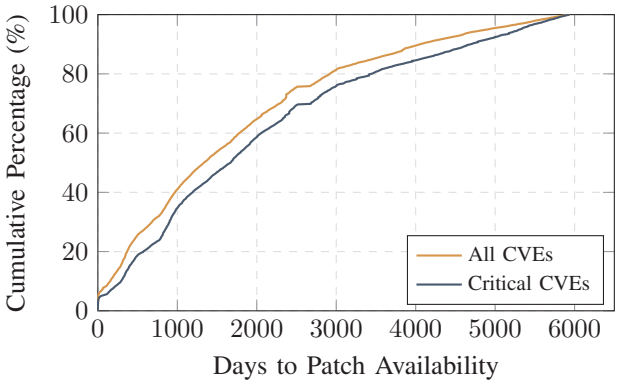


Figure 4. Cumulative distribution of time to patch availability (2009–2024). The curves show the percentage of CVEs patched within a given timeframe. Mean values: approximately 1,732 days (all CVEs) and 2,024 days (critical CVEs).

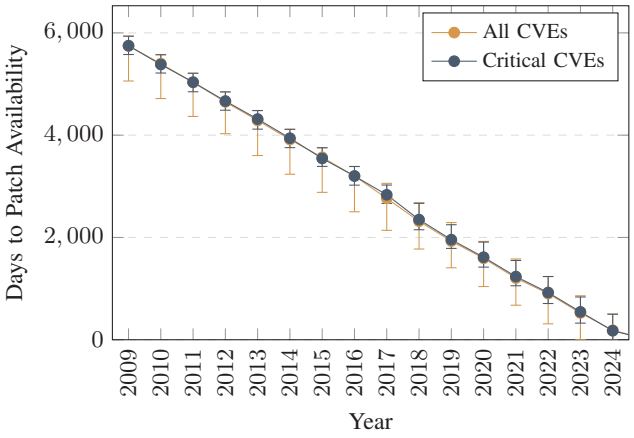


Figure 5. Patch availability for all CVEs and critical CVEs (2009–2024). The average, minimum and maximum durations are specified for each year. In the trend lines, the orange line indicates all CVEs, while the blue line indicates critical CVEs.

Figure 4 illustrates the cumulative distribution of time to patch availability across the entire observation period. Notably, 50 % of all CVEs received patches within 1,335 days of disclosure, while 90 % were patched within 4,054 days. For critical vulnerabilities, the corresponding values were 1,668 and 4,689 days. The distribution demonstrates that despite prioritisation efforts, critical vulnerabilities exhibit longer remediation times on average than the general population, likely reflecting the increased complexity and coordination requirements associated with high-severity issues. The highly skewed distribution, with substantial differences between median and mean values, indicates that while the majority of vulnerabilities are addressed within reasonable timeframes, a significant tail of delayed patches persists across both categories.

Despite the improvements that have been made, no consistent or significant difference has been demonstrated between critical and non-critical issues with regard to patch completion times, as can be seen in Figure 5. The apparent improvement in recent years should be interpreted with caution due to right-censoring: vulnerabilities from 2022 onwards have had less opportunity to exhibit extended patch delays, and currently unpatched vulnerabilities are not represented in these measurements.

Many organisations implement release cycles that are similar across all severity levels. The phenomenon of extended exposure periods can be attributed to various factors, including the increasing complexity of systems, the presence of legacy dependencies, constraints in resources, and the incomplete automation of processes. However, between 2020 and 2024, an observable acceleration occurred, suggesting stronger regulatory and procedural pressure on vendors.

C. Organizational and Sectoral Variations

There are notable disparities between different assignees (assigners). It is evident that certain entities, including specialised security platforms and prominent open-source providers, attained a median patch time of less than five days. This is indicative of sophisticated automation and continuous integration processes. In contrast, slower assigners – which were often commercial software vendors — exhibited median delays of between 2,000 and 4,000 days (cf. Figure 6).

Sectors were assigned by mapping CVE assigners to primary industry classifications using a reproducible, rule-based keyword matching on assigner names, with overlaps between sectors (e.g., Open Source, Commercial Software, Web & Content Management) resolved through a predefined prioritisation order. While this heuristic facilitates large-scale mapping, multi-sector entities and evolving business models may introduce classification uncertainties.

TABLE II. NUMBER OF ALL AND CRITICAL CVES BY SECTOR.

Sector	All CVEs	Critical CVEs
Cloud & Hosting	3700	263
Commercial Software	41583	4658
Consulting & Research	82386	15744
Consumer Electronics	55	9
Education & Non-Profit	33	4
Finance & Insurance	66	2
Hardware	13474	1276
Healthcare	22	10
Industrial & IoT	2493	308
Open Source	28699	2751
Other	51873	4055
Platforms & DevOps	108	1
Security Vendors	7249	848
Telecommunication & Networking	13620	1476
Web & Content Management	95	25

While sample sizes vary considerably across sectors (Table II), the median-based analysis remains robust for identifying general remediation patterns, with smaller sectors (e.g., Healthcare, $n = 22$) providing indicative trends rather than definitive benchmarks.

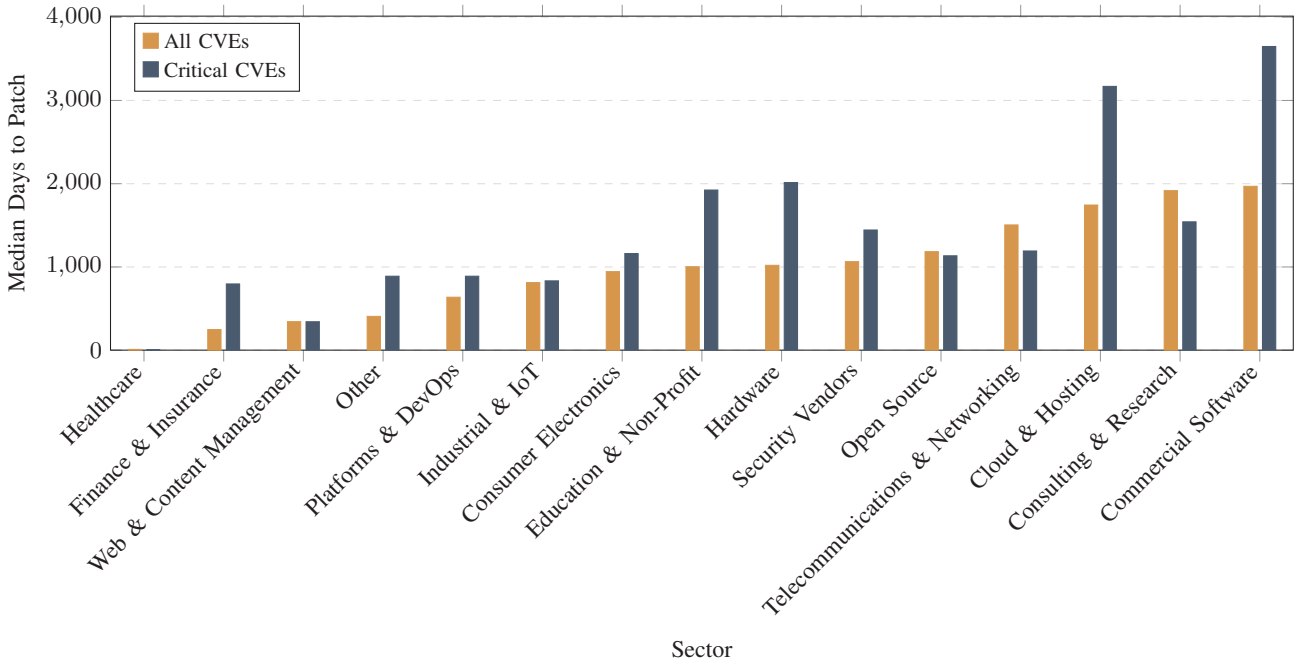


Figure 6. Median time to patch availability by sector (2009 to 2024). Sectors are sorted by overall patch performance (all CVEs). Orange bars represent all CVEs, while blue bars show critical vulnerabilities, revealing significant performance disparities across industries.

A further indication of this is provided by a comparison of performance across different sectors, which shows that open-source communities and cloud providers generally remediate faster than traditional industries, such as commercial software or hardware manufacturing. It is particularly evident in the healthcare, energy, and telecommunications sectors that there is a tendency for shorter patch cycles, which is likely attributable to the presence of more stringent legal and compliance requirements [25][26].

D. Case Study Insights

The quantitative findings are reinforced by qualitative case studies of three landmark critical vulnerabilities (CVSS ≥ 9.0), illustrating real-world manifestation of detection, disclosure, and remediation patterns observed across the 2009-2024 dataset.

The Heartbleed vulnerability (CVE-2014-0160), which was discovered in OpenSSL in April 2014, enabled attackers to read up to 64 KB of server memory, with the potential to expose private keys, passwords and session data for millions of systems worldwide [17]. Despite the rapid provision of patches within two days, the delays between disclosure and remediation in the affected companies averaged more than six months. This was due to widespread dependency on the open-source library and lack of automated detection tools. This case study highlights the challenges associated with the "last mile" of patch deployment, emphasising the necessity for dependency scanning and automated updating within open-source ecosystems to mitigate prolonged exposure windows, which align with the quantitative medians (1,668 days for critical CVEs).

EternalBlue (CVE-2017-0144), which was exploited in the WannaCry ransomware of May 2017, targeted Windows SMBv1 protocol flaws, affecting unpatched Windows systems globally and causing damages in excess of $4 billion [18]. Microsoft released a patch in March 2017 (pre-disclosure), yet six months post-disclosure, 20 % of organisations remained vulnerable, thereby amplifying the subsequent ransomware outbreak. The incident demonstrates the protracted nature of remediation processes in commercial software sectors, extending beyond the established quantitative averages. This emphasises the necessity for regulatory mandates to enforce patch SLAs in critical infrastructures, such as healthcare and telecommunications.

The Log4Shell vulnerability (CVE-2021-44228), which was disclosed in December 2021 in Apache Log4j, enabled remote code execution via malicious logging inputs. This had a significant impact on more than 3 billion devices and triggered immediate zero-day exploits [5]–[7]. Patches were issued within days; however, full remediation took weeks due to supply-chain propagation and configuration complexity, with global adoption lagging as documented in BSI warnings [6]. This finding underscores the existence of persistent organisational delays, thereby signifying the necessity for continuous vulnerability management as opposed to periodic scans, as evidenced by the study's comprehensive patterns.

The collective analysis of these cases serves to reinforce the quantitative patterns identified, thereby illustrating how technical complexity, organisational inertia, and sectoral variations collectively drive the remediation gaps.

V. Discussion And Evaluation

The results of this study highlight both significant progress and persistent structural challenges in the management of critical software vulnerabilities.

Over the period under scrutiny, the time lag between CVE reservation and public disclosure decreased significantly, reaching an average of approximately 33 days in 2024. This phenomenon points to an enhancement in the coordination and responsiveness of the global vulnerability management ecosystem. The expansion of the CNA programme and enhanced collaboration between security researchers, vendors, and vulnerability databases have likely contributed to this acceleration [19][27]. The acceleration of disclosure processes has been demonstrated to increase transparency and enable organisations to initiate defensive measures earlier.

Despite these improvements, the findings demonstrate that faster disclosure does not necessarily lead to faster remediation. The analysis indicates that remediation timelines persistently exceed expectations and demonstrate considerable variability, with a median duration of 1,668 days for critical vulnerabilities. This discrepancy underscores a systemic "last-mile" problem in vulnerability management, where the primary bottleneck shifts from vulnerability discovery to the development and deployment of patches [10].

The existence of this discrepancy can be attributed to a number of factors. In the contemporary context, software systems frequently employ complex dependency chains and interconnected components, a factor that has been shown to complicate the processes of patch development and testing. Moreover, organisational constraints, such as limited resources, operational risks associated with updates, and fragmented asset inventories, have the potential to delay patch deployment, particularly in large or legacy environments.

Sectoral comparisons provide further support for these observations. It is evident that open-source ecosystems and cloud-based platforms frequently demonstrate a higher level of responsiveness, largely attributable to the utilisation of automated development pipelines and continuous integration practices. Conversely, traditional commercial vendors characteristically implement more extended release cycles. It has been demonstrated that regulated sectors, including but not limited to healthcare and telecommunications, exhibit accelerated remediation, a phenomenon that is presumably precipitated by regulatory incentives.

Case studies, including those of Heartbleed, EternalBlue and Log4Shell, illustrate that even when patches are released promptly, organisations frequently require a considerable amount of time to identify affected systems and deploy updates across complex infrastructures. The findings emphasise that effective vulnerability mitigation is contingent not only on vendor response, but also on organisational preparedness and the implementation of mature patch management processes.

The findings indicate an enhancement in vulnerability management with regard to transparency and disclosure efficiency. Nevertheless, the extended remediation timelines underscore a persistent "last-mile" issue in the implementation of patch deployment. In order to address this challenge, it is necessary to implement not only faster vulnerability reporting but also improved automation, better asset visibility, and more mature patch management processes within organisations.

VI. Conclusion and Future Work

The present study examined the processes of detection, disclosure, and remediation of critical vulnerabilities that had severity scores of nine or higher. Integration of data-driven analysis and qualitative case evaluations enabled identification of both structural improvements and persistent challenges in contemporary vulnerability management.

The findings indicate that global disclosure timelines have become considerably reduced, signifying a maturation of the ecosystem of coordinated vulnerability reporting and enhanced cross-organisational communication [24][27]. Nevertheless, the remediation phase continues to demonstrate deficiencies, with notable heterogeneity in patch release times across software vendors and sectors. These delays, frequently attributable to resource constraints, legacy dependencies, and fragmented responsibilities, result in organisations remaining vulnerable for extended periods following the disclosure of vulnerabilities [10].

This work contributes to extant research by quantifying the systemic inefficiencies that persist despite procedural advances. It is also important to note that effective vulnerability management is not just a technical problem, but also a governance challenge [25]. In order to address this challenge, there is a need for synchronised policy, automation and human expertise. Organisations that actively integrate regulatory frameworks, establish prioritised workflows and mandate security accountability are better positioned to reduce the window between discovery and mitigation.

From an applied perspective, this study underscores the necessity for organisations to accord priority to critical vulnerabilities (CVSS ≥ 9.0) through the implementation of established SLAs (cf. Section II) and sector-specific strategies (Subsection IV-C), while concomitantly addressing systemic remediation delays that have been identified across the period 2009 to 2024 (see Figures 4, 5, 6), particularly in commercial software sectors that require a longer timeframe to remediate (cf. Figure 6).

In future research, the exploration of machine learning–driven models, such as exploit prediction scoring systems, in the refinement of prioritisation in dynamic threat environments is recommended. Further investigation into the following areas would be of use in attempting to bridge the gap between awareness and action: automated remediation pipelines; CI/CD-integrated patching; and multi-source vulnerability aggregation [28]. Furthermore, emerging paradigms, such as exposure management and zero-trust architectures offer promising frameworks for the unification of vulnerability management across hybrid infrastructures.

The study indicates that, while the speed of identifying and publishing critical vulnerabilities is improving, sustainable cybersecurity resilience depends on transitioning from reactive

vulnerability management to proactive, continuous exposure governance.

ACKNOWLEDGEMENT

The present paper is founded upon Lena's Master Thesis, which was conducted at the Faculty of Computer Science and Electrical Engineering, Kiel University of Applied Sciences.

REFERENCES

[1] MITRE Corporation, "Frequently asked questions (faqs) - what is cve?", Accessed: 2026-03-14. [Online]. Available: https://www.cve.org/ResourcesSupport/FAQs.

[2] National Institute of Standards and Technology (NIST), "Vulnerability metrics", Accessed: 2026-03-14. [Online]. Available: https://nvd.nist.gov/vuln-metrics/cvss#.

[3] Ponemon Institute, "Cost of a data breach report 2024", 2024, Accessed: 2026-03-14. [Online]. Available: https://table.media/wp-content/uploads/2024/07/30132828/Cost-of-a-Data-Breach-Report-2024.pdf.

[4] Bundesamt für Sicherheit in der Informationstechnik, "The state of it security in germany in 2023 (original title in german: Die Lage der IT-Sicherheit in Deutschland 2023)", 2023, Accessed: 2026-03-14. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Lageberichte/Lagebericht2023.pdf.

[5] National Institute of Standards and Technology (NIST), "CVE-2021-44228 detail", Accessed: 2026-03-14. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2021-44228.

[6] Bundesamt für Sicherheit in der Informationstechnik, "Critical vulnerability published in log4j (cve-2021-44228) (original title in german: Kritische Schwachstelle in log4j veröffentlicht (CVE-2021-44228))", Accessed: 2026-03-14. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Cybersicherheitswarnungen/DE/2021/2021-549032-10F2.pdf?__blob=publicationFile&v=5.

[7] CrowdStrike Intelligence Team, "Log4j2 vulnerability "log4shell" (CVE-2021-44228)", 2021, Accessed: 2026-03-14. [Online]. Available: https://www.crowdstrike.com/en-us/blog/log4j2-vulnerability-analysis-and-mitigation-recommendations/.

[8] Forum of Incident Response and Security Teams (FIRST), "Common vulnerability scoring system v4.0 – user guide", Accessed: 2026-03-14. [Online]. Available: https://www.first.org/cvss/v4.0/user-guide.

[9] Bundesamt für Sicherheit in der Informationstechnik (BSI), "OPS.1.1.3: Patch and change management (original title in german: OPS.1.1.3: Patch- und Änderungsmanagement)", 2021, Accessed: 2026-03-14. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/IT-GS-Kompendium_Einzel_PDFs_2021/04_OPS_Betrieb/OPS_1_1_3_Patch_und_Aenderungsmanagement_Edition_2021.pdf?__blob=publicationFile&v=2.

[10] S. Alexiou, "Practical patch management and mitigation", *ISACA Journal*, vol. 2019, no. 3, pp. 1–6, 2019. Accessed: 2026-03-14. [Online]. Available: https://www.isaca.org/resources/isaca-journal/issues/2019/volume-3/practical-patch-management-and-mitigation.

[11] M. Souppaya and K. Scarfone, "Guide to enterprise patch management planning: Preventive maintenance for technology", National Institute of Standards and Technology (NIST), Tech. Rep. NIST SP 800-40 Rev. 4, 2022. Accessed: 2026-03-14. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-40r4.pdf4.

[12] J. R. C. Nurse, "To patch or not to patch: Motivations, challenges, and implications for cybersecurity", 2025, Accessed: 2026-03-14. [Online]. Available: https://arxiv.org/pdf/2502.17703.

[13] P. Przymus, M. Fejzer, J. Narębsk and K. Stencel, "The secret life of cves", *arXiv preprint*, 2025. Accessed: 2026-03-14. [Online]. Available: https://arxiv.org/pdf/2504.03863.

[14] P. Przymus, M. Fejzer, J. Narębsk, K. Rykaczewski and K. Stencel, "Out of sight, still at risk: The lifecycle of transitive vulnerabilities in maven", *arXiv preprint*, 2025. Accessed: 2026-03-14. [Online]. Available: https://arxiv.org/pdf/2504.04803.

[15] National Institute of Standards and Technology (NIST), "NVD data feeds - JSON 2.0 feeds", 28th May 2025, Accessed: 2026-03-14. [Online]. Available: https://nvd.nist.gov/vuln/data-feeds.

[16] MITRE Corporation, "CVE list downloads", 20th May 2025, Accessed: 2026-03-14. [Online]. Available: https://www.cve.org/Downloads.

[17] National Institute of Standards and Technology (NIST), "CVE-2014-0160 detail", 2014, Accessed: 2026-03-14. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2014-0160.

[18] National Institute of Standards and Technology (NIST), "CVE-2017-0144 detail", 2017, Accessed: 2026-03-14. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2017-0144.

[19] MITRE Corporation, "CVE numbering authority (cna) operational rules", Accessed: 2026-03-14. [Online]. Available: https://www.cve.org/ResourcesSupport/AllResources/CNARules.

[20] MITRE Corporation, "List of partners", https://www.cve.org/PartnerInformation/ListofPartners, Accessed: 2026-03-14.

[21] MITRE Corporation, "Glossary - cve record", https://www.cve.org/ResourcesSupport/Glossary#glossaryRecord, Accessed: 2026-03-14.

[22] MITRE Corporation, "Process", https://www.cve.org/About/Process, Accessed: 2026-03-14.

[23] National Institute of Standards and Technology (NIST), "CVEs and the NVD process", 2024, Accessed: 2026-03-14. [Online]. Available: https://nvd.nist.gov/general/cve-process.

[24] Bundesamt für Sicherheit in der Informationstechnik (BSI), "Bsi guideline on the coordinated vulnerability disclosure (cvd) process (original title in german: Leitlinie des BSI zum Coordinated Vulnerability Disclosure (CVD)-Prozess)", 2022, Accessed: 2026-03-14. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/CVD/CVD-Leitlinie.pdf?__blob=publicationFile&v=4.

[25] Bundesamt für Sicherheit in der Informationstechnik (BSI), "Study on the effectiveness of it security laws among operators of critical infrastructures (original title in german: Untersuchung zur Wirksamkeit der IT-Sicherheitsgesetze unter Betreibern Kritischer Infrastrukturen)", 2023, Accessed: 2026-03-14. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/KRITIS/evaluierung-itsig2-ergebnisbericht.pdf?__blob=publicationFile&v=3.

[26] World Economic Forum, "Global cybersecurity outlook 2022", 2025, Accessed: 2026-03-14. [Online]. Available: https://reports.weforum.org/docs/WEF_Global_Cybersecurity_Outlook_2025.pdf.

[27] MITRE Corporation, "CVE® 25 years - 25th anniversary report october 2024", https://www.cve.org/Resources/Media/Cve25YearsAnniversaryReport.pdf, 2024, Accessed: 2026-03-14.

[28] Forum of Incident Response and Security Teams (FIRST), "Exploit prediction scoring system (epss) - frequently asked questions", Accessed: 2026-03-14. [Online]. Available: https://www.first.org/epss/faq.

GNSS Spoofing Simulator

Tobias Reichel* , Mathias Gerstner,† Andreas Attenberger* , Klara Dološ*
* Central Office for Information Technology in the Security Sector
Munich, Germany
e-mail: {tobias.reichel, andreas.attenberger,klara.dolos}@zitis.bund.de
†Dept. Informatics and Mathematics, OTH Regensburg
Regensburg, Germany
e-mail: mathias.gerstner@oth-regensburg.de

***Abstract*—Modern systems rely heavily on satellite navigation systems for precise positioning, making resilience against spoofing attacks essential. Because satellite navigation signals are openly broadcast and unencrypted, robust anti-spoofing mechanisms must be developed and thoroughly tested. However, existing evaluation methods typically require costly hardware, limiting accessibility for research purposes. This work introduces a fully software-based satellite navigation spoofing simulator that enables realistic attack emulation and anti-spoofing validation without specialized equipment. It is capable of simulating two different satellite systems on one band at the same time, as well as generating two distinct signals, one for simulating the satellites and one for simulating the satellite navigation Spoofer. Validation results show that software-only spoofing provides an effective, low-cost method for advancing satellite navigation systems security research.**

Keywords-gnss; spoofing; simulator.

I. Introduction

Modern embedded systems rely heavily on navigation and location services, including Global Navigation Satellite System (GNSS). However, malicious attacks targeting GNSS data can be executed and remain largely undetectable at present, particularly in post-event analyses such as forensic investigations [1]. To mitigate data loss in such cases, several alternatives for increasing position data reliability have been proposed, including positioning via cellular towers [2] or Starlink [3], but each approach introduces significant drawbacks. Especially in remote environments, reliable and tamper-resistant GNSS functionality remains essential for accurate position determination.

Spoofing can be formally described in different ways. In this paper, the threat model is about data spoofing, where the aim is to replay or generate signals similar to those transmitted by GNSS satellites to miscalculate the position as given in [4]. Such attacks commonly employ a Software Defined Radio (SDR), a computer-controlled radio device capable of emitting synthesized signals. By transmitting a stronger signal than the authentic satellite broadcast, an attacker can force a receiver to lock onto the spoofed signal. However, this approach is inherently flawed: receivers can detect inconsistencies if the signal exhibits unrealistic physical properties, such as Doppler shift or differences in timing and amplitude. Furthermore, direction-finding antennas can determine the true source of a signal. These characteristics can be leveraged to design effective anti-spoofing mechanisms.

In Germany, and in most parts of the world, GNSS spoofing is prohibited, even for research purposes.[5] To avoid interference with critical infrastructure, researchers typically transmit signals over cables or use shielding tents to block high-frequency emissions. However, cable-based testing fails to reproduce the spatial characteristics of real-world spoofing, and shielding tents are often expensive and too small for realistic navigation experiments. The next section outlines existing testing approaches. Section III introduces a simulator that enables GNSS spoofing experiments without specialized hardware and discusses how hardware components could be integrated into the simulation. Section IV presents the validation of the proposed setup.

II. Related Work

The idea of GNSS signal simulation is not new, and numerous commercial tools are available. In, the first open-source project for Global Positioning System (GPS) signal generation was released [6]. Later, in, it was extended to support not only pre-generated and replayable data but also real-time operation [7]. Similar developments exist for other constellations, including Galileo [8], Beidou [9], and Global Navigation Satellite System (GLONASS) [10]. Based on our evaluations, only the open-source generators for GPS and Galileo currently provide sufficiently accurate signal quality. However, the proposed setup can be adapted to other generators once their signal fidelity improves.

For selecting a GNSS receiver that does not rely on dedicated hardware—i.e., one capable of processing recorded I/O files or streamed I/O data—two suitable options are SoftGNSS [11] and gnss_sdr [12]. Both tools can produce human-readable output in various formats and with different levels of detail.

There are multiple ways to study GNSS spoofing using these tools. Typically, they are deployed in hardware-based environments, as demonstrated in [13][14][15]. Even when gnss_sdr is used, it is usually combined with hardware components such as a GNSS receiver or an SDR. Spoofing is then simulated by adding an additional SDR and running multiple instances of the signal generators. In contrast, the setup described below requires no hardware at all; multiple generators can be synchronized and executed simultaneously in a fully software-based environment.

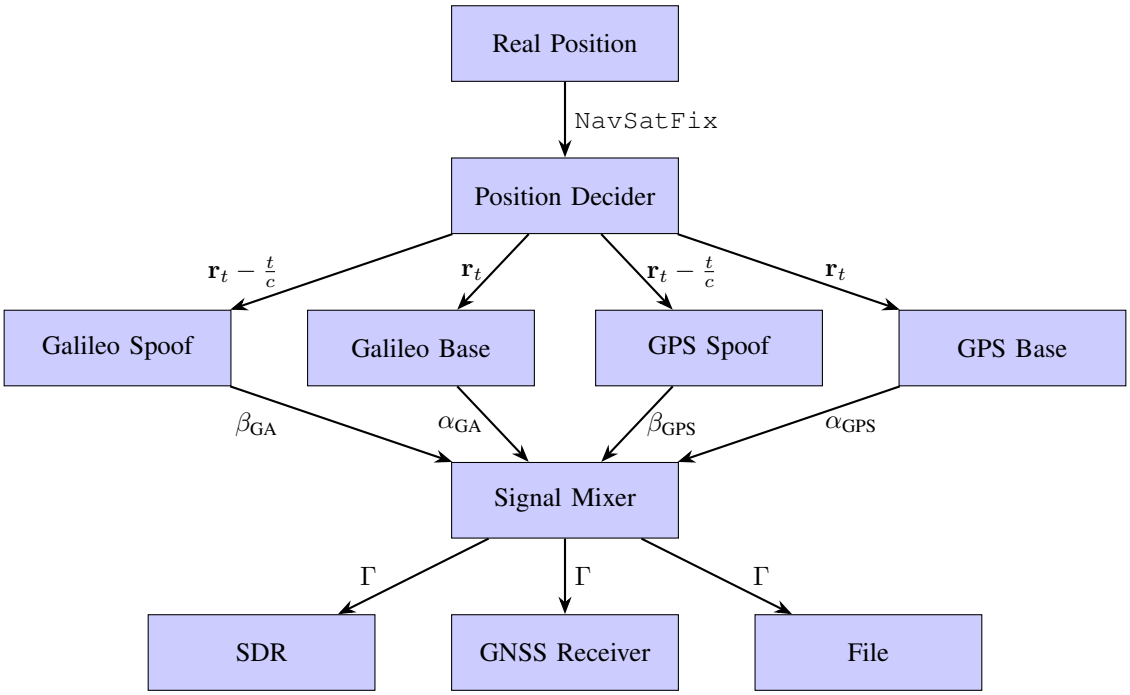


Figure 1. Systematic overview of the GNSS spoofing simulation setup.

III. Simulation Setup

To simulate a GNSS spoofing attack, a real acquisition of the receiver's position must be emulated first. For this purpose, a combination of GPS and Galileo signals generated by the tools in [7][8] can be produced using GNU Radio, a signal processing software. In the same manner, a spoofed version of the GNSS data can be synthesized. By combining the authentic acquisition with the spoofed data and forwarding it to a GNSS receiver, one can obtain a position fix or evaluate the receiver's anti-spoofing mechanisms.

The first requirement, illustrated in Figure 1, is a starting position representing the true receiver location, $\mathbf{r}_t = (x, y, z)_t$, which corresponds to the Cartesian coordinates (x, y, z) at time t. To integrate the simulation with real-world scenarios or external simulators, this position is transmitted dynamically via ROS2 using a `NavSatFix` message to the Position Decider, as shown in Figure 1. A brief description of the message fields is provided in Table I. The essential parameters are latitude, longitude, and altitude, as they are updated dynamically. The ROS2 message is then reformatted into a 24-byte vector, where each of the three parameters is encoded as an 8-byte value.

For spoofing, the transmitted position is not the true location but a slowly drifting value, $\mathbf{r}_t - \frac{t}{c}$, where c is a constant, such that $\frac{t}{c} \approx 10^{-4}$. Because the simulation does not include a physical receiver capable of orienting itself in the Galileo or GPS reference frames, ground-truth satellite signal data must be generated. These baseline signals are denoted as α_{GPS} and α_{GA}. In contrast, the spoofed GPS and Galileo signals, affected by the slow drift, are denoted as β_{GPS} and β_{GA}, representing

TABLE I. Description of a ROS NavSatFix message structure.

Field	Type	Description
header.stamp	time	Timestamp of the message
header.frame_id	string	Reference frame for the data
status.status	int8	Status of the satellite fix (e.g., STATUS_FIX, STATUS_NO_FIX)
status.service	uint16	Type of service (e.g., GPS, GLONASS)
latitude	float64	Latitude in degrees
longitude	float64	Longitude in degrees
altitude	float64	Altitude in meters above the WGS 84 ellipsoid
position_covariance	float64[9]	Row-major 3x3 covariance matrix for position
position_covariance_type	uint8	Type of covariance (e.g., unknown, approximated, diagonal known, known)

TABLE II. COMPARISON OF MESSAGE COUNTS, FIRST TIMESTAMP, MIN-MAX LATITUDE/LONGITUDE DIFFERENCES, AND FIX METADATA (0 –INVALID, 1 – GPS FIX, 2 – DGPS FIX 3 – 3D FIX, 4 – REALT-TIME KINEMATIC (RTK) FIX, 5 – RTK FLOAT, 6 – DEAD RECKONING, 7 – MANUAL INPUT.) ACROSS GNSS DATA SOURCES (GOOD AND SPOOFED SCENARIOS).)

Data Source	Num of mgs	Timestamp	Lat diff	Lon diff	Fix
Original ublox log	552	14:16:05	0.00002133	0.00002899	5
gnss_sdr α_{GPS}	223	00:13:13	0.00007283	0.00004543	6
gnss_sdr α_{GA}	605	00:02:43	0.00001224	0.00002130	5
gnss_sdr $\alpha_{\text{GPS}}, \alpha_{\text{GA}}$	2862	00:02:18	0.00012332	0.00011897	7
ublox α_{GPS}	226	No time fix	0.726741	0.586268	1
ublox α_{GA}	–	No time fix	–	–	0
ublox $\alpha_{\text{GPS}}, \alpha_{\text{GA}}$	231	No time fix	0.000392	0.000271	3
gnss_sdr $\alpha_{\text{GPS}}, \beta_{\text{GPS}}$	11552	00:00:24	0.000144	39.000095	4
gnss_sdr $\alpha_{\text{GA}}, \beta_{\text{GA}}$	4142	00:01:21	0.006285	20.047235	4
gnss_sdr $\alpha_{\text{GPS}}, \alpha_{\text{GA}}, \beta_{\text{GPS}}, \beta_{\text{GA}}$	142	00:01:13	0.777552	34.434319	4
ublox $\alpha_{\text{GPS}}, \beta_{\text{GPS}}$	178	00:00:35	0.000222	0.001033	0
ublox $\alpha_{\text{GA}}, \beta_{\text{GA}}$	–	No time fix	–	–	0
ublox $\alpha_{\text{GPS}}, \alpha_{\text{GA}}, \beta_{\text{GPS}}, \beta_A$	186	No time fix	0.000224	0.000220	0
ublox $\alpha_{\text{GPS}}, \alpha_{\text{GA}}$ and $\beta_{\text{GPS}}, \beta_A$	185	No time fix	32.666489	45.748938	0

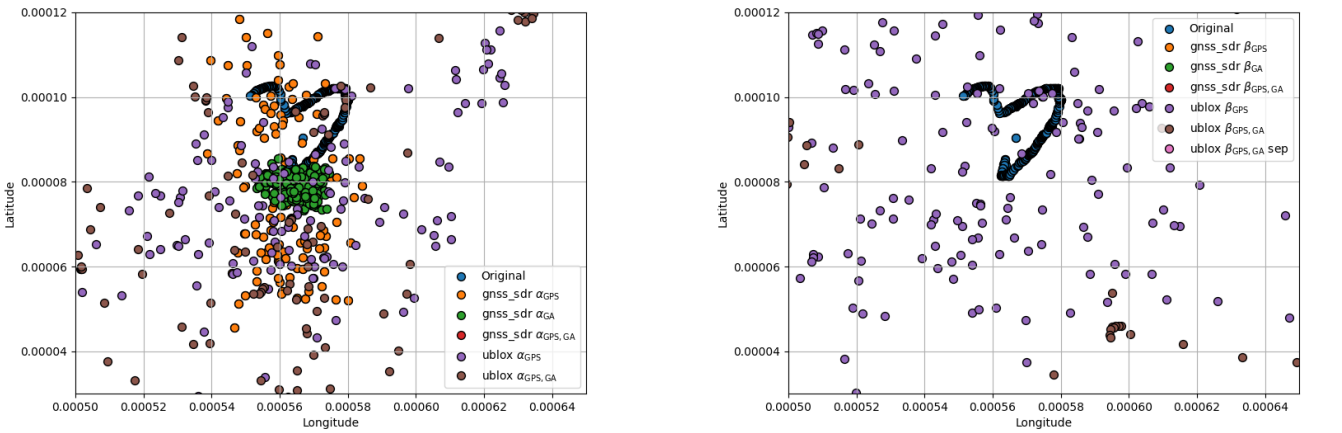
the signals that a typical GNSS spoofer would emit.

The second requirement is valid pre-recorded reference data for the signal generators, which determines both the radius of the data-gathering region and the simulation time. This radius is defined by the set of satellites visible from both the true and spoofed positions, which should be similar. Assigning different gains to the generators $\alpha_{\text{GPS}}, \alpha_{\text{GA}}, \beta_{\text{GPS}}$, and β_{GA} produces different outcomes. Since receivers generally lock onto signals with higher gain—and satellite signals are inherently weak—the spoofed generators β_{GPS} and β_{GA} must be assigned higher gain than the baseline generators. In the validation setup, a gain difference of 30 dB is applied. By combining the signal data $\alpha_{\text{GPS}}, \alpha_{\text{GA}}, \beta_{\text{GPS}}, \beta_{\text{GA}}$ from all four sources as complex number in the following way

$$\Gamma = (\alpha_{\text{GPS}} + \alpha_{\text{GA}}) + (\beta_{\text{GPS}} + \beta_{\text{GA}}),$$

the resulting signal Γ can be written to a file, transmitted via an SDR, or fed directly into a GNSS receiver. Sending Γ to gnss_sdr, a software-based GNSS receiver, is most easily achieved using ZeroMQ (ZMQ). GNU Radio provides multiple sinks capable of transmitting data to common SDRs, sending it over ZMQ, or writing it to a file, all of which are handled in the Signal Mixer as seen in Figure 1.

Ultimately, the receiver determines whether it accepts the spoofed signal, the authentic signal, or fails to acquire a fix. To support this evaluation, various algorithms can be configured in gnss_sdr under the acquisition and tracking modules. For acquisition, we selected Parallel Code Phase Search (PCPS), which employs Fast Fourier Transform techniques to accelerate processing. Advanced anti-spoofing mechanisms, such as those described in [16], may also be integrated. In the tracking module, the configuration specifies how carrier-to-



(a) Location points of the different simulations and the test recording.

(b) Location points of the different spoofed simulations and the test recording.

Figure 2. Comparison of GNSS location points under normal and spoofed conditions.

TABLE III. COMPARISON OF GENERATED MESSAGE TYPES ACROSS THE SIMULATIONS.

Data Source	RMC	GGA	GSA	GSV	Observation File	C/N_0
Original ublox Log	yes	yes	yes	yes	yes	yes
gnss_sdr α_{GPS}	yes	yes	yes	no	no	no
gnss_sdr α_{GA}	yes	yes	yes	no	no	no
gnss_sdr $\alpha_{\text{GPS}}, \alpha_{\text{GA}}$	yes	yes	yes	yes	yes	yes
ublox α_{GPS}	yes	yes	yes	yes	yes	yes
ublox α_{GA}	yes	no	no	no	yes	no
ublox $\alpha_{\text{GPS}}, \alpha_{\text{GA}}$	yes	no	no	no	yes	yes
gnss_sdr $\alpha_{\text{GPS}}, \beta_{\text{GPS}}$	yes	yes	no	yes	no	no
gnss_sdr $\alpha_{\text{GA}}, \beta_{\text{GA}}$	yes	yes	yes	no	no	no
gnss_sdr $\alpha_{\text{GPS}}, \alpha_{\text{GA}}, \beta_{\text{GPS}}, \beta_{\text{GA}}$	yes	yes	yes	yes	yes	yes
ublox $\alpha_{\text{GPS}}, \beta_{\text{GPS}}$	yes	yes	yes	yes	yes	yes
ublox $\alpha_{\text{GA}}, \beta_{\text{GA}}$	yes	no	no	no	yes	no
ublox $\alpha_{\text{GPS}}, \alpha_{\text{GA}}, \beta_{\text{GPS}}, \beta_A$	yes	no	no	no	yes	yes
ublox $\alpha_{\text{GPS}}, \alpha_{\text{GA}}$ and $\beta_{\text{GPS}}, \beta_A$	yes	no	no	yes	yes	yes

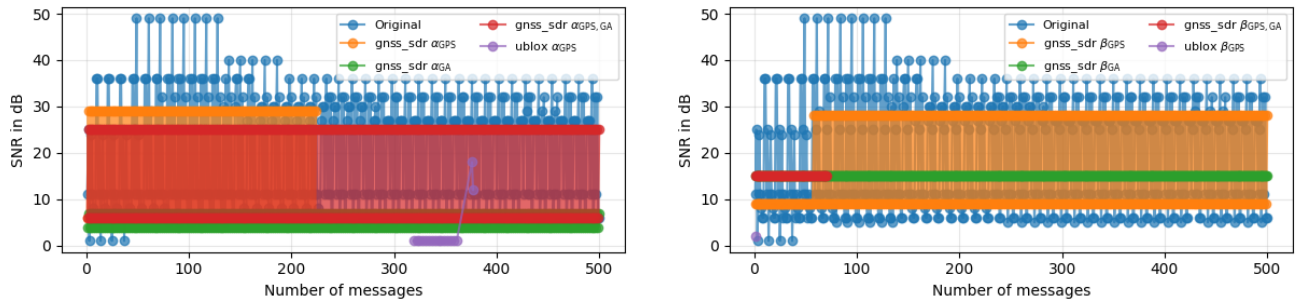
noise density, code and carrier lock, discriminators, and low-pass filters are applied. We selected a Delay Lock Loop (DLL) architecture and a Phase-Locked Loop (PLL) architecture, both of which are already implemented in the framework.

IV. VALIDATION

For the validation, a test recording from the 332nd day of the year 2025 was collected using a u-blox ZED-X20P together with a Survey GNSS Tripleband + L-band antenna from Ardusimple. The GNSS receiver's position was approximately 49°N, 12°E and was logged with six decimal places of accuracy. The receiver recorded raw data (UBX format), which were converted into an observation file, an National Marine Electronics Association (NMEA) file, and a GeoJSON file to enable analysis of the different data types. The observation file contains information about all satellites in view, whereas the NMEA and GeoJSON files are only generated when the receiver obtains a valid fix. The ephemeris files required for the signal generators were obtained from the open-access databases Crustal Dynamics Data Information System (CDDIS)[17] for GPS and International GNSS Service (IGS)[18] for Galileo.

To validate the data generation, a comparison was performed between the test recording, the files generated by our simulation setup, and measurements from the same GNSS receiver receiving signals from two Ettus B200 devices and our simulation inside a shielding tent. Table II lists all measurements and basic statistics. Notably, while gnss_sdr produces significantly more messages, it remains accurate and obtains an appropriate fix. Its timestamps, however, begin at the start time of the ephemeris file. The hardware GNSS receiver was more difficult to deceive. Despite attempts to disable built-in spoofing protection, these mechanisms likely remained partially active, resulting in compromised data. The following analysis focuses first on the position data, then on the signal strength extracted from the NMEA messages, and finally on the Carrier-to-Noise Ratio (C/N_0) values from the observation files.

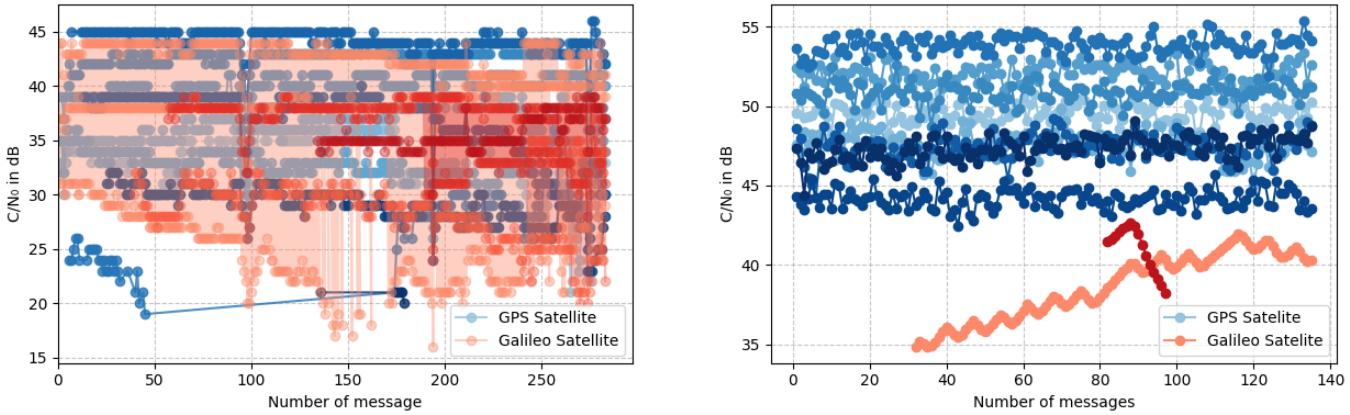
Figure 2a shows the location data extracted from the GeoJSON files for both the base simulations and the test recording. The test recording exhibits much lower variance than the simulated data, although the overall error remains small, as confirmed by the latitudinal and longitudinal differences in Table III. In Figure 2b, which includes all spoofed simulations and the original data, a clear shift to the east and south is visible. This demonstrates that the spoofing attack had an effect, although



(a) SNR values of the different simulations and the test recording.

(b) SNR values of the spoofed simulations and the test recording.

Figure 3. SNR values of simulation with gnss_sdr and the test recording.



(a) C/N_0 of the test recording.

(b) C/N_0 of GPS and Galileo Signal Input.

Figure 4. Comparison of test recording and simulation with gnss_sd.r

the simulated variance is slightly larger than in the real-world data. This is a relevant aspect for anti-spoofing algorithm development.

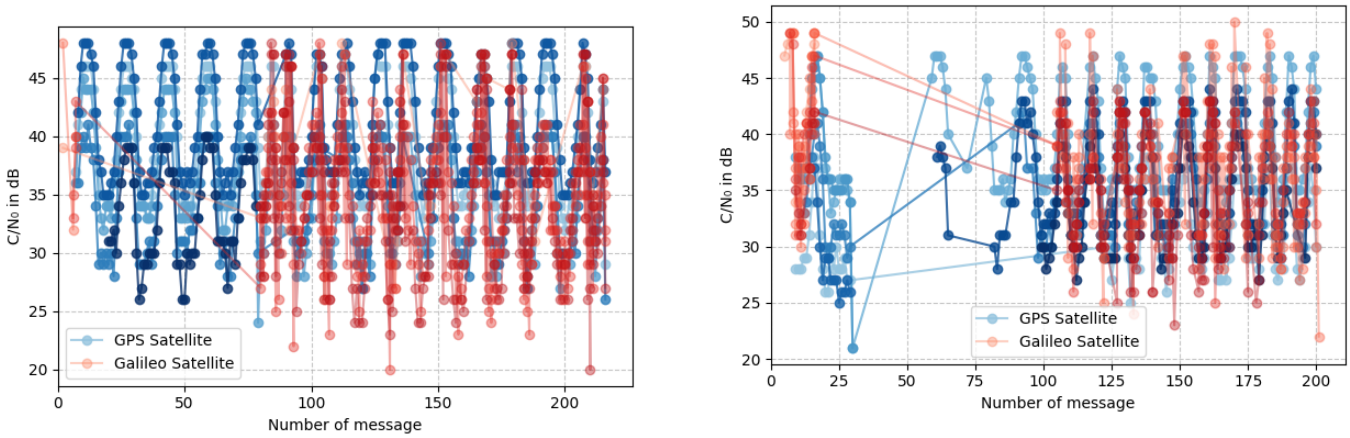
In the next step, only a subset of the NMEA data was analysed, since gnss_sdr produces only Recommended Minimum Navigation Data (RMC), Global Positioning System Fix Data (GGA), GNSS DOP and Active Satellites (GSA), and GNSS Satellites in View (GSV) messages. Table III lists which simulations produced which message types. The first timestamps appear in the RMC and GGA messages. Table II shows that the hardware GNSS receiver did not accept the spoofed time, whereas gnss_sdr did. The Signal to Noice Ratio (SNR) values, contained in the GSV messages, were then examined. Figure 3a overlays all SNR values per message and shows that the mean and variance match well across setups. For the spoofed simulations, Figure 3b shows similar behaviour, although the Galileo-only and combined GPS and Galileo spoofing simulations exhibit degraded data quality.

Finally, the observation files were analysed. Not all simulations produced these files, and even when available, the relevant C/N_0 values may be missing. Interestingly, timestamps missing in Table II could be extracted from the observation files, although they were never used by the receiver. Simulations that produced C/N_0 data are listed in Table III. Each figure in this section shows one simulation, with all satellites and their corresponding C/N_0 values.

The first notable result is that the simulation can generate data similar to the test recording. The test recording is stable overall, with outliers for satellites barely visible (Figure 4a). gnss_sdr also produces stable data but occasionally exhibits artefacts where a satellite ID is permanently switched (Figure 4b).

A second observation is that the combined GPS and Galileo signal input produces more reliable and abundant data, as reflected in Table III. This behaviour was expected but is nevertheless important to confirm.

The final analysis concerns simulations where spoofed signals were mixed in. Three cases were examined: one using gnss_sdr



(a) C/N_0 of spoofed and base GPS and Galileo Signal Input.

(b) C/N_0 of spoofed and base GPS and Galileo Signal Input, where spoof and base were transmitted with separate SDR.

Figure 5. Comparison between spoofed signal simulation in a ublox receiver, where the signals are mixed via GNU radio or transmitted via different SDRs.

and two using the hardware GNSS receiver with spoofed GPS and Galileo signals. For the SDRs, one simulation transmitted all signals through a single device, while the other separated base and spoofed signals across two different SDRs.

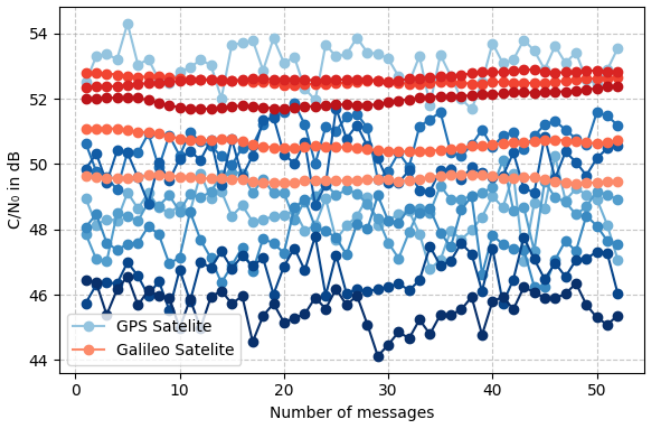


Figure 6. C/N_0 of GPS and Galileo spoofed simulation with gnss_sdr.

The spoofed signal (Figure 6) is significantly more irregular and exhibits stronger fluctuations than the baseline signal (Figure 4b). Comparing it to the real receiver (Figure 5a), the software simulation produced fewer satellites, but similar data points. The number of satellites acknowledged by the receiver drops when the base and spoofed signals are transmitted through separate SDRs (Figure 5b), but the C/N_0 exhibits significantly larger fluctuations likely due to the receiver switching between two independent sources. gnss_sdr handles this situation more gracefully (Figure 6), although the artefacts remain visible. It should be mentioned that the overall values of the C/N_0 in the spoofed simulation with gnss_sdr are much higher than those of the receiver. This is probably caused by interference in the air and the distance between the SDR and the receiver.

V. Discussion

A further issue is that some data could not be produced. In the case of the hardware GNSS receiver, this was likely due to built-in anti-spoofing mechanisms, even though raw values could still be extracted from the observation file. Conversely, gnss_sdr only generates files when it has a valid fixed position. This becomes problematic when analysing spoofing behaviour, because no data are available during the initial phase of the attack. Future work should address this limitation. Additionally, several data types were not analysed but could provide valuable insights. For example, other NMEA message types or navigation files, rather than observation files, may contain useful information.

Some limitations are also given by the signal generators [7] and [8], which we chose to use, caused by the fact that they only generate data for the L1 or E1 bands. Therefore, interesting cross validations between the bands cannot be done. Finally, the spoofed data were less stable and exhibited more fluctuations than the non-spoofed data. This is likely due to the use of identical ephemeris files, which produce identical satellite identifiers and signal characteristics. It should be investigated how the receiver behaves when slightly different ephemeris files are used.

VI. Conclusion and Future Work

This work demonstrates a method to simulate GNSS spoofing attacks without relying on expensive hardware, providing a useful tool for developing and evaluating mitigation strategies. Three major findings stand out. First, combining multiple GNSS signals significantly strengthens the spoofing effect and can deceive receivers more easily. Second, analysing only the coordinates is insufficient to determine whether a spoofing attack has occurred. Third, executing a real-world spoofing attack is considerably more challenging due to the widespread implementation of anti-spoofing mechanisms.

During the validation of the hardware-free GNSS spoofing simulator, several aspects were identified that warrant further investigation, as discussed in the previous section. Future work should focus on supporting additional satellite systems, improving data acquisition, and enhancing data generation by using varied ephemeris files.

Another promising direction would be to investigate how multi-band receivers behave when only a single band is simulated in detail. This would help determine whether a one-band model is sufficient for meaningful system-level evaluation or whether cross-band interactions require a more comprehensive multi-band simulation approach. It would also be valuable to conduct controlled tests on a certified GNSS test range or laboratory environment where such experiments are legally permitted. Finally, future work could include a brief comparison between the hardware-free simulator and a commercial GNSS spoofing system to better understand the differences in signal characteristics, system behaviour, and practical limitations under safe and authorized test conditions.

Acknowledgments

This work was supported by the project 'Digital Forensics in IT Systems (DiForIT)', funded by the German Federal Ministry for Economic Affairs and Climate Action (BMWK).

References

[1] T. Reichel et al., "A forensic analysis of GNSS spoofing attacks on autonomous vehicles", in *Proceedings of the Sixteenth International Conference on Cloud Computing, GRIDs, and Virtualization*, 2025, pp. 32–39.

[2] K. Muthineni, A. Artemenko, J. Vidal, and M. Nájar, "A survey of 5G-based positioning for industry 4.0: State of the art and enhanced techniques", in *2023 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, 2023, pp. 120–125. DOI: 10.1109/EuCNC/6GSummit58263.2023.10188352.

[3] Y. Song, *Positioning and navigation methods based on starlink signals: A comprehensive review*, 2024, https://www.researchgate.net/publication/388122630, Accessed: Feb. 23, 2026.

[4] M. Bartock et al., "Foundational PNT profile: Applying the cybersecurity framework for the responsible use of positioning, navigation, and timing (PNT) services", National Institute of Standards and Technology, Tech. Rep. NIST IR 8323 Rev. 1, Jan. 2023, Accessed: Feb. 23, 2026.

[5] Federal Republic of Germany, *Telecommunications act (tkg) § 149 – administrative offenses*, 2021, https://www.gesetze-im-internet.de/tkg_2021/__149.html, Accessed: Feb. 23, 2026.

[6] T. Ebinuma, *Gps-sdr-sim*, 2015, https://github.com/osqzss/gps-sdr-sim, Accessed: Feb. 23, 2026.

[7] gym487, *Gps-sdr-sim-realtime*, 2017, https://github.com/gym487/gps-sdr-sim-realtime, Accessed: Feb. 23, 2026.

[8] H. Sathaye, M. Motallebighomi, and A. Ranganathan, "Galileo-SDR-SIM: An open-source tool for generating galileo satellite signals", in *Proceedings of the 36th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2023)*, 2023, pp. 3470–3480.

[9] yangfan852219770, *Beidou-sdr-sim*, 2021, https://github.com/yangfan852219770/beidou-sdr-sim, Accessed: Feb. 23, 2026.

[10] A. A. Maksutov, D. A. Valter, G. V. Borisenko, and K. A. Ovchinnikov, "Real-time simulation of the GLONASS system signals using SDR", in *2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, 2019, pp. 26–28. DOI: 10.1109/EIConRus.2019.8657287.

[11] P. Berglez et al., "Development of a dual frequency software-based GNSS receiver", in *Proceedings of the ION GNSS 2010*, 2010, pp. 1967–1974.

[12] C. Fernández-Prades, J. I. Arribas, and P. Closas, *GNSS-SDR: An open source tool for researchers and developers*, pp. 780–789, 2011, https://gnss-sdr.org, Accessed: Feb. 23, 2026.

[13] W. Feng, J.-M. Friedt, G. Goavec-Merou, and F. Meyer, "Software-defined radio implemented GPS spoofing and its computationally efficient detection and suppression", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 36, no. 3, pp. 36–52, 2021. DOI: 10.1109/MAES.2020.3040491.

[14] M. Ali, F. Zahra, and A. A. Raja, "Robust spoofing detection in gnss-sdr systems: A two-stage method for real-time signal integrity", in *2024 3rd International Conference on Emerging Trends in Electrical, Control, and Telecommunication Engineering (ETECTE)*, 2024, pp. 1–6. DOI: 10.1109/ETECTE63967.2024.10823726.

[15] N. Stenberg, E. Axell, J. Rantakokko, and G. Hendeby, "Results on GNSS spoofing mitigation using multiple receivers", *NAVIGATION: Journal of the Institute of Navigation*, vol. 69, no. 1, Mar. 2022. DOI: 10.33012/navi.510.

[16] C. Fernandez-Prades, *GNSS-SDR documentation: Acquisition blocks*, 2021, https://gnss-sdr.org/docs/sp-blocks/acquisition/, Accessed: Feb. 23, 2026.

[17] C. E. Noll, "The crustal dynamics data information system: A resource to support scientific analysis using space geodesy", *Advances in Space Research*, vol. 45, no. 12, pp. 1421–1440, 2010, ISSN: 0273-1177. DOI: 10.1016/j.asr.2010.01.018.

[18] International GNSS Service (IGS), *BRDC navigation files, day 332, 2025*, https://igs.bkg.bund.de/root_ftp/IGS/BRDC/2025/332/, Accessed: Feb. 23, 2026.

 ISBN: 978-1-68558-442-9

Introducing the Cyber-Physical Data Flow Diagram to Improve Threat Modelling of Internet of Things Devices

Simon Liebl [1,2], Ian Ferguson [2], Andreas Aßmuth [3], Natalie Coull [2], George R. S. Weir [4]
[1] emgarde, Ebermannsdorf, Germany
e-mail: simon.liebl@emgarde.de
[2] Abertay University, Dundee, UK
e-mail: {i.ferguson | n.coull}@abertay.ac.uk
[3] Kiel University of Applied Sciences, Kiel, Germany
e-mail: andreas.assmuth@haw-kiel.de
[4] University of Strathclyde, Glasgow, UK
e-mail: george.weir@strath.ac.uk

***Abstract*—A growing number of Internet of Things (IoT) devices are used across consumer, medical, and industrial domains. They interact with their environment through sensors and actuators and connect to networks such as the Internet. Because sensors may collect sensitive data and actuators can trigger physical actions, security, privacy, and safety are major challenges. Threat modelling can help identify risks, but established IT-focused methods transfer to the IoT only to a limited extent. In this paper, a new modelling technique specifically for IoT devices called Cyber-Physical Data Flow Diagram (CPDFD) is proposed that also allows modelling of hardware with the aim to support manufacturers in identifying threats and developing countermeasures. The technique was examined through an experimental study and a survey with interviews. The results suggest that numerous other attack scenarios can be found through the modelling technique, improving the identification of threats to IoT devices.**

Keywords-Security; Internet of Things; Embedded Systems; Threat Modelling; Cyber-Physical Data Flow Diagram.

I. Introduction

The Internet of Things (IoT) is the interconnection of billions of devices via the Internet or another network. These devices are used in a wide variety of areas, such as the smart home, medical, and industrial applications. Their key characteristics include interacting with the environment via sensors and actuators and communicating with other devices and systems. Networking them aims to extend functionality, integrate services, improve usability, increase efficiency, and reduce costs. However, adding multiple network interfaces also expands the attack surface and increases the risk of cyber attacks. It seems that the number of published incidents is steadily increasing [1], indicating that IoT security is a serious problem. In recent years, incidents have been reported in various IoT application fields, such as industrial (e.g., attacks on Ukraine's power grid [2]), medical (e.g., insulin pumps and ventilators [3]), automotive (e.g., hack of Tesla and its Wall Connector [4][5]), infrastructure (e.g., traffic lights [6]), and smart home (e.g., vacuum robot [7]).

The described small sample of security incidents related to IoT devices and systems shows that there is a systematic problem threatening security, privacy, and safety. Therefore, new approaches are required to better protect IoT devices from such threats. A promising approach that is already established in the IT domain is the risk assessment using threat modelling. Threat modelling is the systematic approach to identifying threats and vulnerabilities to a particular system. Appropriate countermeasures can then be defined. The process can also be carried out early in the product lifecycle, which supports the goal security by design. However, common threat modelling approaches from the IT domain cannot necessarily be adopted to IoT devices. Reasons for this are that IoT devices consist of resource-constrained embedded systems and that their interaction with the real world poses an increased risk to the privacy and safety of users. As part of this work, the modelling technique Data Flow Diagram (DFD), which is often used in threat modelling, was specifically examined. Several adaptations are proposed to increase the applicability of DFDs to IoT devices, including two new elements and the possibility to create a Hardware Diagram (HWD). This extension, called Cyber-Physical Data Flow Diagram (CPDFD), has the aim to enable more detailed modelling of the components of IoT devices and thus allow better threat identification compared to regular DFDs. This technique is intended to support device manufacturers in identifying threats and developing appropriate countermeasures.

The remainder of this paper is structured as follows: in Section II, the fundamentals of IoT devices and threat modelling are provided. Section III presents related work regarding IoT security, threat modelling, and DFDs. In Section IV, the proposed extension CPDFD is introduced. The methodology for evaluating CPDFDs consisting of two studies is described in Section V, followed by the presentation of the results. These results are then discussed in Section VII. The paper ends with conclusions in Section VIII.

II. Background

A. Internet of Things Devices

The IoT is a "group of infrastructures interconnecting connecting objects and allowing their management, data mining and the access to the data they generate" [8]. These objects are devices extended with network connectivity and computing capabilities and require usually minimal human intervention [9]. The architecture of the IoT is often structured into different

 ISBN: 978-1-68558-442-9

layers [10]. The perception layer, the lowest layer, consists of these physical objects, such as sensors, actuators, or Radio-Frequency Identification (RFID) tags. These device are also used in a Cyber-Physical System (CPS), for example, to control physical processes in the real world [11]. For such systems, real-time requirements are essential, which differentiates them clearly from IT systems. The impact of IoT devices can thus affect security, privacy, and safety and have consequences such as the disclosure of sensitive data in medical applications or human injury in industrial applications.

IoT devices are embedded systems, which are specific computer systems built for a custom purpose. Their basic components consist of hardware, software, and data [12]. Examples for hardware components are microcontroller, memory chip, Printed Circuit Board (PCB), security chip, power supply, and various sensors and actuators. Software components include firmware, basic libraries for cryptography and logging, and protocol stacks for numerous IoT protocols, such as Bluetooth, ZigBee, and MQTT. Besides firmware, devices store access data, keys, and configuration and log files, among others.

B. Threat Modelling

In this section, the DFD and two published threat modelling techniques are introduced. The small excerpt of techniques is necessary to understand the remainder of this paper.

1) Data Flow Diagram: The DFD is a graphical modelling technique used historically in the field of software engineering and system analysis [13]. It quickly gained popularity due to its intuitive nature and ability to capture both high-level and detailed views of system processes and data. A DFD consists only of four elements: *Process*, *Data Store*, *External Entity*, and *Data Flow*. This enables a simple visual representation of complex systems and illustrates how data is input, processed, stored, and output. Processes represent the transformation of data and can be thought of any running code. Data stores depict any type of data storage, such as files or databases. External entities are used to describe external data sources or destinations, e.g., people or any code outside your control. Data flows are used to connect the other three elements. Besides their use in software engineering, security experts found their application also valuable in the field of cyber security. The analysis of the system is an important part of a security assessment conducted to find threats and vulnerabilities. In order to limit the assessment for complex systems, another element called *Trust Boundary* or *Trust Area* was added to visualise different trust levels, initiating the second version of DFDs [14]. Since DFDs are typically created during early design, they naturally support the analysis of architectural security issues rather than implementation-level defects.

2) STRIDE and LINDDUN: The CIA triad – Confidentiality, Integrity, and Availability – forms the basis of information security, often complemented by goals such as authenticity, non-repudiation, and authorisation. STRIDE covers the threats spoofing, tampering, repudiation, information disclosure, denial of service, and elevation of privilege [14], and was developed at Microsoft to identify such threats during design [15]. It maps directly to these six security goals.

STRIDE can be applied to DFDs, but checking all six threats for every element is time-consuming. Since some elements are more susceptible to specific threats, STRIDE-per-Element and STRIDE-per-Interaction were introduced to streamline the analysis [14].

LINDDUN is a framework for privacy threat modelling. Similar to STRIDE, LINDDUN is an acronym that stands for linkability, identifiability, non-repudiation, detectability, disclosure of information, unawareness, and non-compliance and can be combined with DFDs as well [16]. The threats are therefore not focused on security goals, but are instead aimed at privacy goals.

III. Related Work

IoT security and privacy is still a huge problem, which is why a lot of research is being conducted in a variety of directions. Due to the many articles around IoT security, there are several reviews with the aim to summarise, among others, threats, attacks, challenges, and solutions [17]–[21]. Some researchers focused specifically on IoT devices, often using hands-on approaches to show how embedded systems can be attacked, which threats arise, and how they can be mitigated [22]–[26]. Regarding threat modelling and risk assessment, there are a couple of articles with the aim of adapting these methods for the IoT. In [27], they aim to automate this process for IoT systems by providing their own methodology. The authors of [28] present a STRIDE and DFD-based threat modelling approach specifically for CPSs with special focus on human injury, equipment damage as well as black-out. Many articles address specific fields of application such as automotive [29], building and home automation [30], agriculture [31], and healthcare devices [32]. DFDs are also relevant to this work. In this regard, [33] and [34] are particularly worth mentioning, as they propose adjustments to DFDs with the aim of improving threat identification and generation.

In summary, there is a lot of research in the area of IoT security and privacy, also with special emphasis on threat modelling. In the case of IoT devices in particular, the differences between embedded systems and IT systems were recognised and threats analysed that emerge through sensors, actuators, and interaction with the environment and other systems. However, the hands-on approaches and the various methods for the different IoT applications emphasise that a common technique for modelling IoT devices has not yet been established.

IV. Cyber-Physical Data Flow Diagram

This section introduces the proposed modelling technique CPDFD by describing its aims and adaptations.

A. Aim

As mentioned before, there are several issues with modelling IoT devices. One big difference between IT and IoT applications is the interaction with the physical world. While this is an

integral part of the latter, it is hardly present in IT applications. In IoT systems, a sensor, the data source, usually measures an environmental value, such as temperature, photographic picture, or heart activity. The processing of the data can in turn lead to physical operations by the actuator, the data sink, and thus result in changes in the environment. For example, a valve is opened, a car battery is charged, or a door is unlocked. From a security point of view, these sensor values and actuator actions are particularly interesting, as they can threaten the privacy and safety of users. It is therefore necessary to address this difference, the interaction with the physical environment, also in the modelling.

Another difference between IT and IoT applications is the location and purpose. IoT devices are used in critical applications, such as healthcare, automotive, and automation. Many of them are placed outdoors, e.g., security cameras, charging stations, or pipeline valves. The criticality as well as the accessibility of the devices make hardware attacks interesting for attackers and are frequently reported [35]. Successful attacks give deep insight into the device and often lead to access to Intellectual Property (IP), sensitive data, credentials, and cryptographic secrets. Therefore, hardware attacks, which are often out of scope in IT applications, need to be considered due to the accessibility and the frequent critical application.

One core element of IoT devices is the communication with other devices and systems. Besides dozens of wireless communication protocols, such as Z-Wave, Wi-Fi, and NFC, wired protocols including Modbus, Ethernet, and USB are commonly used. Additionally, several common protocols used in embedded systems are used, for instance, UART, RS-232, and JTAG. With the multitude of protocols and interfaces in use, it is therefore easy to lose track which of them are actually utilised in an IoT device. However, these various protocols and interfaces increase the attack surface and can become a gateway for attackers. The goal is therefore to be able to represent these interfaces in a model in order to get a quick overview of the connectivity of a device.

There is another difference between the development of IoT devices and IT applications. Modelling the latter is mainly done by software and network engineers that commonly have a computer science background. However, modelling an IoT device may also require the help of hardware and embedded engineers with a background frequently in mechanical or electrical engineering. Therefore, a modelling technique for IoT devices needs to be simple and at the same time comprehensive in order to be applicable by people with different backgrounds.

B. The Technique

As mentioned before, the CPDFD is proposed to support modelling of IoT devices. The technique extends the DFD, which is commonly used for security assessments and is known by many security experts and engineers. The following paragraphs describe adjustments to the DFD in order to achieve the previously described aims.

It is difficult to represent the interaction with the physical environment through sensors and actuators using the five elements of a DFD. Depending on the sensor characteristics (e.g., simple electrical resistors or modules with a serial interface), it can make sense to represent them as *Process*, *Data Store*, or *External Entity* or even as a combination with *Data Flows*. However, this question of detail would take a lot of time and contradict the principle of simplicity of DFDs. If one does not know how to represent a specific component right away, it could be omitted. This in turn would lead to a situation where measured values which might threaten the privacy, for example, would not be recorded. This applies analogously to actuators and safety as a protection goal. It is therefore proposed to add a new element called *Physical Link* to the DFD notation in order to be able to model these interactions with the physical environment. The goal of this element is to give sensors and actuators a first-class representation in the diagrams, highlighting issues with the goals of privacy and safety.

Another aim of the CPDFD technique is to enable the creation of hardware models in order to support the identification of hardware attacks, among others. In theory, there are dozens of modelling techniques (see [36]). However, the authors are not aware of any hardware modelling technique that is commonly used for the threat analysis. Hardware components were integrated into DFDs in a few cases (e.g., [37][38]). It is proposed to use the DFD notation to create a separate HWD. This is achieved by splitting the elements into logical and physical ones. Regular DFDs are mainly used for modelling software-related components, which are now referred to as logical elements. The additional physical elements are used for the creation of the HWD. The physical data store can be used to model a flash or One Time Programmable (OTP) memory, for instance. Likewise, the physical processor and the physical trust area can be used to model microprocessors and PCBs, for example. Modelling of data flows is not necessary in the HWD.

The CPDFD technique should also allow users to get a quick overview of the device's communication interfaces. Similar to the *Physical Link* before, modelling the interfaces through one of the standard elements could have the effect that these are not modelled. It is therefore proposed to model them as a separate element named *Interface* as part of the HWD.

All elements of a CPDFD and their use in a DFD and HWD are shown in Table I. The proposed extension has further advantages. A DFD can be augmented with links to the hardware model. For example, for data stores, such as a file, it can be specified in which memory it is stored and for data flows, the utilised communication interface can be provided. This additional context can lead to a more precise description of an attack scenarios and thus support the threat identification. The new elements *Physical Link* and *Interface* have another benefit when it comes to software-aided modelling. Threat rules for automatic threat generation can be specified more precisely or can be created specifically for the two elements. This can reduce false positives, i.e., wrongly identified threats,

TABLE I. THE ELEMENTS OF A CPDFD. THE LAST TWO COLUMNS INDICATE THEIR USE IN DFDS AND HWDS.

Element	Symbol	Meaning & Examples	DFD	HWD
Process	Name	Executed code Log. Process: Code in C, web server Phy. Processor: Microprocessor, TPM	☒ ☐	☐ ☒
Data Store	Name	Things that store data Log. Data Store: File, database Phy. Data Store: FLASH, OTP memory	☒ ☐	☐ ☒
External Entity Interactor	Name	People or code outside your control Log. External Entity: Browser, cloud service Phy. External Entity: User, machine	☒ ☒	☐ ☐
Data Flow	Name Name	Communication between elements (Log.) Data Flow: Data flow with protocol (stack)	☒	☐
Trust Area	Name	Delimited area of trust Log. Trust Area: Net segment, container Phy. Trust Area: PCB, casing	☒ ☒	☐ ☒
Physical Link	Name	Link between physical and logical world Physical Link: Microphone, battery, motor	☒	☒
Interface	Name	Communication interface Interface: Wi-Fi, USB, JTAG	☐	☒

and false negatives, i.e., undetected threats. The commonly used techniques STRIDE-per-Element and LINDDUN can still be used for CPDFDs and also provide more accurate results. Last but not least, the CPDFD technique establishes a unifying notation for HWDs and DFDs. The resulting diagrams remain simple and clear, despite the addition of two elements and the distinction between physical and logical elements. They can be created and understood by hardware, embedded, and software engineers and other people as well.

V. METHODOLOGY

The methodology for evaluating the improvements of the CP-DFD is introduced in this section. It consists of an experimental study and a survey with interviews.

A. Experimental Study

This section introduces an experimental study with the objective to compare CPDFDs with DFDs by quantitatively comparing the number of attack scenarios identified by two groups, each using one modelling technique. Participants of the study, who were students of computer science-related study programmes, were put in the following situation: As innovative students, they have applied their previously acquired knowledge in practice and developed a smart IoT device alongside their studies. They now want to bring the device to market, but they still have concerns about security and privacy. Therefore, each participant had the task of examining the device for threats and vulnerabilities. The analysed device was the open source device Jaimico [39], which is a combination of voice assistant and health monitor. The companion robot can be worn on the shoulders in daily life and can be used as a voice assistant or health monitor to measure body temperature and heart activity by connecting to a wearable. In addition to analysing health data in the cloud, the device offers the special feature of detecting Covid19 through analysing recorded coughing and body temperature. In this study, participants were supported by a custom software tool called TTModeler [40][41], but the supported modelling technique, specifically the two new elements, differed for both groups. The modelling technique thus serves as an independent variable. The produced project file and a concluding questionnaire served as the data source for further analysis. For the sake of completeness, it should be noted that there was a third group with a different software tool, but this is not of interest for this paper.

The aim of this study is to gather evidence on the usability of each tool, which is the "degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use" [42]. For this purpose, effectiveness is defined as "accuracy and completeness with which users achieve specified goals" and efficiency as the resources expended in relation to the effectiveness [42]. More precisely, it was measured how much time is spent on the creation of a specific model and how many attack scenarios were found in this context, enabling the quantification in identified attack scenarios per minute for the hardware model, among others. This can be used to determine the efficiency. The impact of the additional elements *Physical Link* and *Interface* was specifically examined.

It was decided to conduct the study with students because they are available at short notice and in larger numbers than, for example, developers from industry. Students from study programmes related to computer science were invited to participate in the study. All participating students were in the fourth year of their studies. They represent the actual users well, as they have, for example, only rudimentary experience in the field of security, as it is the case with many firmware and software developers. A total of 46 students volunteered to participate in the study. Five of them took part in a preliminary study, which was conducted to find issues in the procedure and task description, and their results are not considered.

This results in a sample size of 41. Due to the omitted third group and unexpected cancellations, 12 participants used the modelling technique CPDFD, denoted as G_{CPDFD}, and 14 participants used the DFD technique (G_{DFD}).

Participants first read an information document covering the CIA, STRIDE, and LINDDUN threat models, as well as IoT privacy threats defined by [43]. They then received a second document introducing DFDs, including STRIDE-per-Element, LINDDUN-per-Element, and example diagrams. Group G_{CPDFD} received a modified version explaining CPDFD elements instead of standard DFDs. Next, participants read the description of the example device Jaimico and watched a video demonstrating the main features of the assigned tool, including how to model the "Register device" use case and add threats. They then worked on the task, with the option to stop at any time within a three-hour limit. Finally, they completed the questionnaire and submitted their project file.

The submitted project files and questionnaires served as the data sources. Project files were analysed for diagrams, elements, and attack scenarios. Duplicates and out-of-scope scenarios were marked and excluded. Remaining scenarios were filtered and labelled as considered attack scenarios. Because the device description was partly fictitious, scenarios were classified as having either low or high probability of correctness. Most were rated high, with exceptions such as "Malicious file on USB flash drive", since the USB port was specified as charging-only. Each scenario was assigned a type: *generic*, *outlined*, or *custom*, indicating the level of detail and difficulty of identification. *Generic* scenarios were directly derived from STRIDE or LINDDUN, *outlined* scenarios added some detail (e.g., "Clear text data storage of the database"), and *custom* scenarios were highly specific or not obviously derived from a threat model (e.g., "Unsecure Bluetooth version"). Only high-probability scenarios of type *outlined* or *custom* were considered relevant.

The tool was modified to track modelling time. After completing the task, participants answered a 21-item questionnaire, including eight items on threat identification and 13 on usability.

The further analysis of the collected data was carried out with the help of descriptive and inferential statistical methods [44] and may be visualised using a box plot [45]. The significance level for rejecting a null hypothesis is set at $\alpha = 5\%$. Depending on whether the results are normally distributed, either the *t*-test or the Mann-Whitney *U* test is used to test for a difference between two groups. Subsequently, Cohen's d is calculated to quantify the effect size [46][47]. It can be assumed that there is a small effect for $0.2 < d < 0.5$, a medium effect for $0.5 \leqslant d < 0.8$, and a large effect for $d \geqslant 0.8$.

B. Survey and Interviews

The last study aims to get feedback from people who might actually utilise CPDFDs. These include device manufacturers as well as consultancies. Manufacturers often do not have security experts themselves and therefore hire external companies. Participants apply the methodology (Thing Threat Modelling (TTM)), technique (CPDFD), and tool (TTModeler) on their own devices and provide afterwards feedback through a questionnaire and an interview. The aim is to specifically ask for opinions on the HWD and the new elements.

TABLE II. OVERVIEW OF PARTICIPANT AND COMPANY CHARACTERISTICS.

Category	Distribution
Company type	8 device manufacturers; 7 service providers
Participant role	9 security experts; 6 engineers
Company size	7 large; 3 medium-sized; 4 small; 1 micro
Field of application	8 industrial; 3 consumer; 2 automotive; 1 medical; 1 infrastructure
Experience	46.7% embedded security; 80.0% DFD; 60.0% threat modelling

All persons with a connection to cyber security, IoT, or embedded systems were invited for the study. In total, 15 participants could be recruited for the study. Table II categorizes participants according to the type of company, the size of the company (according to [48]), and the experience. Noteworthy is that two participants had no cyber security experience (13.3%).

At the start of the study, participants received an introduction to threat modelling, the adaptations used in this work, and their task. After completing and evaluating the contributions, they filled out a questionnaire and answered a few interview questions. The questionnaire, consisting mainly of five-point Likert items [49] and free-text fields, together with the interview, served as the data sources. Questionnaire responses were analysed descriptively, while the free-text fields and interview data were examined qualitatively.

VI. RESULTS

The following section summarises the results of both studies.

A. Experimental Study

This section presents the results of the experimental study, summarised in Table III. The first tests on the results using the Shapiro-Wilk test showed that the sample is not normally distributed. This could be caused by the rather small sample size, for example. It was therefore decided to consistently use non-parametric methods.

Table III shows the duration needed to conduct the study for both groups. Note that this duration represents the pure interaction time with the tool and does not include the learning time required for the threat models, CPDFD/DFD, and the introduction to Jaimico. Group G_{CPDFD} has a wide range of 80 to 142 minutes ($M = 111$ min, $SD = 20$ min). In contrast, G_{DFD} has a higher mean value of 120 minutes ($SD = 10$ min) and a narrower range of 98 to 132 minutes. Despite these differences, there is a large overlap of the ranges and it indicates that the technique used does not have a significant influence on the execution duration (Mann-Whitney *U* test, $U = 62.5$, $p = .280$, $p > \alpha$).

The number of considered attack scenarios, i.e., without duplicates, high probability of correctness and out of scope (see subsection V-A), have a large range. Participants of G_{CPDFD} found in total between 2 and 182 attack scenarios ($M = 69.21$, $SD = 40.44$) and G_{DFD} found between 2 and 74 scenarios ($M = 45.17$, $SD = 19.85$). The scenarios start at 2, as some

TABLE III. OVERVIEW OF THE PROJECT FILE ANALYSIS.

	G_{CPDFD}	G_{DFD}
Duration	111 min ± 20 min	120 min ± 10 min
Considered attack scenarios	69.21 ± 40.44	45.17 ± 19.85
Type: Generic	19.29 ± 22.08	17.92 ± 14.64
	27.86% ± 31.90%	39.67% ± 32.41%
Type: Outlined	10.86 ± 5.84	9.58 ± 7.95
	15.69% ± 8.44%	21.22% ± 17.60%
Type: Custom	39.07 ± 17.17	17.67 ± 10.00
	56.45% ± 24.80%	39.11% ± 22.15%
Relevant attack scenarios	49.93 ± 21.00	27.25 ± 11.32
	72.14% ± 30.34%	60.33% ± 25.07%
Goal: Security	44.93 ± 19.48	27.00 ± 11.14
	89.99% ± 39.01%	99.08% ± 40.89%
Goal: Privacy	5.00 ± 2.11	0.25 ± 0.45
	10.01% ± 4.23%	0.92% ± 1.66%
Physical Link-related	10.86 ± 5.17	
	21.75% ± 10.36%	
Interface-related	11.21 ± 5.10	
	22.46% ± 10.22%	
Model: Hardware	29.07 ± 11.28	9.50 ± 6.67
	58.23% ± 22.58%	34.86% ± 24.47%
Model: Software	4.29 ± 8.09	4.67 ± 5.77
	8.58% ± 16.20%	17.13% ± 21.19%
Model: Data flow	16.57 ± 8.71	13.08 ± 9.40
	33.19% ± 17.44%	48.01% ± 34.51%
Time for hardware model	26 min ± 14 min	12 min ± 12 min
Time for software model	9 min ± 7 min	8 min ± 7 min
Time for data flow model	64 min ± 21 min	96 min ± 17 min
Time per attack scenario	2 min ± 0 min	4 min ± 0 min
- for hardware model	1 min ± 0 min	1 min ± 1 min
- for software model	2 min ± 2 min	2 min ± 2 min
- for data flow model	4 min ± 1 min	7 min ± 1 min

Note: The values of the last two columns show the mean value and the standard deviation. Furthermore, the table is split in three sections. The percentages in the second section refer to the considered attack scenarios, those in the third section to the relevant attack scenarios.

participants had mainly modelled the example use case showed in the introductory video. Their type, *generic*, *outlined*, and *custom*, can be further analysed. The results for the types *generic* and *outlined* are similar and with overlapping error bars (see Table III). A difference can be seen for the type *custom*. The groups G_{CPDFD} and G_{DFD} found on average 39.07 (SD = 17.17) and 17.67 (SD = 10.00) scenarios respectively. The Mann-Whitney U test shows that both groups are significantly different from each other. Therefore, G_{CPDFD} found significantly more attack scenarios of the high ranked type *custom* than G_{DFD} (U = 155.0, p = .000, $p \leqslant \alpha$, d_{Cohen} = 1.50, large effect). This means that G_{CPDFD} found more scenarios with high information value that are more difficult to find.

The attack scenarios were filtered again to include only those of the type *outlined* and *custom* as explained in subsection V-A. This reduced list of relevant attack scenarios is visualised in Figure 1. The main range of G_{CPDFD} is between 33 and 74 scenarios with two outliers at 2 and 94, while G_{DFD} ranges between 2 and 39. The ranges overlap, but the boxes highlighting the lower and upper quartiles do not. The median of G_{CPDFD} (49) differs from G_{DFD} (30) by 19 scenarios. The result of the Mann-Whitney U test shows that group G_{CPDFD} identified significantly more attack scenarios than G_{DFD} (U = 147.0, p = .001, $p \leqslant \alpha$, d_{Cohen} = 1.32, large effect). This clearly shows that group G_{CPDFD}, which had the *Physical Link*

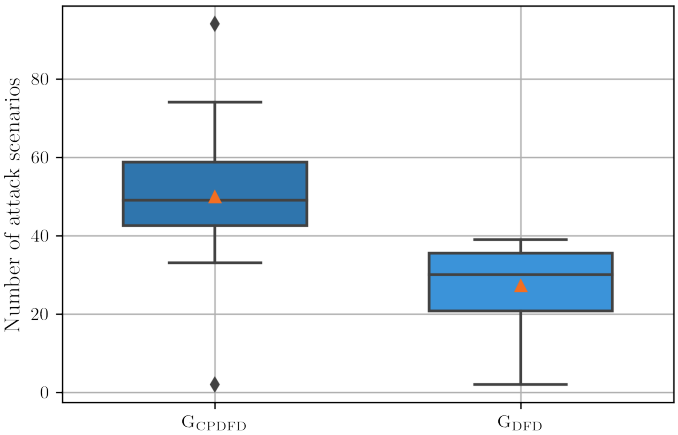


Figure 1. Relevant attack scenarios for groups G_{CPDFD} and G_{DFD} as box plot.

and *Interface* available, identified substantially more attack scenarios.

Attack scenarios were categorised whether these threaten the security or privacy of the device under consideration. Group G_{CPDFD} identified 5.00 scenarios on average (SD = 2.11), while G_{DFD} identified less than 1 scenario on average (M = 0.25, SD = 0.45). Therefore, they identified significantly more privacy attack scenarios than G_{DFD} (Mann-Whitney U test, U = 160.5, p = .000, $p \leqslant \alpha$, d_{Cohen} = 3.01, large effect). A more detailed analysis shows that out of these 5 scenarios of G_{CPDFD}, 3.64 scenarios were identified through the element *Physical Link* (SD = 1.69), 1.21 through the *Interface* (SD = 0.58), and only 0.14 scenarios through other elements (SD = 0.36). Consequently, this suggests that the used technique, CPDFD or DFD, has an influence on the identified attack scenarios against privacy.

Figure 2 shows the average number of identified attack scenarios per group. Furthermore, it highlights how many scenarios of G_{CPDFD} relate to the additional elements of a CPDFD – *Physical Link* and *Interface*. The comparison of G_{CPDFD} and G_{DFD} shows that both groups found on average almost the same number of scenarios for category *Others*. The extra scenarios of G_{CPDFD} relate to the new elements. Therefore, this suggests that further attack scenarios can be found through the two elements.

Table III includes in which model, hardware, software, or data flow, the attack scenarios were identified. The main difference can be seen for the hardware model. Group G_{CPDFD} has a range of 14 to 45 identified scenarios, without a single outlier at 2, and a median of 29.5. G_{DFD} has a range of 0 to 19 and a median of 11.5. The Mann-Whitney U test shows that the groups are significantly different (U = 156.0, p = .000, $p \leqslant \alpha$, d_{Cohen} = 2.08, large effect). Again, the difference between G_{CPDFD} and G_{DFD} arises from the two new elements. The median *Physical Link* and *Interface*-related scenarios for the former are 6 and 10.5 scenarios, respectively. The median scenarios for the standard elements are 13 and similar to G_{DFD}. The results indicate that it is worth creating a hardware model and that there is an improvement due to the new elements.

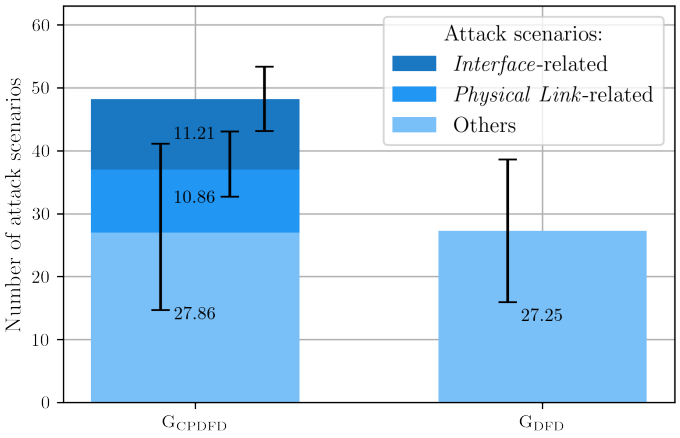


Figure 2. Average attack scenarios per group. In group G_{CPDFD}, scenarios relating to the elements *Physical Link* and *Interface* are highlighted.

Furthermore, the tool TTModeler tracked how much time participants spent on creating and analysing the different models, enabling the calculation of the duration per attack scenario (see Table III). Group G_{CPDFD} needed on average 2 minutes per scenario ($SD = 0$ min), while G_{DFD} needed 4 minutes ($SD = 0$ min). The average time per attack scenario in the hardware model was even less for both groups, because both needed 1 minute per scenario on average. These scores indicate the improved efficiency of CPDFDs and also of a hardware model in general.

It was recorded how participants modelled certain components. The technique CPDFD would classify microphone, loudspeaker, touch screen, and temperature sensor as element *Physical Link*, for example. While almost all participants in group G_{CPDFD} modelled the microphone as *Physical Link* (93%), the other group was divided. One quarter modelled the microphone as *Process*, 17% as *External Entity*, and the majority of 58% did not model it at all. For the loudspeaker, the picture is almost the same: only one participant had modelled it as *Data Store* instead of *Process*. While the majority of G_{CPDFD} also modelled the touch screen, the temperature sensor, and the PCB, the share for G_{DFD} was one third or less. The battery was not modelled by anyone of G_{DFD}, but by 86% of G_{CPDFD}. Likewise, the same was analysed for the interfaces JTAG, Bluetooth, Wi-Fi, and USB. Almost all participants of G_{CPDFD} modelled these components as element *Interface* (86%, 93%, 86%, and 79%, respectively). In contrast, only 8% of G_{DFD} modelled JTAG, Bluetooth, and Wi-Fi and 25% modelled the USB interface. Since the majority of G_{CPDFD} modelled all these components, while G_{DFD} did not, this suggests an added value of the two elements *Physical Link* and *Interface*.

Last, a brief review of the questionnaire. The only item of interest in this study was about the simplicity of creating a CPDFD/DFD. A total of 61.54% of G_{CPDFD} agreed that creating a DFD is easy, while 41.67% agreed of G_{DFD}. This seems to indicate that the two new elements of a CPDFD have not increased the difficulty of a DFD, but actually made it easier. However, this difference is not significant (Mann-Whitney U test, $U = 13.5$, $p = .911$, $p > \alpha$). Therefore, the difficulty of creating a CPDFD and DFD can be considered as equal.

B. Survey and Interviews

This section presents the results of the survey and interviews with device manufacturers and consulting companies. The precise duration for conducting the study was not measured. Participants indicated that they invested from two hours up to two days. Most of them applied the TTM methodology on their own devices. A few also carried out the analysis for other devices, e.g., power inverter or smart home. Some people used the results from previous assessments for a side-by-side comparison.

a) Questionnaire: All participants agreed that it is worth to create and analyse a hardware model (13 strongly agreed, 2 agreed) and no one disagreed that reusing the elements of a DFD is the right technique for this (4 strongly agreed, 5 agreed, 3 indecisive). It was also asked about the added value of the two new elements, resulting in exactly the same outcome. The majority of 58.3% strongly agreed that both elements *Physical Link* and *Interface* provided added value (17% agreed, 8% each indecisive, disagreed, and strongly disagreed). They also agreed (72.7%) that the elements contribute to a more accurate modelling of IoT devices. Additional feedback was provided noting that the two elements facilitate modelling, but that the interface needs to be more integrated. It was also remarked that the meaning of both elements is similar. The results suggest that the hardware model, the *Physical Link*, and the *Interface* improve the threat identification.

b) Interviews: This section summarises the findings of the interview analysis. Statements by individual persons are anonymised. If necessary, the person is described by categories introduced in subsection V-B. In these cases, the role of the persons and their company type are indicated in parentheses, together with their participant number, e.g., (P_1, security expert, manufacturer).

When participants were asked about the relevance of a hardware model, all agreed that this is an important part. Several participants stated that the hardware model was the best part of the TTM methodology. Hardware attacks are not considered sufficiently and "without them one cannot find the crux of the matter" (P_8, security expert, manufacturer). One participant had tried to represent hardware security in a model in the past using the system modelling language. However, he said that a structured threat analysis was not possible. In general, it was seen as positive that one can quickly see how the device is constructed and which interfaces and components, such as Trusted Platform Modules (TPMs), it uses.

The extension of the DFD notation with the two elements *Physical Link* and *Interface* was seen positive, but some participants were not fully convinced. One participant found the extension very good, as she "struggled in the past to map certain hardware components in a well-distinguishable yet abstract way" (P_{13}, security expert, service provider). The *Physical Link* was found to be good, as data is generated there and

actuators can influence the environment. “Even if a sensor is analogue, that does not mean you don‘t have to think about it” (P_{10}, security expert, service provider). More feedback was given for the *Interface*. It is useful and important, as “debug interfaces are often overlooked” (P_9, security expert, service provider). In a certain way, one is also forced to think about interfaces. However, a few participants noted that they did not fully understand the difference between both elements. The elements could also make the models more abstract and thus more vague in certain cases. One participant noted that the new elements are useful; their use is left to the user.

The feedback from the participants clearly shows that the hardware model brings improvements and is important for the threat analysis. Overall, the two new elements were also seen positively and their enrichment appreciated.

VII. Discussion

A. Hardware Model

In the experimental study, the median of group G_{CPDFD} identified 29.5 attack scenarios in the hardware model (59.1% of all), G_{DFD} found at least 11.5 scenarios (42.2%). The efficiency was also higher for the hardware model (1 minute per scenario) than for the average (2 and 4 minutes per scenario for G_{CPDFD} and G_{DFD} respectively). In the survey, all participants, security experts and engineers, agreed that it is worth to create and analyse a hardware model (see subsection VI-B). Furthermore, they reaffirmed the importance of the hardware model in the interviews. One of the advantages they mentioned was the quick overview of all components and interfaces. In conclusion, the quantitative and qualitative results speak in favour of a hardware model. This suggests that using a hardware model may improve threat identification.

B. Physical Link and Interface

The two new elements *Physical Link* and *Interface* were evaluated separately, but the jointly collected results differ only slightly. Their results are therefore summarised together first and then briefly discussed individually.

In the experimental study, the two groups G_{CPDFD} and G_{DFD} had the single difference that the former had the two new elements available in the diagram editor. However, G_{CPDFD} found significantly more scenarios than G_{DFD} (49.9 and 27.3 on average respectively) (see subsection VI-A). The assignment of the scenarios to elements showed that G_{CPDFD} also discovered 27.9 scenarios for the standard elements. The additional scenarios were initiated by the *Physical Link* (10.9) and the *Interface* (11.2). There was also a difference regarding privacy-related threats. Group G_{CPDFD} found on average 5.0 scenarios while the G_{DFD} found less than 1 scenario. The difference is not statistically significant, but can be explained by the fact that G_{CPDFD} found 3.6 scenarios with the *Physical Link* and 1.2 scenarios with the *Interface*. This effect could also be seen in the hardware model. The median found 29.5 and 11.5 scenarios for G_{CPDFD} and G_{DFD}, respectively. More than half of the scenarios of the former were based on the *Physical Link* (6) and *Interface* (10.5). It was also analysed how participants modelled specific components. For example, almost all of group G_{CPDFD} modelled the microphone as *Physical Link*. In contrast, only 42% of G_{DFD} modelled it. They were indecisive which element to use, as 3 participants modelled it as *Process* and 2 as *External Entity*. The interfaces such as JTAG and Bluetooth were modelled by about 90% of G_{CPDFD} while these were modelled by only one third or less of G_{DFD}. This means that the existence of the two new elements leads to such components being modelled at all. The two elements do not make the technique more difficult, as the questionnaire showed. In the survey, the majority of 75% agreed that each *Physical Link* and *Interface* provide added value and 73% agreed that they contribute to more accurate modelling of IoT devices (see subsection VI-B). In the interviews, the elements were generally seen positive. One security expert confirmed the problems with modelling certain hardware components without the new elements. The *Physical Link* was said to be good as data is generated there and actuators can influence the environment and the *Interface* was conceived to be important as debug interfaces are often overlooked, among others. However, it was noted a few times that the differentiation between both elements is not clear and that they could be summarised in one element. In summary, there are many arguments in favour of introducing the two new elements. The *Physical Link* leads to several additional scenarios, as seen in the experimental study, although not as many as for the *Interface*. The element can be used to represent sensors, which in turn have mainly contributed to the identification of threats against privacy. This added value was also endorsed by the survey participants. The results thus suggest that the *Physical Link* improves the threat identification. The situation is similar for the *Interface*. Quantitatively, even more scenarios could be identified with this element. The importance of interfaces was highlighted by students as well as security experts and engineers. Therefore, the results suggest the *Interface* improves the threat identification as well.

C. Limitations

Some limitations must be taken into account in the reported results. The HWD produced dozens of attack scenarios, but several of them may also be found in DFDs, creating overlaps. Consequently, the number of additional scenarios contributed by the HWD is lower than the raw count suggests. The quantitative effects of the two new elements must also be considered with caution. In the experimental study, participants of G_{CPDFD} had predefined stencils for the *Physical Link*, e.g., microphone and loudspeaker, and for the *Interface*, e.g., JTAG and Bluetooth. The list of dozens of stencils had to be searched through first, but these increase the likelihood that such components will be modelled. The modelling then automatically led to further scenarios. For a more detailed examination of the effects of the two elements, further groups would have been necessary, which would not have had predefined stencils available.

VIII. Conclusions and Future Work

The many security incidents in the IoT highlight the ongoing need for improved security. Although numerous researchers

address IoT security, systematic approaches are often lacking. This work introduces a modelling technique tailored to IoT devices to simplify modelling and enhance threat identification, helping manufacturers address weaknesses early in the product development cycle and support security by design. The proposed CPDFD technique considers the special circumstances of IoT devices. Due to their use in critical applications as well as the high accessibility, the creation of a hardware model was proposed. In addition, due to the interaction with the environment and the high connectivity of IoT devices, the two new elements *Physical Link* and *Interface* were introduced. Across two studies, an experimental quantitative comparison was conducted and the perspectives of security experts and engineers were collected. In summary, all three changes of the CPDFD technique indicate improvements compared to the standard DFD. The technique enables more detailed modelling of IoT devices, which in turn leads to more identified attack scenarios, as the results demonstrated. This suggests that CPDFDs improve the identification of threats. In the future, the better integration of HWDs and DFDs will be addressed to exploit synergy effects and improve the identification of attack scenarios and countermeasures.

REFERENCES

[1] SonicWall, *Annual number of Internet of Things (IoT) malware attacks worldwide from 2018 to 2022*, Statista, 2023. Accessed: 2026-03-14. [Online]. Available: https://www.statista.com/statistics/1377569/worldwide-annual-internet-of-things-attacks/.

[2] D. E. Whitehead, K. Owens, D. Gammel, and J. Smith, 'Ukraine cyber-induced power outage: Analysis and practical mitigation strategies', in *2017 70th Annual Conference for Protective Relay Engineers (CPRE)*, 2017, pp. 1–8. DOI: 10.1109/CPRE.2017.8090056.

[3] D. Truxius et al., 'Cyber Security Review of Network-Connected Medical Devices', German Federal Office for Information Security (BSI), Tech. Rep., Dec. 2020. Accessed: 2026-03-14. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/DigitaleGesellschaft/ManiMed_Abschlussbericht_EN.pdf?__blob=publicationFile&v=1.

[4] D. Comobo, *How I got access to 25+ Tesla's around the world. By accident. And curiosity.* Jan. 2022. Accessed: 2026-03-14. [Online]. Available: https://medium.com/@david_colombo/how-i-got-access-to-25-teslas-around-the-world-by-accident-and-curiosity-8b9ef040a028.

[5] C. R. Lab, *2025: Unpacking the Tesla Wall Connector exploit chain and its broader cybersecurity implication*, Aug. 2025. Accessed: 2026-03-14. [Online]. Available: https://vicone.com/blog/from-pwn2own-automotive-2025-unpacking-the-tesla-wall-connector-exploit-chain-and-its-broader-cybersecurity-implication.

[6] B. Ghena, W. Beyer, A. Hillaker, J. Pevarnek, and J. A. Halderman, 'Green Lights Forever: Analyzing the Security of Traffic Infrastructure', in *8th USENIX Workshop on Offensive Technologies (WOOT 14)*, San Diego, CA: USENIX Association, Aug. 2014. Accessed: 2026-03-14. [Online]. Available: https://www.usenix.org/conference/woot14/workshop-program/presentation/ghena.

[7] D. Giese, 'Reverse engineering and hacking Ecovacs robots', DEFCON 32, 11th Aug. 2024. Accessed: 2026-03-14. [Online]. Available: https://dontvacuum.me/talks/DEFCON32/DEFCON32_reveng_hacking_ecovacs_robots.pdf.

[8] B. Dorsemaine, J.-P. Gaulier, J.-P. Wary, N. Kheir, and P. Urien, 'Internet of Things: A Definition & Taxonomy', in *2015 9th International Conference on Next Generation Mobile Applications, Services and Technologies*, Cambridge, United Kingdom: IEEE, Sep. 2015, pp. 72–77, ISBN: 978-1-4799-8660-6. DOI: 10.1109/NGMAST.2015.71. Accessed: 2026-03-14. [Online]. Available: http://ieeexplore.ieee.org/document/7373221/.

[9] K. Rose, S. Eldridge, and L. Chapin, 'The internet of things: An overview', *The internet society (ISOC)*, vol. 80, pp. 1–50, 2015, Publisher: Reston, VA.

[10] M. A. Iqbal, S. Hussain, X. Huanlai, and M. A. Imran, *Enabling the internet of things: fundamentals, design, and applications* (Wiley - IEE), First edition. Hoboken, NJ: Wiley, 2020, ISBN: 978-1-119-70125-5.

[11] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, 'The industrial internet of things (IIoT): An analysis framework', *Computers in Industry*, vol. 101, pp. 1–12, Oct. 2018, ISSN: 01663615. DOI: 10.1016/j.compind.2018.04.015. Accessed: 2026-03-14. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0166361517307285.

[12] S. Liebl et al., 'Analyzing the Attack Surface and Threats of Industrial Internet of Things Devices', *International Journal on Advances in Security*, 1 & 2, vol. 14, pp. 59–70, Dec. 2021, ISSN: 1942-2636. Accessed: 2026-03-14. [Online]. Available: https://www.thinkmind.org/index.php?view=article&articleid=sec_v14_n12_2021_6.

[13] T. DeMarco, 'Structure Analysis and System Specification', in *Pioneers and Their Contributions to Software Engineering: sd&m Conference on Software Pioneers, Bonn, June 28/29, 2001, Original Historic Contributions*, M. Broy and E. Denert, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 255–288, ISBN: 978-3-642-48354-7. DOI: 10.1007/978-3-642-48354-7_9. Accessed: 2026-03-14. [Online]. Available: https://doi.org/10.1007/978-3-642-48354-7_9.

[14] A. Shostack, *Threat modeling: designing for security*. Indianapolis, IN: Wiley, 2014, ISBN: 978-1-118-80999-0.

[15] L. Kohnfelder and P. Garg, 'The threats to our products', Microsoft, Tech. Rep., Apr. 1999.

[16] M. Deng, K. Wuyts, R. Scandariato, B. Preneel, and W. Joosen, 'A privacy threat analysis framework: Supporting the elicitation and fulfillment of privacy requirements', *Requirements Engineering*, vol. 16, no. 1, pp. 3–32, Mar. 2011, ISSN: 0947-3602, 1432-010X. DOI: 10.1007/s00766-010-0115-7. Accessed: 2026-03-14. [Online]. Available: http://link.springer.com/10.1007/s00766-010-0115-7.

[17] Y. Harbi, Z. Aliouat, S. Harous, A. Bentaleb, and A. Refoufi, 'A Review of Security in Internet of Things', *Wireless Personal Communications*, vol. 108, no. 1, pp. 325–344, Sep. 2019, ISSN: 0929-6212, 1572-834X. DOI: 10.1007/s11277-019-06405-y. Accessed: 2026-03-14. [Online]. Available: http://link.springer.com/10.1007/s11277-019-06405-y.

[18] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, 'Internet of Things security: A survey', *Journal of Network and Computer Applications*, vol. 88, pp. 10–28, Jun. 2017, ISSN: 10848045. DOI: 10.1016/j.jnca.2017.04.002. Accessed: 2026-03-14. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1084804517301455.

[19] E. Leloglu, 'A Review of Security Concerns in Internet of Things', *Journal of Computer and Communications*, vol. 05, no. 01, pp. 121–136, 2017, ISSN: 2327-5219, 2327-5227. DOI: 10.4236/jcc.2017.51010. Accessed: 2026-03-14. [Online]. Available: http://www.scirp.org/journal/doi.aspx?DOI=10.4236/jcc.2017.51010.

[20] O. I. Abiodun, E. O. Abiodun, M. Alawida, R. S. Alkhawaldeh, and H. Arshad, 'A Review on the Security of the Internet of Things: Challenges and Solutions', *Wireless Personal Communications*, vol. 119, no. 3, pp. 2603–2637, Aug. 2021, ISSN: 0929-6212, 1572-834X. DOI: 10.1007/s11277-021-08348-9. Accessed: 2026-03-14. [Online]. Available: https://link.springer.com/10.1007/s11277-021-08348-9.

[21] M. Scott, 'A Survey Study of Common Security Failures and Mitigations for the Internet of Things (IoT)', *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 15, no. 2, pp. 9–15, Jun. 2023. Accessed: 2026-03-14. [Online]. Available: https://jtec.utem.edu.my/jtec/article/view/6259.

[22] O. Arias, J. Wurm, K. Hoang, and Y. Jin, 'Privacy and Security in Internet of Things and Wearable Devices', *IEEE Transactions on Multi-Scale Computing Systems*, vol. 1, no. 2, pp. 99–109, 2015. DOI: 10.1109/TMSCS.2015.2498605.

[23] C. Kolias, A. Stavrou, J. Voas, I. Bojanova, and R. Kuhn, 'Learning Internet-of-Things Security "Hands-On"', *IEEE Security & Privacy*, vol. 14, no. 1, pp. 37–46, 2016. DOI: 10.1109/MSP.2016.4.

[24] J. Wurm, K. Hoang, O. Arias, A.-R. Sadeghi, and Y. Jin, 'Security analysis on consumer and industrial IoT devices', in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2016, pp. 519–524. DOI: 10.1109/ASPDAC.2016.7428064.

[25] A. W. Atamli and A. Martin, 'Threat-Based Security Analysis for the Internet of Things', in *2014 International Workshop on Secure Internet of Things*, 2014, pp. 35–43. DOI: 10.1109/SIoT.2014.10.

[26] S. Iskhakov, A. Shelupanov, and A. Mitsel, 'Internet of Things: Security of Embedded Devices', in *2018 3rd Russian-Pacific Conference on Computer Technology and Applications (RPC)*, 2018, pp. 1–4. DOI: 10.1109/RPC.2018.8482148.

[27] V. Casola, A. De Benedictis, M. Rak, and U. Villano, 'Toward the automation of threat modeling and risk assessment in IoT systems', *Internet of Things*, vol. 7, p. 100 056, Sep. 2019. DOI: 10.1016/j.iot.2019.100056. Accessed: 2026-03-14.

[28] R. Khan, K. McLaughlin, D. Laverty, and S. Sezer, 'STRIDE-based threat modeling for cyber-physical systems', in *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, 2017, pp. 1–6. DOI: 10.1109/ISGTEurope.2017.8260283.

[29] M. Hamad, M. Nolte, and V. Prevelakis, 'Towards Comprehensive Threat Modeling for Vehicles', *Publications Institute of Computer and Network Engineering*, 2016. DOI: 10.24355/dbbs.084-201806251532-0.

[30] D. Meyer, J. Haase, M. Eckert, and B. Klauer, 'A threat-model for building and home automation', in *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, 2016, pp. 860–866. DOI: 10.1109/INDIN.2016.7819280.

[31] M. R. A. Asif, K. F. Hasan, M. Z. Islam, and R. Khondoker, 'STRIDE-based Cyber Security Threat Modeling for IoT-enabled Precision Agriculture Systems', in *2021 3rd International Conference on Sustainable Technologies for Industry 4.0 (STI)*, 2021, pp. 1–6. DOI: 10.1109/STI53101.2021.9732597.

[32] A. Omotosho, B. A. Haruna, and O. M. Olaniyi, 'Threat Modeling of Internet of Things Health Devices', *Journal of Applied Security Research*, vol. 14, no. 1, pp. 106–121, Jan. 2019. DOI: 10.1080/19361610.2019.1545278. Accessed: 2026-03-14. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/19361610.2019.1545278.

[33] B. J. Berger, K. Sohr, and R. Koschke, 'Automatically Extracting Threats from Extended Data Flow Diagrams', in *Engineering Secure Software and Systems*, J. Caballero, E. Bodden, and E. Athanasopoulos, Eds., vol. 9639, Cham: Springer International Publishing, 2016, pp. 56–71, ISBN: 978-3-319-30805-0. DOI: 10.1007/978-3-319-30806-7_4. Accessed: 2026-03-14. [Online]. Available: http://link.springer.com/10.1007/978-3-319-30806-7_4.

[34] L. Sion, K. Yskout, D. Van Landuyt, and W. Joosen, 'Solution-aware data flow diagrams for security threat modeling', in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, Pau France: ACM, Apr. 2018, pp. 1425–1432, ISBN: 978-1-4503-5191-1. DOI: 10.1145/3167132.3167285. Accessed: 2026-03-14. [Online]. Available: https://dl.acm.org/doi/10.1145/3167132.3167285.

[35] Fraunhofer Institute for Applied and Integrated Security (AISEC), 'A Study on Hardware Attacks against Microcontrollers', German Federal Office for Information Security (BSI), Tech. Rep., Mar. 2023.

[36] D. D. Gajski, S. Abdi, A. Gerstlauer, and G. Schirner, *Embedded System Design: Modeling, Synthesis and Verification*. Boston, MA: Springer US, 2009. DOI: 10.1007/978-1-4419-0504-8. Accessed: 2026-03-14. [Online]. Available: https://link.springer.com/10.1007/978-1-4419-0504-8.

[37] E. Bochniewicz et al., *Playbook for Threat Modeling Medical Devices*. MITRE Corporation and Medical Device Innovation Consortium (MDIC).

[38] M. Wolf, 'Combining safety and security threat modeling to improve automotive penetration testing', 2019, Publisher: Universität Ulm. DOI: 10.18725/OPARU-13062. Accessed: 2026-03-14. [Online]. Available: https://oparu.uni-ulm.de/xmlui/handle/123456789/13119.

[39] electrouser865, *Jaimico: The New I-health Care Companion Robot*. Accessed: 2026-03-14. [Online]. Available: https://www.electromaker.io/project/view/jaimico-the-new-i-health-care-companion-robot.

[40] emgarde, 'emgarde | TTModeler Pro', Accessed: 2026-03-14. [Online]. Available: https://emgarde.de.

[41] S. Liebl, *TTModeler*. Accessed: 2026-03-14. [Online]. Available: https://github.com/SecSimon/TTM.

[42] International Organization for Standardization and International Electrotechnical Commission, *ISO/IEC 25022:2016: Systems and Software Engineering - Systems and Software Quality Requirements and Evaluation (SQuaRE) - Measure of Quality in Use* (International standard). ISO, 2016. Accessed: 2026-03-14. [Online]. Available: https://books.google.de/books?id=Wzc1ygEACAAJ.

[43] J. Ziegeldorf, O. Morchon, and K. Wehrle, 'Privacy in the internet of things: Threats and challenges', *Security and Communication Networks*, vol. 7, Dec. 2014. DOI: 10.1002/sec.795.

[44] S. H. Simpson, 'Creating a Data Analysis Plan: What to Consider When Choosing Statistics for a Study.', *The Canadian journal of hospital pharmacy*, vol. 68 4, pp. 311–7, 2015.

[45] R. McGill, J. W. Tukey, and W. A. Larsen, 'Variations of box plots', *The American Statistician*, vol. 32, no. 1, pp. 12–16, 1978, Publisher: Taylor & Francis Group.

[46] J. Cohen, *Statistical power analysis for the behavioral sciences*, 2nd ed. Hillsdale, N.J: L. Erlbaum Associates, 1988, ISBN: 978-0-8058-0283-2.

[47] J. Hartung, G. Knapp, and B. K. Sinha, *Statistical Meta-Analysis with Applications* (Wiley Series in Probability and Statistics). Hoboken, NJ, USA: John Wiley & Sons, Inc., Jul. 2008, ISBN: 978-0-470-29089-7. DOI: 10.1002/9780470386347.

[48] Foreign, Commonwealth & Development Office, *Small to medium sized enterprise (SME) action plan*, May 2023. Accessed: 2026-03-14. [Online]. Available: https://www.gov.uk/government/publications/fcdo-small-to-medium-sized-enterprise-sme-action-plan/small-to-medium-sized-enterprise-sme-action-plan.

[49] R. Likert, 'A technique for the measurement of attitudes.', *Archives of Psychology*, vol. 22 140, pp. 55–55, 1932.

SEPP 4.0 - Evaluation of Hands-On IoT-Security Exercises

Louis Ebneth and Sebastian Fischer
Faculty of Computer Science and Mathematics
Ostbayerische Technische Hochschule Regensburg
Regensburg, Germany
e-mail: louis1.ebneth@st.oth-regensburg.de | sebastian.fischer@oth-regensburg.de

***Abstract*—Teaching Internet of Things (IoT) security effectively remains a challenge due to the gap between theoretical concepts and their application in real-world systems. This paper presents the fourth iteration of the Security Education and Penetration-Testing Platform (SEPP) project, a gamified, hands-on learning environment that uses vulnerabilities in real IoT devices to teach IoT security concepts to computer science students. Over the course of the past semester, a weekly exercise session was conducted, during which ten previously developed exercise sheets were evaluated through observations and student questionnaires. The results show high student engagement, strong perceived learning outcomes, and clear benefits from working with realistic attack scenarios and real IoT devices. However, the evaluation also revealed areas in need of improvement, including device selection, sequencing of the exercises, clarity of instructions, preparation of the exercise sessions, and time management. Based on these findings, this paper details the next steps in the further development of the SEPP platform and the continuous improvement of practical IoT security education at the OTH Regensburg.**

Keywords-Internet of Things; IoT Security; Cybersecurity Education; Hands-On Learning.

I. Introduction

The Internet of Things (IoT) has become an integral part of our daily lives. From smart home appliances, to wearables, and industrial equipment, the ability to connect devices and systems across the internet has opened up new possibilities for automation, control, and monitoring in all aspects of life [1]. The heterogeneous nature of the IoT ecosystem and the rapid development of IoT devices in recent years have resulted in a lack of robust security mechanisms and created vulnerabilities that expose IoT systems to a wide range of cyberthreats [2][3].

A. SEPP - Security Education and Penetration-Testing Platform for IoT

The SEPP platform presents a gamified approach to teaching IoT Security to computer science students by providing the students with a way of engaging with real-world examples of the cybersecurity risks introduced in the IoT Security lecture through a series of hands-on exercises [4][5].

The platform simulates a home environment equipped with various IoT devices that contain security flaws, such as smart plugs, security cameras, or light bulbs. Each device is associated with one or more exercise sheets that have been created by students as part of their bachelor's thesis or study project.

B. Evaluation of the Exercises

Over the course of the past semester, two students conducted a weekly exercise session, where they evaluated the ten exercise sheets based on observations during the session and a questionnaire that the participating students had to fill out at the end of the session [6][7].

Section 2 provides a brief overview of the exercise contents. Section 3 discusses the first evaluation results. Section 4 outlines the next steps in the further development of the SEPP platform.

II. Exercises

The exercise sheets were designed as the practical part of the IoT Security lecture and are therefore aligned to complement its contents. The students work with the industry-standard penetration testing tools introduced in the lecture and perform a wide variety of different attacks and audits on real IoT devices.

The exercises cover a wide range of protocols used in IoT devices, including Wi-Fi, Bluetooth, Bluetooth Low Energy (BLE), Telnet, Secure Shell (SSH), Message Queuing Telemetry Transport (MQTT), Transport Layer Security (TLS), and more.

Exemplary tasks include:

1) Determining the Internet Protocol (IP) address of a smart outlet by conducting a network scan using the network exploration tool Nmap.
2) Capturing control commands sent to an IoT light bulb via the Telnet protocol using Wireshark and subsequently performing a replay attack by retransmitting the captured packets with Python.
3) Performing an Address Resolution Protocol (ARP) spoofing-based Man-in-the-Middle (MITM) attack on a Siemens LOGO! DDC module using Ettercap to sniff and intercept its network traffic.
4) Executing Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks on various devices using a Python script.
5) Gaining access to a web interface protected by default credentials through a dictionary-based brute-force attack with the Python package Selenium.
6) Hijacking the active user session of a web interface by recreating a session cookie from a Wireshark capture containing a transmission of the session ID.

III. Discussion of the First Results

In general, the feedback on the exercises was found to be positive. The connection to the IoT Security lecture was perceived as apparent, and the students stated that they were able

to apply what they had learned. The students highly praised the hands-on approach and especially emphasized the joy of being able to compromise real IoT devices, particularly when the instructions provided them with very limited information about the device to start with. The different attack vectors covered in the exercises were reviewed as interesting and educational, offering clear perspectives on cybersecurity issues in IoT devices. For most exercises, the instructions were rated as understandable and easy to follow, and one exercise even received praise for its puzzle-like design. The students were surprised by the ease of using tools like Nmap and Wireshark to collect the data necessary to perform the various attacks. The simplicity of various attacks was also pointed out. This included the ability to perform a DoS attack with a simple Python script and the ability to perform a MITM attack with a single Ettercap command.

The following subsections highlight the core issues that were found during the evaluation and how to best address them.

A. Device Choice

The choice of the IoT device was found to be suboptimal for some exercises.

For example, the students pointed out that determining the success of a DoS attack on a smart Wi-Fi outlet is rather difficult if there is no device plugged into it that would indicate the failure of sending control sequences from the smartphone application to the outlet. Another exercise had the students perform a replay attack on an IoT light bulb. However, the device was unable to handle the increased number of requests created from multiple groups of students doing this at the same time. The attack therefore resulted in a DoS scenario, with the bulb becoming unresponsive for a few minutes.

The victim devices used in these exercises therefore need to be reevaluated. They should provide clear feedback on the success of an attack whenever possible and should not exhibit any unintended behavior.

B. Repetition of Attacks

The steps of first identifying a device on the network with Nmap and subsequently performing a DoS attack on it were included in four of the ten exercise sheets. Repeating the same attack, especially in an equivalent fashion, should be avoided, as this does not provide any additional educational value.

C. Arrangement within the Semester

The first exercise used Nmap to identify a smart outlet, launched a DoS attack, and analyzed the resulting traffic with Wireshark. The second exercise covered vulnerabilities in Wi-Fi Protected Access (WPA) and Wi-Fi Protected Setup (WPS) and again required device identification using Nmap. However, the chapters that introduce Nmap, WPA, and Wireshark have not been discussed in the lecture at this point of the semester.

This emphasizes the need to re-evaluate the arrangement of the exercises on the semester timeline, as it is not feasible to have the students working with tools before they have been introduced in the lecture.

D. Instructions and Guidance

All instructions should be clear and easy to follow.

Over the course of the evaluation, multiple errors in the exercise instructions were identified that need to be corrected. This includes incorrect sample commands and Python scripts that did not produce the desired results. Additionally, since the exercise sheets were created at home by different students, discrepancies arose between the instructions and the final computer lab setup. This includes incorrect Wi-Fi Service Set Identifiers (SSIDs) and passwords, as well as mismatched Media Access Control (MAC) addresses, all of which need to be corrected as well. In some cases, the students remarked that the Python scripts they had to complete as part of the exercise did not contain any information about what to do beyond a *"TODO"* comment marking the line. Additional comments that explain what is to be done need to be added to these scripts. Some instructions were also found to be too vague and need to be concretized. One example had students analyze a Wireshark network capture without specifying what to look for in the packets, another step only referred to *"any Wi-Fi device"*, which left the students wondering which one to use.

The tools and Python libraries that are used in the exercises, but not covered by the lecture, should be introduced with a short description of the tool and should provide concrete example commands. This can be done in short on the instruction sheet itself or on a separate information sheet for a more extensive resource. Exemplary resources include Ettercap, Reaver, Selenium, and Bleak.

Some students also provided feedback on the lack of such information and examples for the tools that were introduced in the lecture. However, this could easily have been solved by attending the lecture or taking a look at the course material. In addition, online research and AI tools, such as ChatGPT, can provide further guidance.

E. Understanding

The goals and key takeaways of each exercise should be clear. This was not always the case.

One exercise had the students read out information from a Bluetooth smart lock using the nRF Connect app. The subsequent steps of the exercise did not utilize this information further or explain how it could be used, which left the students wondering about the relevance of this step. Another exercise had the students remark that a session hijacking attack on a Siemens LOGO! DDC module was too easy, as the session ID was part of the Uniform Resource Locator (URL) of the web interface and could therefore easily be captured with Wireshark, and that the web interface did not provide feedback when the attack was successful.

It should be made clear that these are vulnerabilities in real IoT devices and not custom-engineered lab creations.

F. Preparation of the Exercise Sessions

In an ideal situation, all devices would be firmly integrated into the SEPP platform and configured as necessary for the

exercises. However, this is not yet possible due to the still-evolving nature of the project.

The exercise leader must therefore be provided with a clear and easy-to-follow setup guide for each exercise that outlines what has to be prepared and checked before the exercise session. This is currently only the case for three out of the ten exercises, with one of the guides being received as insufficient.

G. Wi-Fi Issues

The original setup of the SEPP had a Raspberry Pi 5 hosting a Wi-Fi network to which the students and IoT devices were connected. This network was found to be rather inconsistent and unable to support the desired number of devices. Midway through the semester, a GLMT300N-V2 Mini Smart Router was added to serve as an access point and replace the Raspberry Pi's Wi-Fi.

H. Timely Constraints

The weekly exercise session has a duration of 90 minutes. The time required for the individual exercise sheets varied greatly. Some exercise sheets were too extensive to be completed in 90 minutes, while the students finished others within 45 minutes. The length of existing exercises must be adjusted to approximately match the length of the exercise session. Future exercise sheets should also be designed with the time frame in mind and need to be tested for compliance prior to their inclusion in the course.

Three areas were clearly identified where time could be saved during the sessions:

1) Provisioning of Necessary Tools: About a third of the first exercise session was lost because the students had to install the required tools. The resources and tools required for the exercise should therefore be communicated in advance of the weekly session. Later sessions had a message posted on the E-learning platform a couple of days prior, that provided the students with a list of the necessary tools and instructed them to install these. Tools that are used repeatedly across most exercises, such as Nmap and Wireshark, can be communicated to all students in the first lecture at the beginning of the semester. Alternatively, every student can be tasked with installing a Kali Linux virtual machine in a free virtualization software like VirtualBox, as this distribution already contains most of the required tools. For this, links to setup tutorials should be provided. Another option is to provide students with ready-to-go native Kali Linux laptops and Android phones in the computer lab. This would also eliminate issues, such as a virtual machine not having access to the Wi-Fi adapter of the host device, and therefore being unable to switch it into monitor mode, or issues with the iOS version of the nRF Connect app not displaying certain information, such as the Bluetooth address of an IoT device.

2) Handling of Lengthy Subtasks: Some steps, such as a full Nmap User Datagram Protocol (UDP) port scan, are known to take quite a while. The instructions did not mention limiting the range of ports to scan, which caused the students to wait for the full scan to finish. This was especially prominent if the results of such a step were needed for the further steps of the exercise. Appropriate tips should be added to steps like this in order to minimize wasted time.

3) Distribution of Files: The students work with numerous Python scripts throughout the exercises. In some cases, these scripts were provided only as plain text or images on the exercise sheets, which the participants had to manually transcribe into their editors. One exercise also involves a Java-based server application for which only a compiled **.jar* file and screenshots of the source code were available.

A central Git repository is proposed to manage and efficiently distribute all source files. This repository could also be extended to hold and distribute the exercise sheets.

IV. Conclusion and Future Work

The initial results of the evaluation indicate that the SEPP platform is a promising and well-received tool to educate future IT professionals about IoT cybersecurity. However, they also show that it is still an evolving project that is far from complete and will require continuous improvement in the future. The issues mentioned in Section 3 will be addressed over the course of the next semester, as the lecture is only held in the winter term.

The results of this paper can also be used outside the SEPP platform to better design similar practice-oriented exercises based on this experience. Therefore, the results not only serve to improve the platform, but can also help other educators with the development of better tasks.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010, Retrieved: 2026.02.18, ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2010.05.010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128610001568.

[2] R. H. Weber, "Internet of Things – New security and privacy challenges," *Computer Law & Security Review*, vol. 26, no. 1, pp. 23–30, 2010, Retrieved: 2026.02.18, ISSN: 2212-473X. DOI: https://doi.org/10.1016/j.clsr.2009.11.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0267364909001939.

[3] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: The road ahead," *Computer Networks*, vol. 76, pp. 146–164, 2015, Retrieved: 2026.02.18, ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2014.11.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128614003971.

[4] D. Hauser, J. Graf, S. Fischer, and R. Hackenberg, "SEPP: Security Education and Penetration-Testing Platform for IoT," in *Proceedings of The European Conference on Education 2025*, Retrieved: 2026.02.18, 2025, pp. 443–453. DOI: https://doi.org/10.22492/issn.2188-1162.2025.36.

[5] D. Hauser and S. Fischer, "SEPP 2.0 - Advanced IoT Hacking Scenarios for Hands-on Security Education," 2025, Unpublished.

[6] I. Edelmann, "IoT Vulnerabilities: Evaluation and Improvement of the Exercises," Unpublished bachelor's thesis, 2025.

[7] L. Herrmansdörfer, "Evaluation and Analysis of Cyber-Security Challenges in Relation to the Internet of Things," Unpublished bachelor's thesis, 2025.

The Development of an IoT-Focused Investigative Methodology:

The Case of a Pico 4 Headset

Luke Yates
Dept. of Computing
Buckinghamshire New University
High Wycombe, Buckinghamshire
email: luke.yates@bucks.ac.uk

Professor Ian Fergusson
Dept. of Cybersecurity and Computing
Abertay University
Dundee, Scotland
email: i.ferguson@abertay.ac.uk

Dr Karl van der Schyff
Dept. of Cybersecurity and Computing
Abertay University
Dundee, Scotland
email: k.vanderschyff@abertay.ac.uk

***Abstract*—The paper is submitted to the conference as it is relevant to Internet of Things and Forensics in Constrained Environments. The rapid proliferation of Internet-of-Things (IoT) devices, including Virtual Reality (VR) headsets, has introduced new opportunities and challenges for digital forensics. This study focuses on the Pico 4 VR headset, a next-generation Android-based device, to explore forensic methodologies for extracting and analyzing digital artifacts. Crimes involving VR, such as harassment, fraud, and illegal content distribution, highlight the urgency for specialized investigative approaches. Using the Android Debug Bridge (ADB) suite to capture device data, we developed a forensic methodology that ensures data integrity while navigating the limitations of non-rooted devices. Our investigation reaffirms the potential of ADB commands, including backup and dumpsys, to collect system information, application data, and user-generated content from the Pico 4 in a forensically sound manner. This paper focuses on the most significant aspect of this research, where a custom module for Autopsy facilitated efficient web browser data processing. However, the challenges posed by device security, inaccessible storage areas, and encrypted communications underscore the need for adaptable and innovative forensic techniques. The research describe here contributes to the field by presenting the first digital forensic approach tailored to the Pico 4. The findings emphasize the importance of combining traditional forensic tools with bespoke methodologies to address emerging IoT devices. Our work provides a foundation for future studies on VR-related crime investigations, offering practical insights for digital forensic practitioners navigating the evolving landscape of IoT technologies.**

Keywords–IoT; forensics; Pico 4; Digital forensics; Android devices; VR forensics; Digital forensic methodology.

I. INTRODUCTION

An exponential growth of the Internet of Things (IoT) [1] means that more people are using non-traditional computing devices to access the internet, for legal and non-legal purposes. This includes the use of Virtual Reality (VR) headsets to commit fraud, harassment, identity theft and most seriously, access and distribution of illegal images of children [2]. Utilising modern VR devices to carry out these and potential new types of crime creates the potential to hide evidence from outdated digital forensics methodologies. It is therefore imperative that the science of Digital Forensics is mindful of the challenges VR technology presents, when conducting investigations. Digital forensics experts are required to carry out their investigations in a "forensically sound" manner that seeks to protect the integrity of data obtained from devices, as well as ensuring that no harm comes to devices procured in an investigation. They must also adopt techniques that consider the security restrictions on Android systems such as those on VR headsets and be mindful that there are often restrictions on what can be obtained.

The Pico 4 released in October 2022 [3] by ByteDance headset is an example of a newer Android-based VR device. A range of apps and games are available, both standalone on the device and via connection to a computer or phone. Whilst a considerable amount of research has been carried out on IoT and VR devices, the Pico 4 is relatively new technology and hence it is not known what useful digital forensic data may be stored on it. If investigators are unclear with regards to the digital forensic techniques to use on such a device, inadequate amounts of evidence will likely be collected, negatively impacting the effectiveness of an investigation or even resulting in evidence being excluded from court, which could have disastrous consequences for the case to be made.

This leads to the formulation of two research questions:

RQ1. What type of digital artefacts are typically acquired in a VR-focused digital forensic investigation, and how could these be extracted in a forensically sound manner? To develop a sound investigative methodology, it is important to know which type of artefacts are likely to be encountered and where to locate them. Handling such artefacts in an appropriate manner may require additional understanding – all of which could aid investigators faced with VR-focused investigations.

RQ2. Which techniques are suitable to aid investigators in identifying and reporting relevant digital forensic artefacts as evidence (in a forensically sound manner) when considering Android-based IoT devices like the Pico 4 VR headset? In addition to the type and location of the artefacts, it is also crucial to understand the techniques required to

 ISBN: 978-1-68558-442-9

generate useful information in reporting these findings and their relationship to the case being investigated. Some existing techniques may prove useful – yet others may have to be adapted (erg., Autopsy modules).

Our study is structured as follows: We first present readers with a review of related literature in Section II before providing a complete outline of our methodological approach in Section III. This is followed by the results of our evaluation in Section IV and an associated discussion, including the theoretical and practical implications in Sections V and VI. The study concludes with a brief discussion of the limitations and areas of future research in Section VII and a final conclusion in Section VIII.

II. LITERATURE

Guided by our first research question, the review will first provide readers with an outline of how digital forensic data should be acquired, followed by specifics as to how this relates to Android-based (thus Pico 4 related) devices

A. Acquiring digital forensic data from IoT devices

Regarding digital forensics, Alazab et.al. [4] states that IoT forensics requires a more specialised approach but has the potential to yield highly useful information that traditional computing lacks, such as motion sensor, GPS, camera, and microphone data. However, issues with IoT forensics are also considered, such as "inappropriate handling of data, securing the chain of custody and lack of standardisation". In addition, the concept of "cloud jurisdiction" is discussed. This is where data on the cloud from an IoT device may be stored in other countries, and hence subject to different rules and regulations regarding privacy and data protection, creating new obstacles for digital forensic investigations.

Research carried out on a SteamDeck handheld console systematically identified forensically relevant artefacts [5]. This device is not a typical IoT device and runs a Linux-based "SteamOS" operating system, however the differential forensic analysis approach used in the research holds value for IoT digital forensics. This is where "before and after" snapshots of the device data are taken, to see what has changed. This narrows the search down and expedite the locating of relevant forensic data. The data identified by this research as relevant, included device information, Wi-Fi connections, user accounts, games and apps, screenshots and images taken or downloaded to the device, friends (connections to other users) and financial transactions.

Raymer et.al. identify the importance of forensic techniques on VR devices in regard to the potential for crime, such as online harassment, grooming and cyberbullying [6]. A forensic analysis on a Meta Quest 2 device was performed, providing helpful and relevant information for when tackling VR headsets – specifically Android-based devices. Their research [6] confirmed that data can be retrieved from such a device using forensically sound practice, as well as describing locations and types of useful forensics data (live user activity, user files and live device data). Raymer et.al. [6] described their solution for a formal acquisition process, in which they identified storage types and locations on the device, then recover and analyse digital artifacts pertaining to a digital forensic investigation.

The three main areas investigated were user data, application data and live system data. Raymer et.al. [6] note that a limitation when working with the Quest 2 VR device as with all Android-based devices, is the restriction imposed on access to certain parts of storage, such as system files, and the only way to overcome this being to root the device being examined.

B. Investigating Android-based IoT devices

Taylor et. al. [7] examined the process of investigating mobile apps, as well as issues relating to obtaining digital evidence from such devices. Of particular interest is their research on Android apps and security. The research identified that each Android application runs as a unique user identity, which enhances security, and, consequently, makes acquiring data for digital forensics from such devices, more challenging. Additionally, this research identified the range of crimes that may be carried out by utilising digital devices, such as fraud, theft, money laundering, copyright infringement, and the possession and/or distribution of indecent images.

When considering digital forensics, the advantage of acquiring a rooted device is that it is possible to make physical images of the storage and data on that device, rather than just logical (files and folders) level copies. This allows retrieval of files and folders which have been deleted, as well as all applications and data. However, Android devices are not released in a rooted state [8], and obtaining an IoT Android device that is rooted is not always possible. This is because rooting such a device is extremely difficult to carry out on a new device, and introduces the risk of the evidence being no longer forensically sound. For the owner of the device, there is a risk of compromised security, voided warranty, unavailable services (such as online purchasing and banking), device instability, and in the worst case, the damage or destruction of an expensive piece of hardware. With new devices, such as the Pico 4, manufacturers will lock the boot-loader to discourage rooting attempts, and as rooting usually requires the exploitation of a vulnerability, a methodology may not exist at all to unlock the hardware in this way.

Kara [9] performed specific research on extracting Discord data from an Android device, and whilst successful in retrieving artifacts, such as contacts, messages, and deleted items, as well as understanding the data structure of Discord, this methodology relies on rooting or jail-breaking [10] the Android device, something that is either impossible or at least highly risky when dealing with newly released Android devices. Additionally, this approach copies files to the laboratory computer; files copied to another device will have their dates/times altered, harming their integrity as evidence. Kara [9] also observes that any security measures put into place on the suspect Android device, such as

biometrics or PINs, could make retrieving data a far more complex process.

It is worth observing that whilst Android employs strong mechanisms in its design which can protect privacy, but interfere with digital forensics investigations, user naivety over security and privacy settings can sometimes compromise this and could work in the favour of a digital forensics investigation. The research by Neisse et.al. [11] considers this and notes that the complexity of the permission-granting mechanism in Android devices means that users often fail to understand the privacy implications of granting apps permission to sensitive resources.

C. *The role of the Android Debug Bridge*

Easttom & Chuck [12] describe a methodology for SMART TV forensics in which they justify and then use the Android Debug Bridge (or ADB). Importantly, and as evidenced in their work, this tool is recognised as a means to extract artefacts in a forensically sound manner and therefore features in our methodology.

III. METHODOLOGICAL APPROACH

This section provides an outline of our methodological approach and investigative methodology in particular regard to the artifact developed for Autopsy. We start by first describing the methods used to prepare the hardware and software used. This is followed by details of how the initial data was captured from the Pico 4 headset, after which an explanation is provided of the test apps (and data) used. This section concludes with a detailed outline of the data acquisition process as well as the methods used for searching and data analysis. Note that this study received full ethical clearance from the primary author's university (ref: EMS8927).

Using the techniques identified in the literature study, images of the data were captured from the Pico 4 using ADB's "Android Backup", thus maximising integrity of data and forensic soundness. The first image was taken immediately after a factory reset of the Pico device and resulted in 49.38MB of data stored in the device's "apps" and "shared" folders (also known as the /sdcard folder on Android devices). The device was then used in a variety of common ways as it might be used by a typical user, including installing and using the Wolvic browser to visit websites and download 10 image files (done using the browser's "Save Image As" feature). A second image of the "apps" and "shared" folder data was then captured, this resulted in a file size of 13.38GB. Differential analysis was used to identify new or changed files in this second backup, leading to the identification of the database files relating to web browsing history. The analysis was done using a comparison of MD5 file hashes, to ensure identical files from both images could be safely ignored.

Investigating the browser history has not only become commonplace, but also crucial when trying to establish intent and was hence the main focus of this research. Given our use of the Wolvic browser, we were able to identify an SQLite database file (places.sqlite). The analysis of this database was automated by writing a module for Autopsy, a well-known open-source tool used for digital forensics. Autopsy uses a Jython parser, allowing Python code to interface with the Java-based Autopsy. A code template (for developing Autopsy modules) was acquired from GitHub [13] and rewritten to work with the data extracted from the Wolvic browser. The bespoke code connects to the web browser database file, and reads off rows in the web history table. It then creates an event for each visit to a website URL. It then creates an Autopsy "blackboard" artifact containing retrieved information about the website URL, web page title and date/time visited, and places this new artifact in Autopsy's "Web History" container. The flowchart for this algorithm is presented here:

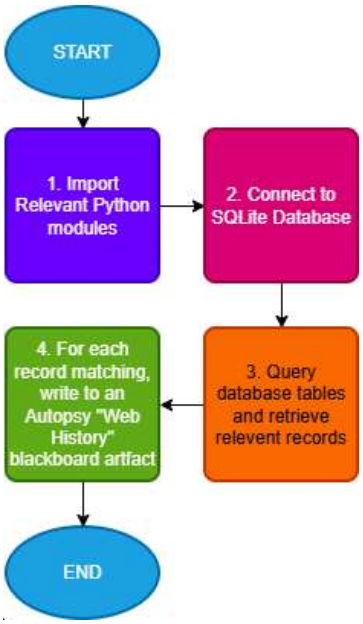


Figure 1. Flowchart for Web Browser DataRetrieval

Additionally, the relevant excerpt of the Python code to read the database and create Autopsy artifacts is shown here:

```
# Code to connect to database and extract browser history items

        # Query the browser history tables in the database and get
necessary columns.
        # we need URL, Title and Visit Date
        try:
            stmt = dbConn.createStatement()
            resultSet = stmt.executeQuery("SELECT moz_places.url as
'URL', moz_places.title as 'Website Title', moz_historyvisits.visit_date as
'Visit Date' FROM moz_places  INNER JOIN moz_historyvisits ON
(moz_historyvisits.place_id = moz_places.id) ORDER BY 'Visit Date'")
        except SQLException as e:
            self.log(Level.INFO, "Error querying database for browser
history tables (" + e.getMessage() + ")")
            return IngestModule.ProcessResult.OK
```

```
        # Cycle through each row and create artifacts
        while resultSet.next():
            try:
                url  = resultSet.getString("URL")
                title = resultSet.getString("Website Title")
                visit_date_local = int(resultSet.getString("Visit Date"))/1000
# UNIX Timestamp is stored in db as milliseconds - needs to be in
seconds!
            except SQLException as e:
                self.log(Level.INFO, "Error getting values from browser
history tables (" + e.getMessage() + ")")


            # Make an artifact on the blackboard, TSK_WEB_HISTORY
and give it attributes for each of the fields
            art =
file.newDataArtifact(BlackboardArtifact.Type.TSK_WEB_HISTORY,
Arrays.asList(
                BlackboardAttribute(BlackboardAttribute.Type.TSK_URL,

WebHistoryDbIngestModuleFactory.moduleName, url),
                BlackboardAttribute(BlackboardAttribute.Type.TSK_TITLE,

WebHistoryDbIngestModuleFactory.moduleName, title),

BlackboardAttribute(BlackboardAttribute.Type.TSK_DATETIME_ACC
ESSED,

WebHistoryDbIngestModuleFactory.moduleName, visit_date_local)
            ))

            try:
                blackboard.postArtifact(art,
WebHistoryDbIngestModuleFactory.moduleName,
self.context.getJobId())
                recordCount +=1
            except Blackboard.BlackboardException as e:
                self.log(Level.SEVERE, "Error indexing artifact " +
art.getDisplayName())

        # Clean up
        stmt.close()
        dbConn.close()
        os.remove(lclDbPath)

    # Feedback to user how many database files were found.
    message =
IngestMessage.createMessage(IngestMessage.MessageType.DATA,
        "Wolvic VR Browser History Analyzer", "Found %d matching
database files" % fileCount)
    IngestServices.getInstance().postMessage(message)

    # feedback to user how many database records were turned into
Autopsy browser history artifacts.
    message =
IngestMessage.createMessage(IngestMessage.MessageType.DATA,
        "Wolvic VR Browser History Analyzer", "Processed %d browser
history artifacts" % recordCount)
    IngestServices.getInstance().postMessage(message)

    return IngestModule.ProcessResult.OK
```

IV. Evaluation of Results

The digital forensics methodology devised and used in this research managed to retrieve a significant yield of files relevant to a forensics investigation, in a forensically sound manner, making them admissible as evidence in a legal case.

A combination of the techniques described in the literature review, as well as the usage of the popular open source forensics application, Autopy as described in section IV, yielded 293 image files, 8 videos, and of particular interest to this paper, 33 database files. System and application log files were also discovered containing details of user accounts created and used on the device – these matched ones created for test purposes during the methodology phase both in name and in creation date and time. Of additional interest were username and password credentials which were not created by test users accounts, coded into several web.cfg files. This information disclosure could pose a security risk.

From these database files, it was possible to retrieve digital artefacts, such as browser history, user input and cookie data from the Wolvic web browser. One of the related SQLite databases named places.SQLlite contained a table called moz_places which held information about every unique website visited, along with the date and time of the most recent visit (held in a UNIX timestamp). Where these websites had been visited more frequently, we found corresponding entries in another database table called moz_historyvisits. This latter table linked to the moz_places table and held access dates and times in UNIX timestamps.

Overall, the bespoke Autopsy module developed during this research, found 100 web history artifacts, including the title, URL and access date/time of every website accessed using Wolvic browser as well as the 10 images downloaded intentionally during the data creation stage. The data revealed by the Autopsy plugin was able to be cross-referenced successfully with the URLs, titles and dates of the sites visited, and datetime file stamps on the image files themselves also corresponded with site visit dates and times, further enhancing the credibility of this evidence.

V. Discussion

This study aimed to develop a digital forensic methodology that could be used to investigate Pico 4 VR headsets. We argue that although similar methodologies have been developed for other VR headsets (erg., Meta Quest), our approach contributes to the field of digital forensics as the first to methodically focus on the Pico 4 as well as showing that existing technology, such as Autopsy can be utilized for newer devices with the help of custom plugins.

Two research questions were posed and addressed in this research:

A. Addressing RQ1: Types of artifacts identified

The investigation of the Pico 4 headset showed that analysis of web browser activity, done in this instance using a bespoke coded plugin solution for Autopsy, can be extremely lucrative for uncovering highly useful artifacts. This can then be used to create a timeline of a suspect's online activities. The evidence is forensically sound as the times and dates discovered by the plugin, were shown to be consistent with actions taken to initially create the artifacts.

B. *Addressing RQ2 – techniques used:*

Once web browser database files have been acquired in a forensically sound manner using the techniques discussed in the literature review and in the methodology, these can be passed to an automated digital forensics application, such as Autopsy, making the process of identification of relevant artifacts less technical, and hence easier. This aids in maintaining the integrity of evidence. Problems exist with tools, such as Autopsy, when using it to analyse new devices, such as the Pico 4, as they will miss certain types of data.

However, it is possible to write modules for Autopsy that can work specifically with newer devices and the data created and stored by their applications. These modules are written in scripting languages, such as Python, and once implemented, can allow Autopsy to scan and detect new artifacts, add them to its catalog and include them in its excellent reporting systems, such as timelines.

VI. THEORETICAL AND PRACTICAL IMPLICATIONS

The research showed that due to the nature of the file and security structure of Android devices, coupled with the freedom for developers to create and store data anywhere on the device, limits what data can be recovered, and means there is potential for misuse by criminals. Various Android apps allow encrypted, private conversations to take place, and the data for these cannot be retrieved or analysed. Whilst this may be reassuring to many users, it creates issues for digital forensic investigators, who may be unable to capture this information for their evidence files. This secrecy afforded on non-rooted devices could enable computer-based crimes, such as harassment, abuse and fraud, as well as aiding communication and planning of non-computer-based crimes which could have even more dire consequences. These limitations mean that there are areas of research into the Pico 4 (and other IoT devices) that lack clarity.

Web browsers, such as the Wolvic browser which can be installed from the Pico app store, successfully yield all of their browser history, cookies, user input and downloaded files. This information is contained within database files, which can be easily read and processed by suitable coded modules for forensic apps, such as Autopsy.

This research provides an example of such scripting using Python coding, and this could be drawn on as a starting place to develop similar scripts for future IoT and VR devices.

VII. LIMITATIONS AND FUTURE RESEARCH

While this study provides a foundational forensic methodology for investigating the Pico 4 VR headset, several limitations remain.

The experimental design involved artificially generating test data on the Pico 4 rather than analyzing real-world forensic case data. While this method ensures controlled conditions for evaluating forensic techniques, it does not account for the complexity of real investigations where data may be deliberately hidden, fragmented, or manipulated. Future studies should seek to apply the methodology in real investigative contexts, analyzing seized VR devices from actual cases to assess the practical challenges and effectiveness of the proposed techniques in more complex scenarios.

A comparative study evaluating forensic techniques across multiple VR headsets, such as the Meta Quest 2 and HTC Vive, could provide a broader understanding of forensic methodologies applicable to different hardware and software ecosystems.

One area requiring further research is the role of forensic automation and artificial intelligence in VR investigations. While this study successfully integrated Autopsy for data analysis, the development of custom forensic modules was not fully explored. Automated techniques, including machine learning-driven artifact classification, could enhance the efficiency of forensic investigations by reducing manual effort and improving the detection of relevant artifacts. Additionally, exploring the use of AI-driven analysis in VR-specific forensics, such as behavioral pattern recognition within virtual environments, could open new avenues for detecting criminal activity within immersive platforms.

VIII. CONCLUSION

The objective of this study was to develop a forensic methodology tailored to the Pico 4 VR headset – with this paper focusing primarily on the Autopsy plugin developed. Our findings indicate that it is indeed possible to use open source techniques to forensically investigate the Pico 4 headset without having to root it. These findings, and our methodology, provide investigators with a practice-oriented approach they could use to investigate VR-based devices. Together with the Autopsy plugin we developed, this study contributes on three fronts. First, it establishes a structured digital forensic methodology tailored for the Pico 4 VR headset. Second, it provides a detailed analysis of artifact extraction techniques and their forensic significance, serving as a practical reference for investigators in the field. Third, it practically demonstrates how investigators could extract evidence from Android-based IoT devices without the need to perform low-level modifications (i.e., rooting) yet still maintain evidence integrity. We argue that, together, these contributions not only advances our theoretical understanding of VR-based evidence but also enables investigator to extract it.

ACKNOWLEDGMENT

The author wishes to thank Professors Karl van der Schyff and Ian Fergusson for their invaluable support and guidance throughout the development of this work.

REFERENCES

[1] Statista, “IoT connections worldwide 2022-2033,” [Internet], 2022. Available from: https://www.statista.com/statistics/1183457/iot- [retrieved: March, 2026].

[2] A. Crawford. “Child abuse material found on VR headsets, police data shows,” BBC News [Internet], 2023. Available from: https://bbc.co.uk/news/uk-64734308 [retrieved: March, 2026].

[3] A. Gutierrez. “Pico 4: features, price and specifications of the new VR headset,” Ludusglobal.com [Internet], LUDUS TECH SL, 2023. Available from: https://www.ludusglobal.com/en/blog/pico-4-features-price-specifications-vr-headset [retrieved: March, 2026].

[4] A. Alazab, A. Khraisat, and S. Singh. “A Review on the Internet of Things (IoT) Forensics: Challenges, Techniques, and Evaluation of Digital Forensic Tools,” IntechOpen [Internet], 2023. Available from: https://www.intechopen.com/online-first/86010 [retrieved: March, 2026].

[5] M. Eichhorn, J. Schneider, and G. Pugliese. “Well Played, Suspect! – Forensic examination of the handheld gaming console ‘Steam Deck’,” Forensic Science International: Digital Investigation, vol. 48, p. 301688, Mar 1, 2024.

[6] E. Raymer, A. MacDermott, and A. Akinbi. “Virtual reality forensics: Forensic analysis of Meta Quest 2,” Forensic Science International: Digital Investigation, vol. 47, p. 301658, Dec 1, 2023. [retrieved: March, 2026].

[7] M. Taylor, G. Hughes, J. Haggerty, D. Gresty, and P. Almond. “Digital evidence from mobile telephone applications,” Computer Law & Security Review, vol. 28, no. 3, pp. 335–339, Jun 1, 2012. [retrieved: March, 2026].

[8] A. Arslan. “Why Doesn’t Android Come Rooted?” MUO [Internet], 2013. Available from: https://www.makeuseof.com/tag/why-doesnt-android-come-rooted [retrieved: March, 2026].

[9] İ. Kara. “Digital Forensic Analysis of Discord Mobile Application on Android Based Smartphones,” Acta Infologica, vol. 0, no. 0, Oct 31, 2022.

[10] K. Gupta, P. Lanka, and V. Cihan. “A holistic digital forensic analysis of Discord – Storage, memory, and network perspectives,” Journal of Forensic Sciences, vol. 69, no. 4, pp. 1320–1333, May 28, 2024.

[11] R. Neisse, G. Steri, D. Geneiatakis, and I. N. Fovino. “A privacy enforcing framework for Android applications,” Computers & Security, vol. 62, pp. 257–277, Sep 2016.

[12] ProQuest. “A Methodology for Smart TV Forensics,” Proquest.com [Internet], 2021. Available from: https://www.proquest.com/docview/2505730094 [retrieved: March, 2026].

[13] M. McKinnon. “Data Source Ingest Module Template,” [Online], 2022. Available from: https://github.com/sleuthkit/autopsy/blob/develop/pythonExamples/dataSourceIngestModule [retrieved: March, 2026].

Performance Analysis of a Container Based Security Approach in 5G Networks

Musab M. W. MohamedAli‡ Alessandro Carrega*‡ Roberto Bruschi*‡ Ramin Rabbani‡

* Department of Electrical, Electronic and Telecommunications Eng., and Naval Architecture (DITEN)
University of Genoa (UniGe), Italy {*name*}.{*surname*}@unige.it
‡ National Laboratory of Smart and Secure Networks (SN2) of the
National Inter-university Consortium for Telecommunications (CNIT), Genoa, Italy {*name*}.{*surname*}@cnit.it

***Abstract*—The cloud-native and virtualized design of 5G networks introduces new security challenges that traditional perimeter-based solutions cannot adequately address. Container-native runtime security tools provide real-time visibility and policy enforcement for containerized network functions. This paper evaluates the performance of Falco and NeuVector for securing 5G core network functions in a cloud-native testbed.In our deployment, both tools operate as node-level agents (DaemonSet-based monitoring components). The impact of each tool on system performance is evaluated separately by measuring CPU usage, memory consumption, and scalability under simulated security event workloads. Experimental results indicate that both solutions introduce moderate CPU overhead during event processing while maintaining stable memory utilization and preserving the functionality of core network services. Falco demonstrates a lower resource footprint, making it suitable for resource-constrained deployments. In contrast, NeuVector provides broader security capabilities, including network policy enforcement and vulnerability scanning, but at the cost of higher resource consumption. These findings highlight the trade-offs between lightweight and comprehensive container security approaches for practical 5G deployments.**

Keywords-5G; Node-Agent; NFV; falco; neuvector; security.

I. Introduction

As Fifth-Generation (5G) networks evolve toward highly virtualized and software-defined architectures, ensuring robust security mechanisms has become a critical requirement. The adoption of cloud-native technologies and Network Function Virtualization (NFV) enables flexibility and scalability but also expands the attack surface of core network functions [1]. Traditional perimeter-based security approaches are incapable of addressing these dynamic and distributed environments, encouraging the need for runtime security mechanisms that can operate natively within containerized infrastructures [1]. Container-native security solutions represent a promising approach for protecting 5G network functions without compromising performance or scalability [2]. These solutions integrate directly with container orchestration platforms and deploy dedicated monitoring agents capable of real-time threat detection, policy enforcement, and behavioral analysis. By decoupling security logic from the primary Virtual Network Function (VNF), such tools enable modular and scalable protection aligned with cloud-native design principles. Falco [3] and NeuVector [4] represent two complementary approaches to container security. Falco is a lightweight runtime security tool that monitors kernel system calls using extended Berkeley Packet Filter (eBPF) probes and evaluates them against predefined rule sets to detect suspicious container behavior such as abnormal process execution, unexpected file access, or unauthorized Kubernetes API interactions. In contrast, NeuVector provides a broader container security platform that combines runtime process monitoring with network traffic inspection, vulnerability scanning, and policy enforcement mechanisms. While Falco focuses primarily on detecting anomalous behavior and generating alerts, NeuVector supports active response capabilities, including policy-based traffic blocking and container isolation. Despite the growing adoption of container security tools in cloud environments, their performance impact and suitability for 5G core networks remain insufficiently explored. In particular, understanding the trade-offs between runtime monitoring and full-stack security platforms is essential for practical 5G deployments.

This paper addresses this gap by presenting a comparative performance evaluation of Falco and NeuVector in a cloud-native 5G testbed. We analyze their effectiveness in detecting security events and assess their impact on system resources, including CPU usage, memory consumption, and scalability under varying workloads. The results provide practical insights into the trade-offs between lightweight and comprehensive container security approaches, offering guidance for selecting suitable runtime security solutions for containerized 5G network functions.

The remainder of the paper is organized as follows: In Section II, we review related work and identify security challenges in 5G NFV environments. Section III presents the experimental setup and methodology used for performance evaluation. In Section IV, we provide a detailed analysis of the results, including performance and security effectiveness comparisons between Falco and NeuVector. Section V offers a comparative analysis of the two security solutions in the context of 5G Core networks. Finally, Section VI concludes the paper and discusses potential directions for future research.

II. Related Work and Security Challenges in 5G NFVs Environments

Several studies have highlighted that the adoption of NFV in 5G networks significantly improves flexibility and scalability, while simultaneously introducing new security concerns [5]. Existing literature identifies multiple challenges that arise from the software-based and dynamic nature of virtualized network functions, motivating the need for runtime and container-native security mechanisms. First, the increased attack surface

 ISBN: 978-1-68558-442-9

introduced by virtualization means that each NFV represents a potential entry point for attackers. The propagation of software-based endpoints complicates security enforcement in large-scale deployments [6]. Moreover, vulnerabilities stemming from misconfigurations or unpatched software further expose NFVs to exploitation [7].

Second, the dynamic lifecycle of VNFs challenges traditional static security mechanisms. The ability to rapidly instantiate, migrate, and terminate NFVs requires adaptive security solutions capable of responding to real-time changes in network topology and configuration[8].

Third, multi-tenancy introduces additional risks when multiple tenants share the same physical and virtual infrastructure. Unsuitable isolation may allow an attack originating from one tenant to propagate across others[9]. These risks are often exacerbated by configuration errors in resource allocation and access control policies[10].

Furthermore, the creation of complex service chains, a defining characteristic of NFVs, increases the difficulty of security management. Service function chaining interconnects multiple VNFs, meaning that the compromise of a single component can impact the entire end-to-end service [11]. Prior work explores secure orchestration and automated verification techniques to mitigate such threats [12].

Finally, the NFV orchestrator represents a critical and high-value target. As the central control entity responsible for managing VNFs, an orchestrator compromise can lead to large-scale service disruption or loss of network control[13]. Consequently, strong authentication, access control, and continuous monitoring mechanisms are essential to protect this component [7]. These challenges identified in prior work motivate the evaluation of container-native runtime security solutions, such as Falco and NeuVector, which are examined in the following sections.

III. Testbed and Experimental Methodology

This section describes the experimental testbed, deployment configuration of the security tools, and the methodology used to evaluate their performance and detection effectiveness.

A. 5G Cloud-Native Testbed Description

We evaluate the performance impact of container-native runtime security on a cloud-native 5G/6G testbed by comparing Falco and NeuVector in identical conditions. The evaluation focuses on (i) computational overhead, (ii) memory footprint, (iii) system load, and (iv) runtime security monitoring effectiveness, both under normal operation and under simulated attack activity. The testbed consists of a three-node Kubernetes cluster deployed as virtual machines: one control-plane node (8 vCPUs, 8 GB RAM) and two worker nodes (6 vCPUs, 8 GB and 16 GB RAM). All nodes run Ubuntu 24.04.3 LTS (kernel 6.8.0-78-generic) with containerd 1.7.27 as the container runtime.

B. Security Tool Deployment Model

In this testbed, Falco and NeuVector were deployed using node-level monitoring components implemented as Kubernetes DaemonSets, allowing each node to run a dedicated security monitoring agent. The two platforms differ in the runtime signals they analyze. Falco operates primarily at the kernel level by monitoring container system calls through rule-based detection policies, while NeuVector analyzes multiple signals including container processes, inter-container network traffic, and policy compliance events.

C. Measurement Methodology

We collect CPU and memory utilization per node and relevant pods, system load averages (1/5/15 min), and CPU idle percentage. Measurements are reported for three stages: *baseline* (no runtime security), *post-deployment* (Falco or NeuVector enabled), and *attack simulation* (event stream generation).

To account for variability in Kubernetes scheduling and resource allocation, a double-baseline measurement strategy was adopted, where baseline resource utilization was recorded both before NeuVector installation and after NeuVector removal but prior to Falco deployment. Observed baseline values after NeuVector removal were consistent with the initial baseline measurements, indicating minimal residual configuration impact, following repeated-measurement guidance for dynamic containerized environments [14].

D. Event Generator Configuration

To evaluate the detection capabilities of Falco and NeuVector, an event generator was used to simulate attack scenarios and security-relevant activities within the Kubernetes environment. The generator produced predefined attack scenarios corresponding to container runtime and network behaviors that trigger Falco rules and NeuVector security policies. Each scenario represents a *unique simulated attack*, defined as a deterministic invocation of a specific attack rule within the generator configuration. Repeated executions of the same scenario were treated as independent attack instances. During the experiment, 400 attack scenarios were generated and logged with timestamps and identifiers. Falco detections were mapped directly to predefined Falco rules (e.g., *Contact K8S API Server From Container*), while NeuVector detections were triggered through runtime behavioral monitoring and network policy enforcement mechanisms. Ground-truth labels were established by correlating event generator logs with Falco and NeuVector alerts based on timestamp, node identifier, and event type. This correlation enabled the calculation of detection metrics including True Positives (TP), False Positives (FP), and False Negatives (FN). To ensure reproducibility, the experiments were conducted on a three-node Kubernetes cluster (Ubuntu 24.04.3, containerd 1.7.27), with Falco and NeuVector deployed via Helm charts in DaemonSet mode. The event generator was executed in loop mode for a 10-minute evaluation window, and logs were collected using `kubectl logs` for post-processing and metric computation.

IV. Experimental Results

This section presents the experimental results obtained from the testbed, including baseline measurements and the

performance impact of NeuVector and Falco during monitoring and attack simulation scenarios.

A. Baseline Measurements

Before deploying runtime security, baseline resource usage was recorded across cluster nodes. CPU utilization was low on all nodes (control plane: 2% / 204 mCPU[1]; Worker Node 1: 6% / 414 mCPU; Worker Node 2: 7% / 440 mCPU), with RAM usage of 1913 MiB (24%), 1824 MiB (23%), and 3124 MiB (19%) respectively. Load averages remained below 1 on all nodes, indicating sufficient capacity prior to enabling security monitoring. The low baseline utilization (2–7% CPU) reflects the controlled experimental setup, where sufficient resources were provisioned to isolate the overhead of runtime security monitoring rather than emulate fully loaded production deployments.

B. NeuVector Performance Under Monitoring and Attack

After deploying NeuVector, CPU utilization increased across nodes: Control Plane Node from 2% (204 mCPU) to 3% (314 mCPU), Worker Node 1 from 6% (414 mCPU) to 8% (535 mCPU), and Worker Node 2 from 7% (440 mCPU) to 11% (705 mCPU). Memory usage increased from 1913 MiB to 2816 MiB on the Control Plane Node, from 1824 MiB to 2532 MiB on Worker Node 1, and from 3124 MiB to 7559 MiB on Worker Node 2. Load averages showed moderate increases but remained within operational limits. These changes reflect the overhead of NeuVector's distributed monitoring and policy enforcement components.

Attack simulation under event-generation workloads, CPU usage peaked at 5% (399 mCPU) for the Control Plane Node, 11% (688 mCPU) for Worker Node 1, and 21% (1279 mCPU) for Worker Node 2, while memory usage increased to 3417 MiB (43%), 3168 MiB (40%), and 9507 MiB (59%) respectively. System load increased substantially, indicating a higher number of queued processes during high event rates. Table I summarizes system performance metrics under NeuVector monitoring during baseline, post-deployment, and attack simulation phases.

NeuVector's detection performance was evaluated using the event generator described in Section III-D. The evaluation focused on standard security effectiveness metrics, including true positive rate (TPR), false positive rate (FPR), false negative rate (FNR), and overall detection capability. During a 10-minute experiment, the event generator produced 400 unique simulated attack events. NeuVector detected and correctly handled 387 of these events, resulting in an overall detection rate of 96.7%. As reported in Table II, presents the security detection metrics obtained for NeuVector a TPR was 85.9% (344 events), while the FNR was 3.3% (13 missed events). In parallel, NeuVector generated alert logs corresponding to both malicious activities and benign events, yielding a FPR of 10.7% (43 events).

Detected events were classified into multiple response categories. The majority of events (85.9%) were categorized as *Deny Actions*, representing critical threats that were actively blocked, while 10.7% were classified as *Alert Actions* with warning severity. NeuVector demonstrated rapid response behavior, with detection and mitigation actions executed in under one second for all analyzed events.

[1]A millicore is a unit of CPU resource allocation in containerized environments, where 1000 m equals one full CPU core.

TABLE I. SYSTEM PERFORMANCE AT BASELINE, BEFORE, AND AFTER ATTACK SIMULATIONS UNDER NEUVECTOR PROTECTION.

Metric	Baseline	Post-Deployment	Attack Simulation
Control Plane Node			
CPU Util.	2% (204 mCPU)	3% (314 mCPU)	5% (399 mCPU)
Memory	1913M (24%)	2816M (35%)	3417M (43%)
1-min	0.09	0.26	0.45
5-min	0.15	0.28	0.69
15-min	0.24	0.32	1.68
Worker Node 1			
CPU Utilization	6% (414 mCPU)	8% (535 mCPU)	11% (688 mCPU)
Memory Usage	1824M (23%)	2532M (32%)	3168M (40%)
1-min	0.29	0.44	1.32
5-min	0.34	0.52	2.41
15-min	0.49	0.61	5.98
Worker Node 2			
CPU Utilization	7% (440 mCPU)	11% (705 mCPU)	21% (1279 mCPU)
Memory Usage	3124M (19%)	7559M (47%)	9507M (59%)
1-min	0.54	0.73	2.20
5-min	0.69	0.82	4.56
15-min	0.77	0.88	11.89
NeuVector Components			
Controller pods	–	154m	320m
Manager pod	–	8m	19m
Enforcer pods	–	234m	602m
Scanner pods	–	3m	270m

TABLE II. NEUVECTOR SECURITY PERFORMANCE METRICS.

Metric	Value	Event Calculation
False Positive Rate	10.7%	43/400
False Negative Rate	3.3%	13/400
True Positive Rate	85.9%	344/400
Overall Detection Rate	96.7%	387/400
Response Time	$<$ 1 second	Real-time

C. Falco Performance Under Monitoring and Attack

To evaluate Falco under comparable conditions, NeuVector was removed and a new baseline was recorded prior to Falco deployment. Baseline CPU usage was 2% (215 mCPU) on the Control Plane Node, 5% (311 mCPU) on Worker Node 1, and 5% (324 mCPU) on Worker Node 2, with RAM usage of 2270 MiB (28%), 2221 MiB (28%), and 2721 MiB (17%) respectively. Load averages remained below 1 across the cluster.

Post-deployment after enabling Falco, control-plane CPU rose to 4% (338 mCPU), while worker nodes increased to 6% (371 mCPU) and 6% (372 mCPU). Memory usage increased slightly to 2338 MiB (29%) on the Control Plane Node, 2332 MiB (29%) on Worker Node 1, and 2830 MiB (17%) on Worker Node 2. Load averages increased moderately, particularly on Worker Node 2, reflecting the cost of continuous syscall monitoring.

Attack simulation under event-generation workloads, CPU peaked at 6% (438 mCPU) on the Control Plane Node, 8%

TABLE III. SYSTEM PERFORMANCE AT BASELINE, BEFORE, AND AFTER ATTACK SIMULATIONS UNDER FALCO PROTECTION.

Metric	Baseline	Post-Deployment	Attack Simulation
Control Plane Node			
CPU Util.	2% (215 mCPU)	4% (338 mCPU)	6% (438 mCPU)
Memory	2270M (28%)	2338M (29%)	2978M (37%)
1-min	0.20	0.46	0.62
5-min	0.21	0.59	0.89
15-min	0.25	0.69	1.08
Worker Node 1			
CPU Util.	5% (311 mCPU)	6% (371 mCPU)	8% (541 mCPU)
Memory	2221M (28%)	2332M (29%)	3086M (39%)
1-min	0.35	0.35	2.04
5-min	0.35	0.50	2.97
15-min	0.37	0.53	3.32
Worker Node 2			
CPU Util.	5% (324 mCPU)	6% (372 mCPU)	9% (582 mCPU)
Memory	2721M (17%)	2830M (17%)	3738M (41%)
1-min	0.55	0.64	1.33
5-min	0.74	0.97	1.61
15-min	0.80	1.60	2.58
Falco Components			
Falco pod	–	95m	180m
Falco exporter	–	12m	35m

(541 mCPU) on Worker Node 1, and 9% (582 mCPU) on Worker Node 2. Falco pod CPU increased from 26m to 45m on the Control Plane Node, from 40m to 68m on Worker Node 1, and from 47m to 89m on Worker Node 2. Memory peaked at 2978 MiB (37%) on the Control Plane Node, 3086 MiB (39%) on Worker Node 1, and 3738 MiB (41%) on Worker Node 2. System load increased markedly on worker nodes, indicating that sustained event streams amplify monitoring cost. Table III reports the corresponding system performance measurements under Falco protection.

Falco's detection performance was evaluated using the event generator described in Section III-D. During the 10-minute experiment, the event generator produced 400 simulated attack events representing the ground-truth malicious activities. Falco successfully detected 370 of these attacks and missed 30, resulting in a TPR of 92.5% and a FNR of 7.5%. However, Falco generated a total of 2,100 alerts during the same period, including alerts triggered by benign system activities. Among these alerts, 1,700 were classified as false positives, resulting in an operational FPR of approximately 82%, computed as $FPR/(TPR + FPR)$. A significant portion of these false positives originated from the Falco rule *"Contact K8S API Server From Container"*, which was frequently triggered by legitimate Flannel CNI communications with the Kubernetes API server. This observation highlights the sensitivity of rule-based runtime monitoring and the importance of rule tuning when deploying Falco in production environments.

Detected events were classified into multiple severity levels. Critical detections accounted for 11.7% (245 events), warning detections for 6.0% (125 events), and notice-level detections for 9.5% (200 events) representing legitimate activities. The remaining 71.4% (1,500 events) corresponded to false positives. For all detected events, Falco demonstrated rapid response behavior, with detection and alert generation occurring in under one second. Table IV summarizes the corresponding detection metrics for Falco.

TABLE IV. FALCO SECURITY PERFORMANCE METRICS.

Metric	Value	Event Calculation
False Positive Rate	82.1%	1,700/2,100
False Negative Rate	7.5%	30/400
True Positive Rate	92.5%	370/400
Overall Detection Rate	92.5%	370/400
Response Time	< 1 second	Real-time

Analysis of false positives revealed that the primary source of alert noise originated from Flannel CNI components communicating with the Kubernetes API server, triggering the *Contact K8S API Server From Container* rule. These results indicate that Falco provides effective runtime threat detection for containerized 5G core network functions while maintaining low latency and operational efficiency.

V. COMPARATIVE ANALYSIS & SECURITY EFFECTIVENESS COMPARISON

This section presents the experimental results obtained from the testbed, including baseline measurements and the performance impact of NeuVector and Falco during monitoring and attack simulation scenarios.

A. Resource Overhead Comparison

This section presents a comparative evaluation of the NeuVector and Falco security platforms in the context of a 5G Core Kubernetes environment, focusing on resource overhead, system load impact, and security effectiveness. The evaluation quantifies CPU utilization, memory consumption, and system load across all cluster nodes to assess the performance characteristics of each security solution. Both platforms maintained system stability during testing, but each demonstrated distinct performance profiles under load conditions.

Figures 1 and 2 show the resource overhead introduced by the security tools relative to the baseline system state under post-deployment conditions. NeuVector introduces a significantly higher resource overhead than Falco. Specifically, NeuVector components consume 416 mCPU and 5,380 MiB of memory across 10 components, while Falco requires only 113 mCPU cores and 319 MiB of memory across 3 components. NeuVector's resource usage is approximately 3.7 higher in CPU and substantially greater in memory compared to Falco, indicating that NeuVector's comprehensive security coverage comes at the cost of increased resource consumption.

System load analysis reveals differences in computational stress between the two platforms, particularly under attack simulation conditions. Under normal operation, NeuVector caused moderate increases in load, most notably on Worker Node 2, where the 15-min load increased from 0.49 to 0.61. Falco, in contrast, demonstrated a more significant load impact,

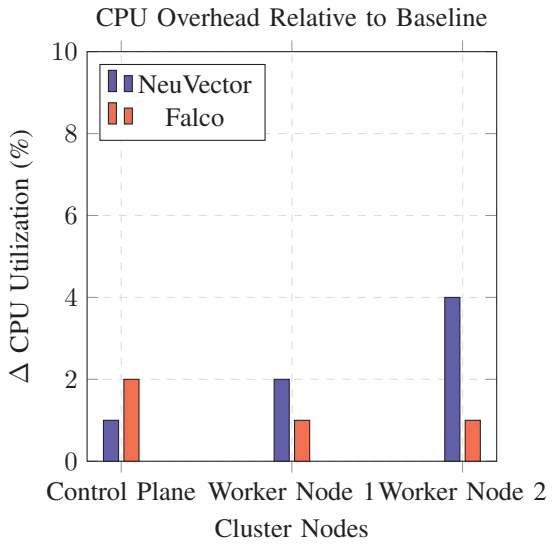


Figure 1. CPU overhead relative to baseline under post-deployment conditions (before attack simulation).

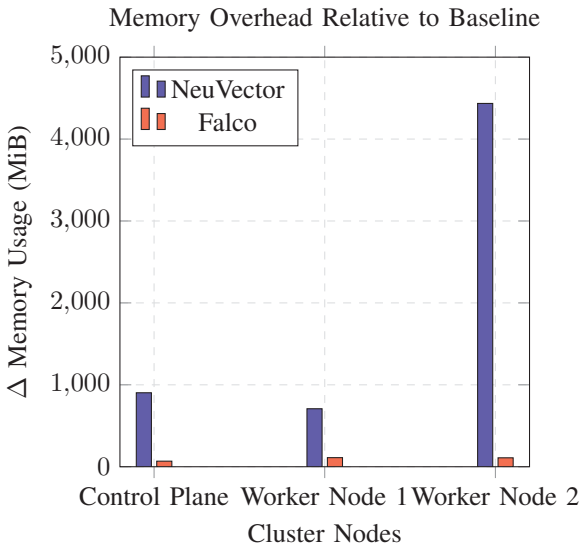


Figure 2. Memory overhead relative to baseline under post-deployment conditions (before attack simulation).

where the 15-min load increased from 0.80 to 1.60 on Worker Node 2.

During attack simulations, the load increases were more significant. NeuVector's system load surged, particularly on Worker Node 2, where the 15-min load rose from 0.88 to 11.89. In comparison, Falco exhibited a increase on Worker Node 1 from 0.53 to 3.32, showing more efficient load management. This indicates that while NeuVector's more comprehensive monitoring incurs significant overhead, Falco's lightweight architecture offers more efficient performance, especially in resource-constrained environments.

Both platforms maintained system stability during the event generator simulations, but with differing impacts on performance. NeuVector resulted in a peak to 21% CPU usage on Worker Node 2 during attack simulations, while Falco only incurred a peak to 9% CPU usage on Worker Node 2 during attack simulations. Memory consumption also differed significantly, with NeuVector consuming significantly more memory due to its broader security capabilities, while Falco demonstrated minimal memory overhead.

B. Scalability Behavior

As the number of simulated subscribers increases, the workload generated by the event generator produces a higher volume of container runtime activities and network events that must be analyzed by the security monitoring tools. Falco processes system call events using lightweight eBPF probes, meaning its monitoring overhead scales with the number of container operations occurring on each node. Consequently, its resource consumption increases approximately in proportion to workload activity. NeuVector, in contrast, performs both container behavior monitoring and network traffic inspection, which introduces additional processing requirements as event volume and traffic intensity grow. As a result, NeuVector generally exhibits higher CPU utilization under increased workloads, although memory consumption remains relatively stable due to its architecture. Overall, the experimental results suggest that both tools exhibit approximately linear growth in overhead with respect to workload intensity, with Falco maintaining a lighter monitoring footprint while NeuVector provides broader security coverage at the cost of higher resource usage.

C. Detection Effectiveness Comparison

Both platforms demonstrated strong threat detection capabilities under controlled testing. Falco achieved a 92.5% TPR, successfully detecting 370 out of 400 simulated attacks. NeuVector had a slightly higher overall detection rate of 96.7%, identifying 387 out of 400 events, but its TPR was 85.9%. While Falco's higher TPR (92.5%) indicates stronger precision in detecting actual threats, NeuVector excelled at broader detection coverage, capturing more security events in total.

The FPR for NeuVector was significantly lower (10.7%) compared to Falco's high FPR of 81%, largely driven by Flannel CNI API communications triggering the "Contact K8S API Server From Container" rule. These results suggest that while NeuVector provides a more precise and effective detection system, Falco's high FPR could hinder its practical deployment in production environments without significant rule customization.

Both tools demonstrated equivalent response times under 1 second, ensuring real-time security protection. However, NeuVector's lower FNR (3.3% vs 7.5%) indicates a more comprehensive coverage of potential threats. NeuVector's low FPR (10.7%) highlights its well-tuned detection rules, while Falco's high FPR (81%) represents a significant operational challenge, which could lead to alert fatigue in production environments. Despite Falco's high TPR, the large number of false positives compromises its usability without further tuning and customization. Additionally, the FNR for Falco

 ISBN: 978-1-68558-442-9

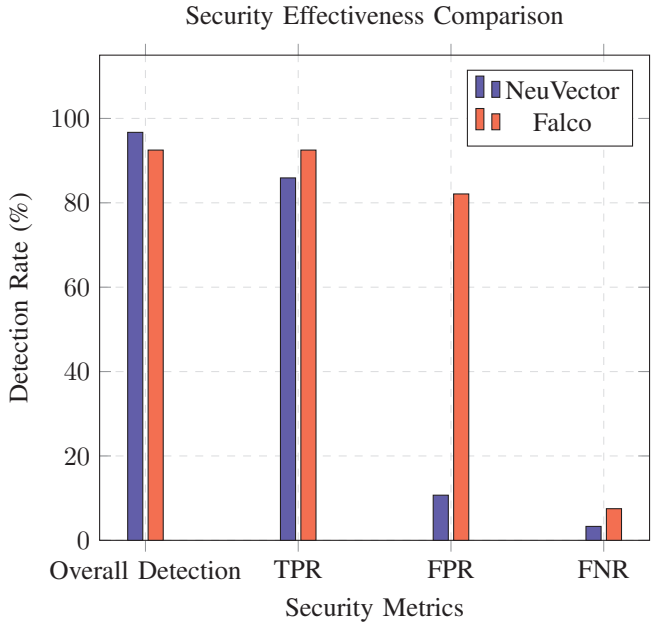


Figure 3. Security Effectiveness Comparison: NeuVector vs Falco.

(7.5%) is relatively higher, suggesting room for improvement in detecting container-specific threats, Figure 3 presents a comparative visualization of the detection effectiveness metrics obtained for NeuVector and Falco.

It is important to note that the reported detection metrics are derived from controlled attack scenarios generated by the event generator operating in loop mode. Consequently, the reported TPR and FPR reflect detection performance for the evaluated synthetic workload rather than the full spectrum of real-world threats. In operational 5G environments, attacks may involve multi-stage exploits, protocol-level manipulation, or previously unseen vulnerabilities that are not represented in the current evaluation scenario.

D. Practical Deployment Implications

Both NeuVector and Falco offer trade-offs between security effectiveness and system performance. NeuVector provides comprehensive security coverage, but with higher resource overhead, while Falco offers efficient runtime security with lower resource consumption but higher false positives.

In terms of 5G Core service resilience, both platforms maintained 99.9%+ service availability and had minimal impact on throughput and system performance. NeuVector introduced a moderate increase in CPU utilization, peaking at 21% CPU usage on Worker Node 2, while Falco showed a more moderate increase at 9% CPU usage on Worker Node 2. Both platforms demonstrated minimal impact on overall system performance, ensuring they meet the stringent requirements for ultra-reliable low-latency communication (URLLC).

In summary, NeuVector offers a broader detection coverage and more precise alerting with lower false negatives but at the expense of higher resource consumption. Falco, on the other hand, provides efficient runtime monitoring with minimal system impact but requires further optimization to address its high FPR. Both platforms meet essential 5G Core performance requirements, but Falco is more suited for resource-constrained environments, while NeuVector is ideal for comprehensive security coverage where performance overhead is less critical.

VI. Conclusion and Future Work

This paper provides a comprehensive evaluation of NeuVector and Falco as container-native security solutions within 5G Core cloud-native environments. The results highlight significant trade-offs between security coverage and performance overhead in cloud-native 5G deployments.

The evaluation revealed that NeuVector consumes 3.7 times more CPU and 16.9 times greater memory compared to Falco. Despite this, NeuVector offered superior security coverage, making it more suitable for production environments requiring comprehensive protection. On the other hand, Falco demonstrated greater resource efficiency, though its high FPR (82.1%) may limit its deployment in production environments without substantial rule customization. igure 4 illustrates the CPU utilization observed across the cluster nodes before and during security monitoring for both NeuVector and Falco.

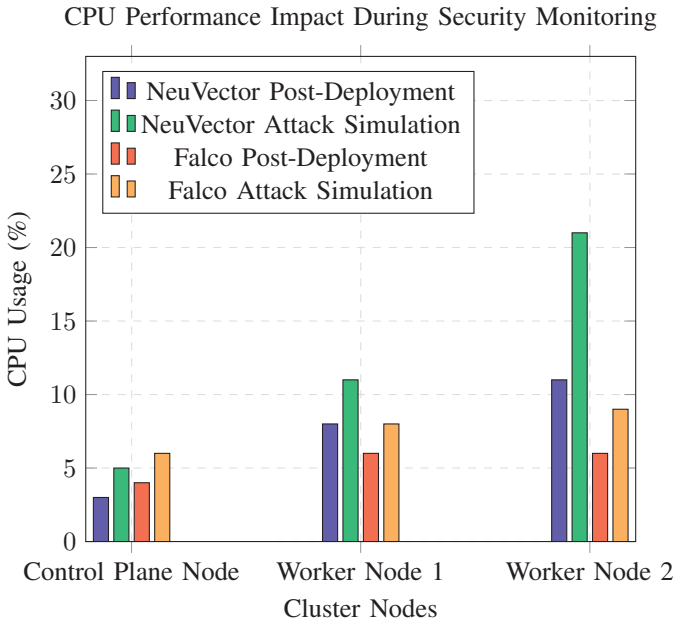


Figure 4. **CPU Performance Impact During Security monitoring: NeuVector vs Falco.**

Under the evaluated workload conditions, both security solutions preserved service continuity and maintained stable operation of the containerized network functions during deployment and attack simulations. These results indicate that container-native security monitoring can be integrated into 5G Core environments without introducing significant service disruption under the tested experimental conditions.

Future research can explore several promising pathways to further enhance the security and performance of container-native security tools in 5G Core networks. First, more realistic

5G-specific attack scenarios will be investigated, including Service-Based Architecture (SBA) exploitation, network slicing isolation bypass attempts, control-plane signaling manipulation, and edge/MEC-related threats. Second, multi-stage attack simulations, behavioral anomaly detection, and machine learning-based false-positive reduction will be explored to improve detection accuracy and operational efficiency, particularly for Falco. Third, the evaluation will be replicated under higher baseline utilization levels representative of production 5G deployments to better characterize the performance–security trade-offs in cost-optimized environments.

In conclusion, this research demonstrates that both NeuVector and Falco can provide effective runtime security monitoring for 5G Core environments under the evaluated workload conditions. While NeuVector offers broader security coverage, it comes with higher resource consumption. Falco provides efficient runtime monitoring with low resource overhead, though its high false-positive rate needs to be addressed before deployment in production environments. Both platforms meet the essential performance requirements of 5G Core services in the tested environment, and the choice between them depends on specific deployment needs, resource constraints, and security priorities.

ACKNOWLEDGMENT

This work has been partially supported by the Horizon European project MARE (grant agreement no. 101191436), by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU (NGEU), partnership on “MUR” (PE000000014 - program “SERICS”).

REFERENCES

[1] U. Kulkarni and S. Fahmy, “Securing the cloud-native 5g control plane”, in *MILCOM 2024 - 2024 IEEE Military Communications Conference (MILCOM)*, 2024. DOI: 10.1109/MILCOM61039.2024.10774032

[2] I. T. Aktolga, E. S. Kuru, Y. Sever, and P. Angin, “Ai-driven container security approaches for 5g and beyond: A survey”, *ITU Journal on Future and Evolving Technologies*, vol. 4, no. 2, 2023.

[3] Falco Security Project, *Falco: Cloud Native Runtime Security*, https://github.com/falcosecurity/falco, Accessed: October 2025.

[4] SUSE Rancher, *NeuVector*, Official documentation describing NeuVector’s features (firewall, process/file-system monitoring, vulnerability scanning) and architecture (Controller, Enforcer, Manager, Scanner).

[5] G. He, X. Liao, and C. Liu, “A security survey of nfv: From causes to practices”, in *2023 3rd International Conference on Consumer Electronics and Computer Engineering (ICCECE)*, 2023. DOI: 10.1109/ICCECE58074.2023.10135454

[6] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, “A comprehensive survey on security in software-defined networking”, *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, 2020.

[7] K. H. Nguyen, N. H. Pham, and J. Shim, “Security challenges in software-defined networking-based 5g networks: A survey”, *IEEE Access*, vol. 9, 2021.

[8] A. Liaqat et al., “Dynamic security in network function virtualization: A survey on methods, challenges, and future directions”, *IEEE Access*, vol. 9, 2021.

[9] M. Ali et al., “Multi-tenancy security in 5g networks: Challenges and countermeasures”, *Journal of Network and Computer Applications*, vol. 173, p. 102 899, 2021.

[10] R. El Ghamri, M. Bouhorma, et al., “Security issues and solutions in multi-tenant environments for network function virtualization”, *Journal of Communications Software and Systems*, vol. 16, no. 3, 2020.

[11] P. Li et al., “Ensuring secure service chains in nfv: State-of-the-art and challenges”, *IEEE Network*, vol. 33, no. 6, 2019.

[12] S. Kim et al., “Secure service chaining in software-defined networking-based 5g environments: A survey”, *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, 2020.

[13] F. Yang et al., *Orchestrator security in nfv/5g*, 2021.

[14] S. Acharekar, A. Goswami, and F. Nair, “Cloud-native performance testing: Strategies for scalability and reliability in modern applications”, *European Journal of Computer Science and Information Technology*, vol. 13, no. 8, 2025.

End-to-End Security of Smart Meter Infrastructure–Based Control Chains: A STRIDE Analysis of Residual Risks Beyond the Smart Meter Gateway

Julian Britz, Sascha Kaven, Felix Scholl, Kolja Eger and Volker Skwarek
Competence Center for Renewable Energies and Energy Efficiency | CyberSec Research and Transfer Centre
University of Applied Sciences Hamburg
Hamburg, Germany
e-mail: {julian.britz|sascha.kaven|felix.scholl|kolja.eger|volker.skwarek}@haw-hamburg.de
Julian Maximilian Behrensen and Milena Zachow
Electrical Engineering and Computer Science
Technical University of Applied Sciences Lübeck
Lübeck, Germany
e-mail: {julian.maximilian.behrensen|milena.zachow}@th-luebeck.de

***Abstract*—In the future, due to the increasing load and generation in energy systems, distribution system operators will have to intervene manually in a network-effective manner. In Germany, the implementation takes place via the legally mandated Smart Meter Gateway architecture, including a large number of process steps and different roles. This results in a multitude of potential cybersecurity risks, some of which have a significant impact on grid stability. This paper therefore examines the risks associated with ad-hoc control in the event of a grid congestion situation. The results can be applied to interventions pursuant to §14a of the German Energy Industry Act in the event of excessive loads and to §9 of the German Renewable Energy Sources Act in the event of excessive generation, as the data flow is identical in both cases. To the best of our knowledge, we therefore provide the first comprehensive threat analysis with a focus on these two legal provisions. We show that various risks prevail, particularly at the endpoints. However, despite the multitude of specifications, the Smart Meter Gateway is not without risks. Across all steps, the use of TLS 1.2 and missing acknowledgments poses a significant security risk. To counteract these, we identified mitigations that need to be taken into account for further legal requirement development. We provide an overview of the German implementation of the Renewable Energy Directive and the measures taken to maintain grid stability. We also highlight the resulting cybersecurity risks and how they can be addressed.**

***Keywords*-Smart Meter Gateway; §14a EnWG; §9 EEG; Load Control; STRIDE Threat Modeling.**

I. Introduction

The German implementation of the Renewable Energy Directive (RED) represents a unique approach. While most countries primarily rely on incentives, Germany enables direct intervention by Distribution System Operators (DSO) [1]. With the accelerating transition to renewable energy sources, power grids face increasing challenges from variable generation and demand patterns. The most prominent is grid congestion, where the available distribution capacity is insufficient to accommodate the current power flows. Therefore, in order to protect grid stability, German DSO have the ability of intervening in home owners systems in an effective manner. Due to the high security requirements, a large number of specifications had to be defined for the implementation of the Smart Meter Gateway (SMGW) architecture. Here, the SMGW serves as the central security anchor and communication hub. End consumers with an annual consumption between 6 000 and 100 000 kWh, as well as systems in accordance with §14a (3) of the German Energy Industry Act (EnWG), are eligible for the installation of SMGW. The mandatory installation quota corresponds to the legal requirements at 20.2% on 30 September 2025, whereas the overall proportion is only 3.8% [2]. In contrast, the proportion of smart meters installed in France reached 90% in 2021 and in the UK more than 60% at the end of 2023 [3][4]. However, it should be noted that their functionality differs fundamentally. While in France and the UK, smart meters are largely responsible for transmitting measured values, in Germany they also act as a central link in the transmission of control signals. This necessitates significantly higher security requirements.

The German energy industry is currently implementing §14a EnWG on the control of flexible loads. According to the legally mandated §14a (3) EnWG flexible loads are defined as electric vehicle charging infrastructure, heat pumps, air conditioning and stationary energy storage systems, which can be summarized as Controllable Local Systems (CLS) [5]. If transmitted measured values indicate a grid congestion situation, §14a EnWG enables DSOs to request temporary load reductions from CLS with an installed power above 4.2 kW. In the interest of fairness, the targeted plants must be selected without discrimination [6].

Furthermore, the transmission of the control signal relies on a complex and highly distributed communication chain spanning across regulated infrastructure, backend systems and customer-owned environments. The backbone of this chain is formed by the SMGW, which is comprehensively secured by the German Federal Office of Information Security (BSI) technical guidelines [7][8]. These guidelines define cryptographic mechanisms, certificate-based authentication and operational processes for secure Wide-Area-Network (WAN), Local-Metrological-Network (LMN), Home-Area-Network (HAN) and Local-Area-Network (LAN) communication. Whereas, the HAN is a subnet of the

 ISBN: 978-1-68558-442-9

LAN containing only the CLS-adapter [9]. Components and responsibilities for these are shown in Figure 1. It should be noted that the role of the External Market Participant (EMT) can also be fulfilled by the supplier or direct marketer. However, an ad-hoc control signal can only be sent by the DSO, as it affects grid stability. Due to the guidelines given, the SMGW and its directly connected communication partners constitute a tightly regulated security domain with a high degree of standardization.

In the WAN, the Gateway Administrator (GWA) is the only role authorized to contact the SMGW without an existing connection. The DSO acts in the role of the EMT. When exclusively receiving data, the passive EMT (pEMT) role applies and when generating and transmitting control signals, the role of the active EMT (aEMT) is used. The specific control commands are transmitted via a proxy mechanisms and executed by customer-side CLS-adapter before it is forwarded either via phase-free relay contacts or digitally to downstream CLS.

The extension of the SMGW centered architecture shifts the effective system boundary into environments that are only partially standardized, heterogeneously implemented and mainly outside the direct control of regulated actors. Customer routers, (cloud-based) Energy Management Systems (EMS) and locally connected devices introduce additional trust boundaries and dependencies not fully addressed by existing technical guidelines. Consequently, there is a recognizable structural asymmetry in the system's security posture. On the one hand, communication segments adjacent to the SMGW are strongly protected through mutual TLS, certificate pinning and strict profile-based authorization, on the other hand, residual risks arise at architectural endpoints. We identified the following predominant risks:

- missing semantic binding of control commands to assets [10],
- availability dependencies on customer-provided infrastructure [11],
- weak security of local device ecosystems behind the CLS-adapter [12],
- information disclosure in backend systems [13],
- Denial of Service (DoS) during SMGW wake up [14],
- the use of TLS version 1.2 [15][16],
- repudiation due to lack of explicit acknowledgments along the §14a EnWG control chain [17]

These weaknesses can undermine both the effectiveness and trustworthiness of §14a EnWG control measures, even in fully compliant implementations.

This paper applies a systematic STRIDE threat modeling approach to an end-to-end ad-hoc control signal transmission process, explicitly including backend systems and the customer domain in scope. Ad-hoc control refers to proxy communication channels that are dynamically established by the SMGW on request, whereas scheduled variants rely on predefined activation times or recurring communication windows [18]. The aim of this paper is not to challenge the robustness of the SMGW security architecture, but to identify and characterize open risks beyond its immediate protection domain.

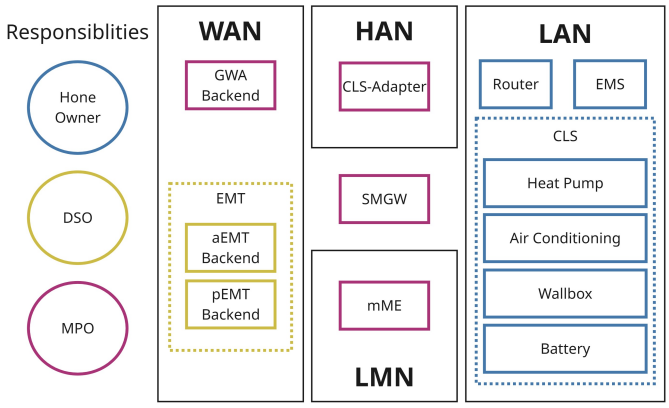


Figure 1. Overview of the SMGW components across the WAN, HAN, LAN and LMN.

In Section II, we present the methodological approach, including definitions, assumptions and limitations. Section III describes the process of transmission of control signals in detail. Based on this, Section IV presents the remaining cybersecurity risks before describing the potential threat scenarios is Section V. Afterwards, necessary mitigations are shown in Section VI. Finally, the results are summarized in Section VII and future work is discussed in Section VIII.

II. Methodology

This paper uses a structured threat modeling approach based on the STRIDE methodology to identify and analyze cybersecurity risks in the §14a EnWG and §9 of the German Renewable Energy Sources Act (EEG) control chain using OWASP Threat Dragon [19]. STRIDE was proposed by Microsoft and represents a bridge for six different types of security threats [20]:

- *Spoofing*: authentication weaknesses,
- *Tampering*: integrity violations,
- *Repudiation*: missing non-repudiation mechanisms,
- *Information Disclosure*: confidentiality violations,
- *Denial of Service*: availability constraints,
- *Elevation of Privilege*: privilege escalation.

It was selected due to its systematic and attacker-oriented classification of threats, making it well suited for complex and distributed systems with multiple trust boundaries and heterogeneous actors [21].

A. Scope Definition

The modeled architecture extends across the control signal transmission chain in accordance with §14a, starting from the backend systems of an aEMT, traversing the gateway administration and communication infrastructure and extending into the customer domain where control actions are executed. In the WAN, the SMGW typically communicates via LTE, CDMA 450 MHz or Broadband Powerline communication (BPL)

[22]. Customer-side components and cloud-based services are deliberately included within the scope.
The model incorporates:

- Backend systems of actors in the WAN
  - GWA, commonly fulfilled by the Metering Point Operator (MPO)
  - EMT, in case of ad-hoc control fulfilled by the DSO
- SMGW with its WAN, LMN and HAN interfaces,
- Metering infrastructure in the LMN with its SMGW interface,
- CLS-adapter with its SMGW and LAN interfaces,
- Proxy-based communication channel between WAN and HAN participants,
- Customer routers, (cloud-based) EMS and CLS.

Supply chain attacks are not within the scope of this analysis. This exclusion applies to vulnerabilities through compromised firmware, third-party software or manufacturer-operated backend platforms. It is nevertheless important to mention that recent findings underline the relevance of such risks, e.g., critical security weaknesses in inverter systems and associated monitoring platforms found by Bitdefender in 2024 [23].

B. Trust Assumptions and Communication Modes

It is assumed that the specifications defined in the BSI TR-03109 series are implemented, meaning that communication segments directly connected to the SMGW rely on mutual TLS using X.509 certificates issued by the Smart Meter Public Key Infrastructure (SM-PKI), certificate pinning and profile-based peer authentication. This includes the WAN-side roles of the GWA and the EMT. For the CLS-adapter and the modern Metering Equipment (mME), a direct trust model is applied, as defined for CLS communication in the BSI TR-03109-5 series [9]. In this, trust anchors are installed prior to first use through organizational processes with self-signed certificates for subsequent TCP/TLS communication.

C. Threat Identification Using STRIDE

Threats were identified and classified according to the six STRIDE categories. Next, they were divided into those that are effectively mitigated by existing technical guidelines and those that remain open due to missing or incomplete safeguards. Threat severity was assessed using the Common Vulnerability Scoring System (CVSS) score to support prioritization. Nevertheless, reliance on numerical risk scores is avoided. The emphasis is on identifying structurally open attack vectors and assessing their potential impact on overall system operation.

D. Analysis Focus and Limitations

The analysis focuses on risks that affect the correctness, availability and accountability of §14a EnWG and §9 EEG control actions. Although our findings do not imply that existing standards are insufficient, they highlight that strong transport-layer and identity security at standardized interfaces do not automatically translate into end-to-end operational security. Given the critical role of grid stability in maintaining reliable power supply and the potential consequences of compromised control action, which range from localized blackouts to cascading failures across regional networks, understanding these residual risks is essential for DSOs and regulators. Therefore, our findings need to be interpreted as complementary to compliance-focused security assessments and as a basis for targeted improvements at system endpoints and organizational boundaries.

III. Ad-Hoc Control Signal Transmission Process

In this section, we describe the data flow in the ad-hoc control signal transmission process step by step. The individual steps are enumerated below and illustrated in Figure 2. The process is based on the metering concept defined by the Association for Electrical, Electronic & Information Technologies (VDE) Forum Network Technology/Network Operation (FNN) and does not provide separate measurement of the CLS [24].

1. First, electrical measurement values are captured at the mME and transmitted to the SMGW via the LMN interface [8]. Communication is based on a communication profile configured and implemented by the GWA. Tariff Use Cases (TAF) determine which data are transmitted and at what times. In case of §14a EnWG, the tariff use case 10 (TAF10) is decisive, involving grid relevant data, such as active and reactive power [25]. The SMGW acts as the primary communication station, actively querying or receiving measurement data. Communication between mME and SMGW is secured using mechanisms defined in BSI TR-03109-1, including authenticated and integrity-protected transport protocols, whereby the SMGW acts as the central trust anchor and only accepts measurement data from administratively configured devices.
2. After collection by the SMGW, the prepared measurement data are transmitted to authorized external systems, either directly to the EMT or via a GWA acting as an intermediary when regulatory requirements mandate encryption. In both cases, transport security is ensured through mutually authenticated TLS connections, with the SMGW acting as the initiating client.
3. Data are received and evaluated by the pEMT to assess the current grid state. Based on this and/or further measurements, the EMT derives control decisions.
4. If the request of a load reduction is necessary, the aEMT sends a request via the web API of the GWA to initiate a CLS communication channel at the SMGW between the aEMT and the CLS-adapter. This channel is established specifically for control communication and is distinct from the measurement data paths.
5. Afterwards, the GWA sends an unencrypted UDP wake-up message to the SMGW requesting the establishment of a TLS connection between them.
6. If certain conditions are met, such as the absence of a current connection, the SMGW establishes a management connection to the GWA. By means of this connection, the GWA requests the establishment of a TLS proxy communication channel between the CLS-adapter and the aEMT. Proxy communication profiles, configured by the GWA beforehand,

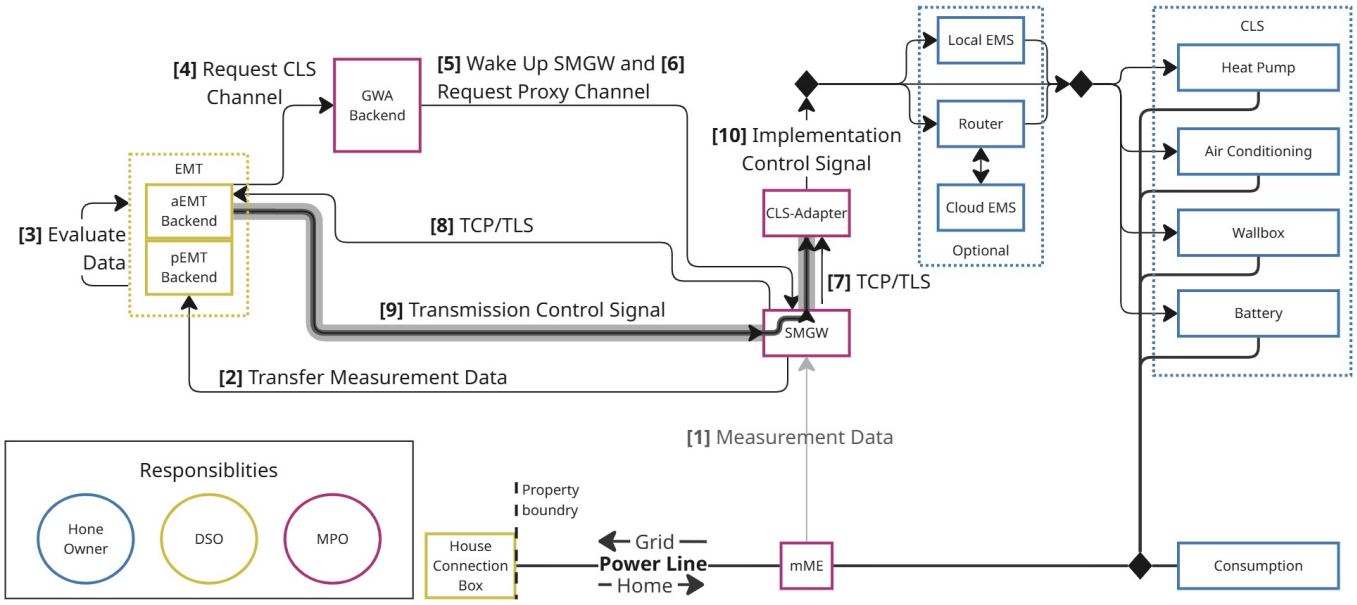


Figure 2. Ad-hoc control process according to §14a EnWG.

define which aEMT is permitted to communicate with which CLS-adapter.

7. Based on the provided conditions, the SMGW opens a TCP/TLS connection to a CLS-adapter in the HAN. This connection forms the HAN-side leg of the end-to-end TLS proxy channel between the CLS and the aEMT. It minimizes coupling between standardized metering functions and application-specific control protocols, but also shifts responsibility for semantic correctness to the communicating endpoints.
8. Subsequently, this is also done on the WAN with the aEMT as the endpoint. Together with HAN-side connection the SMGW forms a transparent, end-to-end proxy channel, without imposing interoperability or security requirements on the application protocol carried inside the tunnel.
9. After this, the SMGW relays application data between the aEMT and the CLS-adapter and acts solely as a transport proxy [18][26]. The application protocol inside the tunnel can conform to IEC 61850, EEBUS (CLS.EEDI) or OpenADR [9].
10. Control signals received by the CLS-adapter are forwarded via ethernet to a EMS or directly to the controllable device [27]. If an EMS is used, it can either be processed locally within a customer-side EMS device or in a cloud-based EMS backend [28]. In the latter case, the signal traverses the customers router and the public Internet to the EMS backend, where it is processed and subsequently sent to the CLS. This communication resides outside the regulated SMGW security domain and marks the responsibility boundary between the MPO and the CLS operator [28][29].

IV. Open Cybersecurity Risks

The SMGW and its standardized communication interfaces are effectively protected by regulatory requirements. Still, a significant number of relevant cybersecurity risks remain open at the architectural endpoints. These open risks concentrate primarily at two locations:

- Backend systems operating in the role of an aEMT and
- Customer domain downstream of the CLS-adapter

Our threat analysis leaks a variety of risks in the ad-hoc control signal transmission process. In total, 108 threats were found, whereby 24 remain without sufficient mitigation. The overall results of the STRIDE analysis are shown in Figure 3 and the threat model and threat report are publicly available at [30].

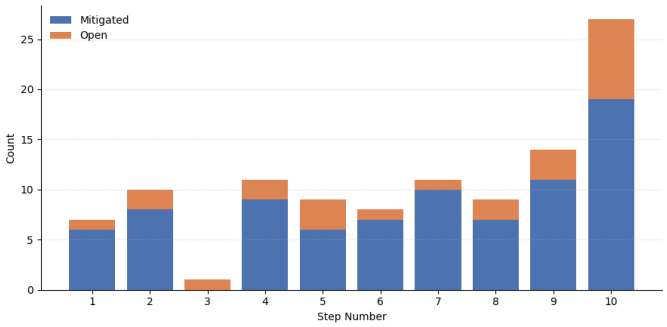


Figure 3. Open and mitigated threats along the process chain.

The identified risks can be grouped into general Cyber-Physical System (CPS) risks that inherently arise whenever control functionality depends on customer-operated infrastructure and structurally novel risks that specifically result from the SMGW architecture.

A. General CPS Risks

1) Availability Dependencies on Customer-Provided Infrastructure: While communication between the SMGW and

backend systems is, with the exception of signal strength and the resulting accessibility issues, designed to be robust, the execution of control commands depends on the availability of local networks, customer routers and in some deployment models (cloud-based) EMS. Through potential disruptions of these components, a systematic availability risk arises, as it would impede signals from reaching the CLS. This is particularly interesting in light of the fact that it cannot be mitigated through cryptographic or protocol-level security measures.

2) Security Gaps Behind the CLS-Adapter: Hardware and Software residing in front of the meter (directly connected to the SMGW) adhere strictly to requirements. For example, the GWA needs to be ISO27001 certified and in case of an external cloud, a risk-analysis is mandatory [31]. The situation is different for EMS and behind-the-meter CLS (connected to the CLS-adapter without direct connection to the SMGW). There are no requirements regarding the use of cloud providers or standardized protocols, whereby the VDE FNN recommends the use of the KNX- and EEBUS communication standards for the transmission of control signals [32]. Still, compromised local devices or insufficiently isolated network segments allow attackers to inject unauthorized control commands, escalate privileges or interfere with legitimate control actions [33].

B. SMGW Architecture Specific Risks

1) Endpoint Identity and Channel Binding Ambiguities: First, risk arises from the semantic binding between control commands, communication channels and controlled assets. Although the proxy-based TCP/TLS connections between the SMGW, the aEMT and the CLS-adapter are strongly authenticated at the transport layer, the association at the aEMT side of an incoming proxy connection with a specific CLS is not enforced by standardized mechanisms. If channel differentiation relies solely on loosely coupled identifiers, control messages may be misattributed or routed to unintended recipients. As a result, correct channel-to-asset binding becomes a shared responsibility between the aEMT and the CLS operator.

2) Information Disclosure at Backend and Endpoint Systems: In addition to attacks with direct effect on the control signals, sensitive operational data, such as grid states and measurement data may be exposed through logging, monitoring interfaces or insufficient access controls at backend systems and customer-side components.

3) SMGW waking-up risks: As the wake-up message, send by the GWA and received by the SMGW, uses UDP, attackers may observe, record and replay structurally valid messages. An attacker may be able to reconstruct valid-looking messages and repeatedly transmit them to the SMGW. Although the consequences of transmitting a valid message are minor and the number of valid messages per minute is bound to 10, there are no requirements limiting the number of invalid messages. While replayed messages can result in a denial of service by triggering the rate limit, invalid messages can lead to a denial of service through resource exhaustion.

4) Mandatory TLS 1.2 Usage: A notable concern arises from the mandated use of TLS version 1.2 for securing communications within all steps [8]. While TLS 1.2 provides robust authentication and encryption, recent research, including algorithm substitution attacks on cryptographic protocols (ASAP) [15] has demonstrated that it remains susceptible to sophisticated subversion attacks. In such algorithm substitution attacks (ASA), adversaries can covertly replace cryptographic implementations with subverted versions that leak secret information embedded within protocol messages. The ASAP paper described how TLS 1.2, unlike TLS 1.3, does not mandate mechanisms to cryptographically bind or attest to the algorithms in use throughout the handshake and data exchange phases. As a result, one can embed maliciously altered random number generators, signature schemes or encryption primitives, which allow the attacker to decrypt the communication, inject false data or otherwise undermine the confidentiality and integrity without raising traditional alarms or detection mechanisms. In addition, TLS 1.3 streamlines the handshake process, which reduces latency and enhances connection setup speed by reducing round trips. It also removed support for vulnerable and outdated cryptographic algorithms, such as AES-CBC, in favor of elliptic curve Diffie-Hellman (ECDHE) and authenticated encryption with associated data (AEAD). The handshake message message is also encrypted and forward secrecy is enhanced, ensuring long-term key compromises do not expose past communication sessions [34].

5) Repudiation Risks Due to Missing End-to-End Acknowledgments: Furthermore, repudiation is likely to happen due to missing acknowledgments at any stage. While certain process steps provide indirect confirmation (e.g., the establishment of a management or proxy connection, TCP), we found that no component explicitly acknowledges successful processing or execution of the requested action. In particular:

- GWA does not explicitly acknowledge aEMT requests to initiate CLS communication
- SMGW does not confirm the receipt of wake-up messages
- Neither EMS nor CLS confirm the successful execution of control commands

The absence of explicit acknowledgments prevents reliable verification of whether a requested load reduction was actually implemented. As CLS are not required to have a separate mME, it is also not possible to determine from the counter values whether the control signal has been carried out or not. This offers the possibility for home owners to actively prevent the realization of control signals.

The VDE FNN is already aware of this problem, which is why it has published a paper entitled "Control with verification in the SMGW" that describes implementation with confirmations [35]. However, this publication is not binding. In general, the control section is completely remodeled. The control signal, including the value to which it has to be reduced, is sent directly from the aEMT to the GWA. The proxy channel is not used in this case. However, sending the power value in the first message accelerates the execution of the control action, reducing opportunities for intermediate verification steps or

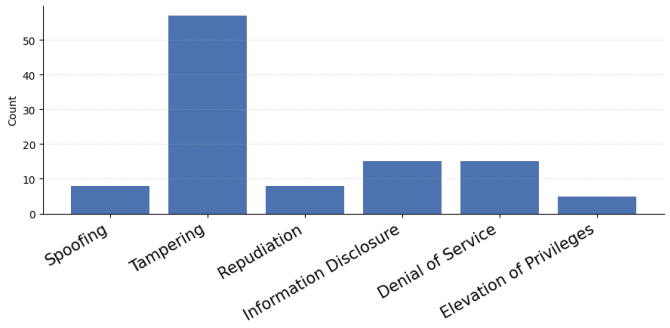


Figure 4. Distribution of identified threat types.

anomaly detection mechanisms that occur during the multi-step communication sequence. Furthermore, this approach does not provide a solution for ensuring the final implementation of the CLS.

C. Representative Threat Scenarios

The following examples illustrate what these risks can look like in practice.

TABLE I. REPRESENTATIVE ATTACKER PROFILES AND OPERATIONAL IMPACT

Scenario	Attacker Profile	Impact
Customer-domain disruption	External network adversary	Command not executed
CLS downstream compromise	Compromised local device or EMS	Unauthorized control injection
Channel misbinding (aEMT)	Compromised backend actor	Unintended load reduction
Backend information leakage	Insider or external backend attacker	Exposure of grid or measurement data
wake-up replay / flooding	External network attacker	Failure to execute control signals
TLS 1.2 subversion	Advanced cryptographic adversary	Confidentiality/integrity compromise
Missing acknowledgments	Home owner	Unverifiable control execution

Table I shows that attacker profiles and motives vary. To minimize risks, appropriate mitigations are presented in the following.

D. Summary of Risk Distribution

While major threats appear on the WAN- and LAN-side endpoints, the BSI decided to require UDP for waking up the SMGW and therefore reduce connection overhead, without rate limiting for valid and invalid messages resulting in a DoS risk. The EMT is responsible for preventing TCP/TLS connection misrouting, while homeowners must ensure secure IT operations within their LAN. The usage of TLS version 1.2 offers attackers additional vectors. Lastly, the repudiation due to missing acknowledgments in all steps remains an unresolved problem with potentially high risks. Although the majority of identified threats, as shown in Figure 4 are tampering cases, most of these risks are already mitigated, reducing their overall impact on the systems security posture.

V. RELEVANT MITIGATIONS

In this section, we summarize the main technical and organizational measures that can reduce the identified residual risks along the ad-hoc control signal transmission chain.

A. Baseline Protections at Standardized Interfaces

Throughout the set of mandatory security mechanisms defined by the BSI TR-03109 series, segments that are directly connected to the SMGW benefit from a high level of security guarantees. These include:

- Mutual TLS authentication with certificate pinning,
- Profile-based authorization of communication partners,
- Cryptographic integrity protection of measurement data,
- Enforced termination of connections if certificate validation fails

The threat model confirms that these measures are effective in mitigating classical network-level attacks in close proximity to the SMGW. However, this does not imply complete end-to-end security. To gain a higher level of safety, further measures need to be implemented, such as acknowledgment messages and rate limiting for valid and invalid messages.

B. Resilience Measures for Customer Domain Availability Dependencies

To avoid risks arising from customer-owned infrastructure, further specifications regarding power and network-related assets are needed. These include an explicit definition of degraded-mode behaviors for loss of connectivity. Concrete fallback scenarios shall therefore be specified and implemented in the assets. In this context, [35] presented a proposal on how to use predefined power envelopes in the event of a communication failure. Furthermore, basic cybersecurity protective measures, such as rate limiting, should be mandatory to avoid cascading retry storms. Since it is infeasible to make household networks inaccessible to attackers, grid-connected devices with a high impact on the stability, like CLS, must implement measures to counter attacks. Regarding the EMT, it is essential to detect suspicious behavior of the systems at an early stage, to have sufficient time to take action. Thus, indisputable confirmation of the realization of the required measure is recommended.

C. Hardening and Segmentation Behind the CLS-Adapter

While the role of the GWA must be ISO27001 certified and a risk analysis must be carried out for the use of external cloud providers, there are no requirements for the use of EMS systems. As previously demonstrated, this area of application in particular poses significant safety risks. At the very least, every EMS cloud provider must have a cloud security concept in place.

D. Strong Channel-to-Asset Binding at aEMT Endpoints

As it is absolutely important for control signals to reach the intended CLS-adapter, errors are not permitted at this point. Therefore, it is crucial that the EMT binds incoming proxy communication channels to a unique combination of

SMGW and CLS-adapter identifier to know exactly which asset will be addressed. This binding needs to be stored safely and out of reach of unauthorized people. In case of ambiguity, the message should be discarded. By enforcing semantic consistency between transport-level connections and application-level control targets, the risk of misdirected or unintended control actions is reduced. It is advisable to publish an implementation guideline on how the data can be stored and processed at EMT side in order to avoid mix-ups.

E. Endpoint Data Protection and Secure Logging Practices

Endpoints, WAN- as well as LAN-side, should complement secure transport mechanisms with disciplined data handling practices to reduce information disclosure risks. This is about minimizing sensitive data in logs and protecting logs through appropriate access controls and integrity mechanisms. It is crucial to note that confidentiality guarantees extend beyond the communication channel and are maintained throughout the data lifecycle.

F. Mitigation of Risks Related to the SMGW Wake-Up Mechanism

The requirements for contacting the SMGW are strict and well defined. Nevertheless, the scope of checks that need to be done by the SMGW to recognize a message as valid or not is comprehensive. To avoid resource exhaustion, we recommend applying early and lightweight filtering of incoming wake-up messages before initiating costly cryptographic validation steps. While TCP/TLS would prevent passive observation and replay of wake-up messages, it would introduce stateful connection handling, which is undesirable for a lightweight wake-up mechanism. Therefore, we rather recommend complementing rate limiting for valid wake-up requests by prioritization mechanisms that ensure legitimate wake-up messages from authorized sources are not starved by excessive invalid or replayed traffic. One simple solution would be extending the rate limit to invalid messages and reducing valid time windows.

G. Mitigation of Risks Related to TLS Version Support

As seen in previous sections, the use of TLS version 1.2 does have security vulnerabilities. This highlights the need to migrate to TLS 1.3, which structurally mitigates the attacks, as shown by [15]. The use of TLS 1.2 should therefore be prohibited in the SMGW environment and at least all parties involved in the SM-PKI should be required to use TLS 1.3. A protocol version fallback to older TLS versions should never be accepted, even if version negotiation fails during the handshake. Therefore, security testing should explicitly verify that downgrade attempts are rejected and that fallback to older TLS versions is technically impossible.

H. Mitigation of Repudiation Risks Through Explicit Acknowledgments

A key problem that extends across all process steps is the missing of explicit acknowledgments. For those involved, it is easy to deny receiving messages and their incorrect implementation. To mitigate these repudiation risks, the §14a EnWG and §9 EEG control process should incorporate explicit acknowledgments to confirm receipt and execution of requested actions.

- Channel establishment requests initiated by the aEMT should be acknowledged by the GWA
- Receipt of wake-up messages by the SMGW should be acknowledged, independent of subsequent management or proxy connection establishment
- Control commands forwarded to the CLS should include a mechanism for the CLS to acknowledge successful receipt and execution, optionally distinguishing between acceptance and actual implementation
- CLS-adapter should provide execution feedback, enabling correlation between issued commands and observed asset behavior

I. Summary

To increase the safety of the §14a EnWG ad-hoc control signal transmission chain, especially endpoint responsibility strengthening is required. While the SMGW centered security architecture provides a robust foundation, end-to-end security can only be achieved if backend systems and customer domain components implement complementary safeguards. While there are some risks that arise at certain points, the security vulnerabilities in the use of TLS 1.2 and the existing deniability also pose risks that extend across the entire process chain. To this end, existing specifications must be revised and expanded to include specifications regarding endpoints.

VI. Conclusion and Future Work

In this paper, we provide to the best of our knowledge the first threat analysis with a focus on §14a EnWG and §9 EEG that is made openly available for DSOs, regulators and researchers. Whereas the ad-hoc control signal transmission chain demonstrates a high level of security within the SMGW domain, it does not achieve end-to-end security when the endpoints are included. The STRIDE analysis shows that risks predominantly arise at architectural endpoints, where no general specifications prevail.

The absence of explicit end-to-end acknowledgments creates significant repudiation risks, limiting the enforcement of grid-serving control actions. To solve this problem, the entire process chain, including the assets, must be able to prove tamper-proof that the control signals have been implemented. The mandatory use of TLS 1.2 represents a systemic weakness, as it remains susceptible to algorithm substitution attacks. It underscores the necessity of a migration to TLS 1.3. Customer-side infrastructures and (cloud-based) EMS introduce additional availability and security dependencies. Here, clearly defined fallback scenarios must be implemented.

Beyond these, the control signal transmission chain is exposed to general supply chain attacks, including compromised firmware, manipulated software updates and subverted third-party components at both backend and device levels.

It should also be noted that encryption algorithms currently in use (elliptic curve and RSA) are not secure in the post-quantum era. Due to the long service life of critical infrastructure, it is therefore important to take early action.

In summary, strengthening endpoint responsibilities, introducing verifiable control execution with proof-of-action mechanisms and extending regulatory requirements beyond the SMGW are necessary to achieve end-to-end security.

The next step would be to validate the identified risks through empirical testing and operational experience. Therefore, a testbed that maps the entire control chain would have to be implemented in order to realistically simulate the cases. A hardware-in-the-loop test environment is ideal for demonstrating the effects on the grid. Co-simulation is required to link the grid simulation with the network simulation. In addition, with the appropriate test environment, risks could be discovered that remain undetected in theory and it could also be used as an exploit demonstrator to show weaknesses and its impact.

This setup can also be used to investigate the handling and risks of the VDE FNN implementation note on the subject of control with verification [35]. In this context, it would be interesting to compare these approaches with the one presented in this paper in order to identify any advantages and disadvantages of the respective models and possibly implement an even better approach.

Future research could address the question of how the process of drafting legislation could be optimized. In the case of the SMGW architecture, it has become apparent that the back-and-forth between legislators and industry leads to delays. Approaches, such as central testing stations, where effects can be tested early in a practical setting, could be helpful in quickly finding suitable solutions.

In addition, there are still considerable uncertainties regarding the controllability of supply chain attacks and the handling of post-quantum cryptography in the current process chain and must be investigated by future research.

Acknowledgment

This work was carried out in the context of the research project SimCyberGrid, which is funded by the German Federal Ministry for Research, Technology and Space (code: 13FH637KA2) and the research project FARFALLE, which is funded by the German Federal Ministry for Economic Affairs and Climate Action (code: 03EI6136C).

References

[1] J. Hawran and P. Suchomski, "International comparison of smart meter solutions and functionalities in Europe", 2026, Accessed: Mar. 13, 2026. [Online]. Available: https://www.ffe.de/veroeffentlichungen/internationaler-vergleich-von-smart-meter-loesungen-und-funktionali-taeten-in-europa/.

[2] Bundesnetzagentur, "Rollout of smart metering systems: Quarterly surveys", 2025, Accessed: Mar. 13, 2026. [Online]. Available: https://www.bundesnetzagentur.de/DE/Fachthemen/ElektrizitaetundGas/NetzzugangMesswesen/Mess-undZaehlwesen/iMSys/artikel.html.

[3] Forschungsstelle für Energiewirtschaft e.V., "Technical framework conditions for demand-side flexibility", 2025, Accessed: Mar. 13, 2026. [Online]. Available: openaccess.ffe.de/wp-content/uploads/2025/05/Final_Technical-Framework-Conditions-for-Demand-Side-Flexibility.pdf.

[4] UK Department for Energy Security and Net Zero, "Smart Meters Consultation: Government Response to the Mid-Point Review – Analytical Annex", 2023, Accessed: Mar. 13, 2026. [Online]. Available: https://assets.publishing.service.gov.uk/media/64a66ec1c531eb001364ff02/smart-meters-consultation-government-response-mid-point-review-analytical-annex.pdf.

[5] *Energy Industry Act*, Federal Republic of Germany, last amended 2025, 2005.

[6] Bundesnetzagentur, "BK6-22-300", 2023, Accessed: Mar. 13, 2026. [Online]. Available: https://www.bundesnetzagentur.de/DE/Beschlusskammern/1_GZ/BK6-GZ/2022/BK6-22-300/Beschluss/BK6-22-300_Beschluss_20231127.pdf.

[7] Bundesamt für Sicherheit in der Informationstechnik, "Protection Profile for a Smart Meter Gateway (SMGW-PP), Version 2.0", Tech. Rep. BSI-CC-PP-0073-V2, 2024.

[8] Bundesamt für Sicherheit in der Informationstechnik, "Requirements for the Interoperability of the Smart Meter PKI, Version 2.0", Tech. Rep. TR-03109-1, 2024.

[9] Bundesamt für Sicherheit in der Informationstechnik, "Communication adapter", Tech. Rep. TR-03109-5, 2023.

[10] C. Freudenmann et al., "Open and Secure: Amending the Security of the BSI Smart Metering Infrastructure to Smart Home Applications via the Smart Meter Gateway", in *Smart Energy Research. At the Crossroads of Engineering, Economics and Computer Science*, C. Derksen and C. Weber, Eds., Cham: Springer International Publishing, 2017, pp. 136–146, ISBN: 978-3-319-66553-5. DOI: 10.1007/978-3-319-66553-5_10.

[11] S. Tanimoto et al., "Risk Assessment of Home Gateway/Smart Meter in Smart Grid Service", in *2016 5th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)*, 2016, pp. 1126–1131. DOI: 10.1109/IIAI-AAI.2016.25.

[12] M. Orlando et al., "A Smart Meter Infrastructure for Smart Grid IoT Applications", *IEEE Internet of Things Journal*, vol. 9, no. 14, pp. 12 529–12 541, 2022, ISSN: 2327-4662, 2372-2541. DOI: 10.1109/JIOT.2021.3137596.

[13] Z. Zhang et al., "Achieving Privacy-Friendly Storage and Secure Statistics for Smart Meter Data on Outsourced Clouds", *IEEE Transactions on Cloud Computing*, vol. 7, no. 3, pp. 638–649, 2019, ISSN: 2168-7161. DOI: 10.1109/TCC.2017.2685583.

[14] H. Kumar, P. Nnaji, and S. Kumar, "Smart Meter Performance Under Wired and Wireless Cyber Security Attack", in *2024 IEEE World AI IoT Congress (AIIoT)*, 2024, pp. 0061–0067. DOI: 10.1109/AIIoT61789.2024.10578962.

[15] S. Berndt et al., "ASAP: Algorithm Substitution Attacks on Cryptographic Protocols", in *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, Accessed: Feb. 07, 2025, Association for Computing Machinery (ACM), 2022, pp. 712–726. DOI: 10.1145/3488932.3517387.

[16] S. B. Alemu, "The transformation of TLSfrom version 1.2 to 1.3Efficiency vs Security vs Interoperability", 2020, Accessed: Mar. 13, 2026. [Online]. Available: https://ethz.ch/content/dam/ethz/special-interest/infk/inst-infsec/appliedcrypto/education/theses/Bachelor's-Thesis-SamuelBedassaAlemu.pdf.

[17] K. Förderer, M. Lösch, R. Növer, M. Ronczka, and H. Schmeck, "Smart Meter Gateways: Options for a BSI-Compliant Integration of Energy Management Systems", *Applied Sciences*, vol. 9, no. 8, p. 1634, 2019, ISSN: 2076-3417. DOI: 10.3390/app9081634.

[18] Bundesamt für Sicherheit in der Informationstechnik, "Appendix 6 Operational processes involving the SMGW, Version 2.0", 2024, Accessed: Mar. 13, 2026. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/

Publikationen/TechnischeRichtlinien/TR03109/TR-03109-1_Anlage_Betriebsprozesse_v2_0.pdf.

[19] OWASP Foundation, "OWASP Threat Dragon Documentation", 2025, Accessed: Mar. 13, 2026. [Online]. Available: https://www.threatdragon.com/docs/.

[20] L. Kohnfelder and P. Garg, "The STRIDE Threat Model", 1999, Accessed: Mar. 13, 2026. [Online]. Available: learn.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20).

[21] R. Khan, K. McLaughlin, D. Laverty, and S. Sezer, "Stride-based threat modeling for cyber-physical systems", in *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, 2017, pp. 1–6. DOI: 10.1109/ISGTEurope.2017.8260283.

[22] VDE Thüringen e.V., "Smart meter gateways in rollout", 2019, Accessed: Mar. 13, 2026. [Online]. Available: https://www.vde-thueringen.de/resource/blob/1929426/3746f3868f025db258e69b42e6540e0e/16-ppc-kohlsdorf-smart-meter-gateways-im-rollout-data.pdf.

[23] J. Siemer, "Bitdefender reports on the risk of hacker attacks on deye inverters and the solarman monitoring platform", pv magazine Deutschland, 2024, Accessed: Mar. 13, 2026. [Online]. Available: https://www.pv-magazine.de/2024/08/08/bitdefender-berichtet-ueber-gefahr-von-hacker-angriffen-auf-deye-wechselrichter-und-solarman-monitoring-plattform/.

[24] VDE Verband der Elektrotechnik, Elektronik und Informationstechnik e. V. – Forum Netztechnik/Netzbetrieb, "Connection and operation of generation plants, storage facilities and consumption devices on the low-voltage grid", 2025, Accessed: Mar. 10, 2026. [Online]. Available: https://www.vde.com/resource/blob/2434724/02833b45ff85c9b9beb3a37073755444/vde-fnn-hinweis-anschluss-betrieb-niederspannung-data.pdf.

[25] Bundesamt für Sicherheit in der Informationstechnik, "Errata for BSI TR-03109-1 V1.0.1 – TAF9 and TAF10", Tech. Rep. Errata für die BSI TR 03109-1 V1.0.1 – TAF 9 und 10, 2019.

[26] T. Riedel and M. Berg, "Guide to regulatory requirements for external market participants (emt)", 2024, Accessed: Mar. 13, 2026. [Online]. Available: https://www.dena.de/fileadmin/dena/Publikationen/PDFs/2024/Leitfaden_Regulatorische_Vorgaben_fuer_externe_Marktteilnehmer.pdf.

[27] VDE Verband der Elektrotechnik, Elektronik und Informationstechnik e. V. – Forum Netztechnik/Netzbetrieb, "Characteristics of the digital interface on controllable devices or on an energy management system", 2024, Accessed: Mar. 13, 2026. [Online]. Available: https://www.vde.com/resource/blob/2292786/5d38accc5ab02ad04df7cab8a64f1f63/impuls--digitale-schnittstelle-data.pdf.

[28] VDE Verband der Elektrotechnik, Elektronik und Informationstechnik e. V. – Forum Netztechnik/Netzbetrieb, "Cybersecurity in the use of cloud-based energy management systems", 2025, Accessed: Mar. 13, 2026. [Online]. Available: https://www.vde.com/resource/blob/2381556/7acb1ddff231b8d83432cf76c920a92e/vde-fnn-hinweis-schnittstellen-steuerungseinrichtung-data.pdf.

[29] Bundesverband der Energie- und Wasserwirtschaft, "Recommendations for connecting and operating controllable consumer devices until technical standards are available", 2025, Accessed: Mar. 13, 2026. [Online]. Available: https://www.bdew.de/media/documents/20250808_BDEW_AWH_Empfehlungen_Anschluss_steuerbare_Verbrauchseinrichtungen.pdf.

[30] J. Britz, J. M. Behrensen, and S. Kaven, "OWASP Threat Model", 2026, Accessed: Mar. 13, 2026. [Online]. Available: https://github.com/SimCyberGrid/STRIDE_14a_EnWG_Control_Chain.

[31] Bundesamt für Sicherheit in der Informationstechnik, "Smart Meter Gateway administration", Tech. Rep. TR-03109-6, 2025.

[32] VDE Verband der Elektrotechnik, Elektronik und Informationstechnik e. V. – Forum Netztechnik/Netzbetrieb, Ed., *Control box specifications: Functional and design features*, 1.5. Berlin, Germany, 2025, p. 223, E-Book/PDF; Bestellnummer 636511.

[33] S. Lakshminarayana et al., *Cybersecurity threats to power grid operations from the demand-side response ecosystem*, Feb. 2, 2025. DOI: 10.48550/arXiv.2310.18820. arXiv: 2310.18820[cs].

[34] A. Ali and V. P. Singh, "Comparative Analysis of Transport Layer Security (TLS) Versions", *International Journal for Research in Applied Science and Engineering Technology*, vol. 11, no. 12, pp. 680–684, 2023, ISSN: 23219653. DOI: 10.22214/ijraset.2023.57430.

[35] Bundesamt für Sicherheit in der Informationstechnik, "Control with verification in the smart meter gateway", Tech. Rep. TR-03109-1 - implementation note, 2025.

From Network Traffic to Data Space: Design, Validation, and Multi-Model Benchmarking

Julian Graf*, Murad Hachani*, Christoph Moser*, Sebastian Fischer*, Rudolf Hackenberg*
*Department of Computer Science and Mathematics
University of Applied Sciences Regensburg
Regensburg, Germany
Email: {julian.graf, christoph.moser, rudolf.hackenberg, sebastian.fischer}@oth-regensburg.de;
murad.hachani@st.oth-regensburg.de

***Abstract*—Encrypted communications and the increasing diversity of cyber-threats challenge traditional Intrusion Detection Systems (IDSs). This paper evaluates a novel network traffic analysis model for intrusion detection. The model offers a prioritized packet processing approach for resource-constrained environments and near real-time analysis of encrypted network traffic metadata. The model defines Primary Target Group (PTG) characteristics to process and structure Internet Protocol (IP) network packets. The corresponding model architecture reflects a metadata driven data space. This is used to derive structural and topology related features. The created Polymetric Queueing Topology Space (PQTS) data space is then used for Machine Learning (ML) driven anomaly detection and attack classification algorithms. The approach is evaluated on TON_IoT and CIC-IDS2017 benchmark data sets using tree-based ML models, particularly Random Forest and LightGBM, for binary and multiclass classification. Experimental results demonstrate strong detection performance while employing a lightweight feature set for prioritization, that is consistent across diverse IP-based networks. These results demonstrate that ML algorithms trained on PQTS data space provides an effective foundation for intrusion detection within heterogeneous network environments.**

Keywords-Network Intrusion Detection; Machine Learning; Feature Engineering; Network Security; Encrypted Network Traffic Classification.

I. Introduction

With the increasing complexity of modern computer networks and the rapid growth in both the number and sophistication of cyber-threats, ensuring network security has become a critical challenge. Traditional signature-based Intrusion Detection Systems (IDSs) remain widely deployed due to their effectiveness in detecting known attack patterns. However, an inherent reliance on predefined signatures limits their ability to identify novel and previously unseen attacks, including zero-day exploits and emerging threats, such as Large Language Model (LLM)-generated malware [1]. Consequently, anomaly-based intrusion detection has gained increasing attention as it focuses on identifying deviations from normal network behavior rather than relying on known attack signature [2]. In parallel, critical infrastructures, enterprise networks, and other specialized domains, such as the automotive industry, have become prime targets for cyber-attacks, thereby increasing the demand for reliable and timely threat detection mechanisms. The large volumes of network traffic generated in such environments require IDSs that are computationally efficient and capable of operating in near real time. In this paper, we evaluate Machine Learning (ML) algorithms based on the subset Primary Target Group (PTG) of the PQTS data space. The model-derived data space is designed to address these challenges by providing the necessary information for prioritized intrusion detection in Internet Protocol (IP)-based network environments.

The structure of the paper is as follows: Section II reviews related work. Section III describes the data sets used for evaluation. Section IV details the system architecture. Section V describes the experimental results. Finally, Section VI summarizes the findings and outlines directions for future work.

II. Related Work

Recent research on network intrusion detection in Internet of Things (IoT) environments is increasingly driven by the need to operate under strict computational constraints while analyzing predominantly encrypted network traffic. For related work, we categorized existing work into (i) approaches that emphasize resource-efficient Network Intrusion Detection System (NIDS) designs for encrypted traffic and (ii) studies that evaluate and benchmark ML algorithms for IDSs under such constraints. The following discussion reviews related work along these two categories.

A substantial body of research addresses the protection of IoT environments through lightweight detection mechanisms, often emphasizing feature reduction and computationally efficient learning algorithms for anomaly detection and traffic classification [3]–[5]. Most of these approaches rely on ML-based models to enable efficient analysis of network traffic on resource constrained devices.

Nguyen et al. propose Realguard, a lightweight deep-learning-based IDS designed for deployment on resource-constrained IoT gateways [6]. Their system combines incremental statistical feature extraction with a compact, fully connected neural network to enable packet-level attack detection at the network edge. While Realguard follows a conventional feature-based NIDS pipeline, our approach adopts a heuristic packet processing model that supports scalable analysis of encrypted IoT traffic through adaptive prioritization mechanisms.

Complementary to lightweight model designs, several studies focus on feature aggregation and temporal abstraction mechanisms. Raskovalov et al. introduce a sliding time window–based and queue-oriented NIDS that relies on flow-level aggregation and compact neural networks [7]. Their approach effectively captures temporal traffic characteristics using a predefined feature set and a queuing mechanism of flows. Our model enables adaptive and scalable analysis of encrypted traffic by selectively prioritizing two layered predefined PTGs. This includes Host, Protocol, and the Universal queues. Additionally, connection flows across the Primary Analysis Cycle (PAC) time intervals are reconstructed. The aggregation of queue-based characteristics combined with prioritization mechanisms and conventional flow-based features expands the existing research by adding multiple layers of information. This facilitates novel and resource-efficient evaluations on aggregated subsets, such as PTG $\subset$ PQTS. Beyond efficiency-oriented detection mechanisms, comparability and reproducibility have emerged as important concerns in ML-based NIDS research. Sarhan et al. address these challenges by proposing a standardized NetFlow-based feature set and benchmarking multiple open data sets using a fixed flow-level representation. While their work facilitates cross-dataset comparability, our approach extends standardized flow-level representations with additional structural and topology-aware information, enabling a more expressive yet lightweight abstraction of IP-based encrypted network traffic [8].

III. Data Set Description

To ensure comparability and validate the proposed model, two publicly available data sets were used: TON_IoT and CIC-IDS2017, which represent different application scenarios. Both data sets are standard benchmarks for the research, development, and evaluation of NIDS and broader cybersecurity analysis. Each contains raw captured network traffic complemented by subsequently labeled attacks. Structural differences and their suitability for this study are discussed in the following section.

A. TON_IoT

The TON_IoT data sets were developed to model realistic Industrial Internet of Things (IIoT) and IoT environments comprising heterogenous sensors, edge and fog systems, and hosts running different versions of Windows and Unix-based operating systems [9]–[16]. It includes network traffic protocols, such as Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and Internet Control Message Protocol (ICMP), as well as common services like Hypertext Transfer Protocol (HTTP), Secure Shell (SSH), Domain Name System (DNS), and File Transfer Protocol (FTP) along with sensor telemetry and operating system logs. The data set shows eight different attack categories, including scanning, Denial of Services (DoS), Distributed Denial of Services (DDoS), ransomware, backdoor, data injection, Man-in-the-Middle (MITM), and password-cracking attacks, all executed against vulnerable services to generate realistic threat behavior. The TON_IoT data set is used to evaluate host- and network-based IDS in IoT environments. Unlike traditional IDS data sets, it emphasizes realistic cyber-physical system behavior, enabling cross-layer intrusion detection.

B. CIC-IDS2017

Another popular data set was published by the Canadian Institute for Cybersecurity (CIC). Sharafaldin et al. focused on generating network traffic that reflects a generic office environment [17]. Hence, they put emphasis on realistic background traffic by emulating naturalistic benign behavior of 25 users based on the HTTP, HTTPS, FTP, SSH, and email protocols, simulating working hours from Monday to Friday. The data set was generated within a fully configured network comprising a modem, firewall, switches, routers, and multiple operating systems, including Windows, Ubuntu, and macOS. Network traffic was collected from 12 machines within the victim network and supplemented with real attacks originating from the attack network, covering both internal LAN communications and Internet traffic. In addition, network traffic data, memory dumps, and system calls from the victim machines were captured during the attacks. More than 80 network flow features were extracted using CICFlowMeter [18]. They recorded 18 different attack classes including different types of Brute Force and DoS, as well as Heartbleed, Web Attacks, Infiltration, Botnet, and DDoS. This data set is widely used for network-based intrusion detection and is often used for supervised and semi-supervised ML. Its increased realism compared to earlier data sets and inclusion of modern attack types make it a standard benchmark in IDS research. With TON_IoT and CIC-IDS2017 we utilize two highly cited benchmark data sets as input for our network analysis model described in the Section IV. The model processes the captured network traffic of TON_IoT and CIC-IDS2017 as pcap format. It utilizes model structure, queue, and topology features to derive the PQTS data space.

IV. Network Analysis Model Description

In a previous study, we presented the architecture of a heuristic packet processing model for network analysis, along with an initial empirical evaluation of its effectiveness [19]. The model derived data space (*Polymetric Queueing Topology Space (PQTS)*) is presented together with tree-based ML approaches. This data space incorporates polymetric features derived from the model architecture to improve ML anomaly detection and classification. This section describes the proposed model, which is to be further evaluated based on predefined PTGs. To create the model data space, the data sets mentioned in Section III were used as input.

Figure 1 shows a simplified architecture image of the model. It presents an illustration of the reconstructed network traffic structure after a complete PAC time interval. After capture in Step 1, the individual packets are sorted into queue-based structures based on device, protocol, and universal PTGs. After the expiration of a fixed PAC time interval, which depends on the network and the communication conditions,

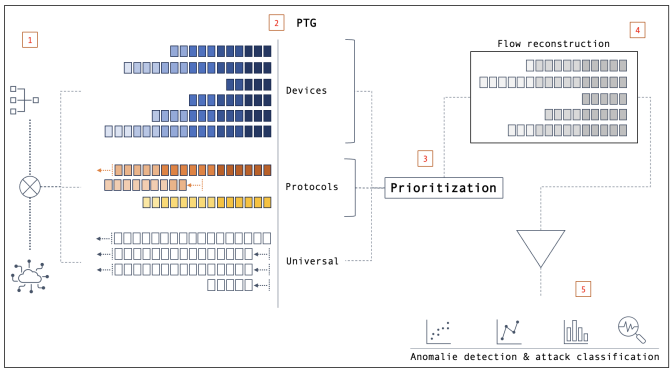


Figure 1. Example model architecture after completed PAC with implicit flow regression.

features are derived from the model structure and stored asPQTS data space. Using these features, a prioritization algorithm calculates, which queue should be marked for further deep attack classification and anomaly detection analysis. The marked queues are then selected and separately analyzed in Step 4. Traffic flows are reconstructed, creating additional features for the PQTS data space. Furthermore, the content of reconstructed flows is analyzed and represented by calculated *Time Series Flow Content Features*. In the final step, the time series feature data and the PQTS model data are merged and made available for further deep ML anomaly detection and intrusion classification methods.

In the following model evaluation, PTG features (Figure 1, designation 1) were used to train the ML models. The configuration parameters are defined with a PAC duration of 30 seconds. For each time interval a classification is performed to determine whether the observed behavior is benign or indicative of an intrusion. In case of multi-class classification, the specific type of attack is additionally identified.

The developed model demonstrator uses network traffic in pcap format as input. The network traffic is replayed and the first part of the model architecture is built as shown in Figure 1 designation 2. The following PTGs are therefore specified:

- **Devices** Including all internal devices.
- **Protocols** Including the captured transport layer protocols used.
- **Universal** Including all network packets in the exact temporal sequence in which they were observed and recorded at the network interface.

For each time interval, 17 different features are calculated. The features used for the ML-models are discussed in more detail in Section V-C.

V. Machine Learning

In this section, a description of the ML-models used for binary and multi-class classification is given. For each of the data sets described in Section III, several models utilizing Random Forest and LightGBM are applied.

A. Preprocessing

This subsection describes the preprocessing steps applied to the output of the model prior to machine learning. The system processes raw network traffic as input. Since features are extracted directly from *pcap*-files or *tshark*, no explicit handling of missing or invalid values is required at the packet level. Traffic is segmented into fixed-length windows of 30 seconds, from which statistical flow-based features are extracted and aggregated. These feature vectors form the input to the subsequent machine-learning models.

Each PAC is assigned a label according to the ground truth information provided by the respective data set. As the data sets differ in structure, data set specific labeling strategies are employed. For the TON_IoT data set, labels are derived from the directory structure provided by the authors, where traffic is organized by attack category and benign activity. For the CIC-IDS2017 data set, labels are assigned based on the attack start and end timestamps specified in the flow reconstructed *csv*-files. A window is labeled as malicious if any packet timestamp overlaps with an attack interval and benign otherwise. No overlapping attack types occur in either data set.

Prior to model training, all numerical features are normalized using z-scores. Finally, the data set is split into training and validation (70/30) subsets, which are used for model selection, hyperparameter tuning, and performance evaluation, respectively.

B. Model Selection

Tree-based learning algorithms are particularly suitable in this context because they can capture non-linear relationships and handle heterogeneous feature distributions, which are typical for network-traffic, while remaining computationally efficient and requiring minimal preprocessing [20]. Random Forest and gradient-boosted decision tree models have been widely adopted in intrusion detection due to their robustness on imbalanced data sets, low inference latency, and limited preprocessing requirements [21]. In addition, their inherent support for feature importance estimation enables transparent analysis of how PQTS derived features contribute to detection performance. Based on these considerations, Random Forest and LightGBM are selected as representative tree-based models for experimental evaluation. Random Forest serves as a robust ensemble baseline with strong generalization properties, while LightGBM represents a gradient-boosted variant optimized for efficiency and scalability. Evaluating both models on the same PQTS feature representation enables a comparative analysis of different tree-based learning paradigms that are not influenced by fundamentally different model classes.

C. Feature Selection

Feature selection is guided by the structure of the PTG subset of PQTS abstraction rather than by explicit feature elimination. All experiments use a fixed set of 17 queue-derived features capturing variability, growth dynamics, and packet processing behavior (Table I), ensuring that observed

performance differences originate from the learning models rather than feature availability.

TABLE I. PQTS FEATURES USED FOR ML TRAINING.

Index	Feature Name
1	gq_mean_protocol_queue_length
2	gq_protocol_queue_length_variance
3	gq_protocol_queue_length_entropy
4	gq_median_protocol_queue_length
5	gq_mean_host_queue_length
6	gq_median_host_queue_length
7	gq_host_queue_length_variance
8	gq_host_queue_length_entropy
9	gq_mean_queue_length
10	gq_median_queue_length
11	gq_popped_pkts_iteration
12	gq_popped_pkts_cumulative
13	gq_protocol_queue_count
14	gq_protocol_queue_count_layer_4_below
15	gq_protocol_queue_count_diff
16	gq_mean_growth_rate
17	gq_median_growth_rate

Feature importance analysis from Random Forest (Table II) shows a consistent ranking across data sets and classification tasks. With respect to the particular PTG queue classification type, entropy- and variance-based queue-features contribute most strongly to detection performance.

TABLE II. BEST HYPERPARAMETERS ON PQTS FOR BINARY AND MULTICLASS CLASSIFICATION.

Set	Model	Best Features	Scores
		Binary Classification	
TON	RF	gq_host_queue_length_entropy	0.2814
		gq_protocol_queue_length_variance	0.1198
		gq_median_host_queue_length	0.1170
CIC	RF	gq_popped_pkts_cumulative	0.2699
		gq_protocol_queue_length_variance	0.0846
		gq_host_queue_length_entropy	0.0844
		Multiclass Classification	
TON	RF	gq_host_queue_length_entropy	0.2352
		gq_median_host_queue_length	0.1133
		gq_protocol_queue_length_variance	0.0869
TON	LGBM	gq_host_queue_length_entropy	0.2779
		gq_popped_pkts_cumulative	0.2330
		gq_mean_growth_rate	0.2259
CIC	RF	gq_popped_pkts_cumulative	0.2970
		gq_host_queue_length_entropy	0.0740
		gq_protocol_queue_length_variance	0.0716
CIC	LGBM	gq_popped_pkts_cumulative	0.7153
		gq_host_queue_length_entropy	0.1406
		gq_protocol_queue_length_variance	0.1045

Overall, the results demonstrate that a compact, model derived PTG feature set is sufficient for effective intrusion detection across data sets without data set specific feature engineering.

D. Model Training and Evaluation

All models are trained on the PQTS subset PTG shown in Table I. For TON_IoT and CIC-IDS2017, we use the same data set split for both binary and multiclass experiments, with data set statistics reported in Table III.

TABLE III. DATA SET OVERVIEW FOR PTG-BASED EXPERIMENTS.

Data Set	Classes	Features	Train	Test
TON_IoT	5	17	11,749	2,938
CIC-IDS2017	18	17	3,873	969

Hyperparameter optimization is performed using grid search with 5-fold cross-validation for all evaluated models. The search spaces applied for the multiclass and binary classification of the corresponding algorithms are shown in IV and V. For Random Forest classifiers, optimization focuses on ensemble size, tree depth, and split criteria, while LightGBM tuning additionally considers learning rate and framework related optimization parameters.

TABLE IV. RANDOM FOREST PARAMETER GRID FOR GRIDSEARCHCV.

Hyperparameter	RF
n_estimators	[100, 200]
max_depth	[10, 20, 30, None]
min_samples_split	[2, 5]
min_samples_leaf	[1, 2]
max_features	[sqrt, log2]
bootstrap	[True]

TABLE V. LIGHTGBM PARAMETER GRID FOR GRIDSEARCHCV.

Hyperparameter	LGBM
n_estimators	[100, 200]
max_depth	[10, 20, -1]
learning_rate	[0.05, 0.1]
num_leaves	[31, 63]
min_child_samples	[20, 50]
subsample	[0.8]
colsample_bytree	[0.8]

All reported classification results are obtained using the selected hyperparameter configurations identified during this optimization process and summarized in Tables VI and VII.

Tables VIII and IX summarize the classification performance of the evaluated models on the PTG feature representation across binary and multiclass intrusion detection tasks. Overall, the results demonstrate that tree-based models can effectively adapt the PTG abstraction, achieving high detection performance even on encrypted network traffic.

For binary classification on the TON_IoT data set, the Random Forest model achieves an F1-score of 0.9947 and

 ISBN: 978-1-68558-442-9

TABLE VI. BEST RANDOM FOREST HYPERPARAMETERS ON PTG FOR BINARY AND MULTICLASS CLASSIFICATION.

Feature	TON		CIC	
	Bin	Mul	Bin	Mul
n_estimators	200	100	150	100
max_depth	10	20	None	20
min_samples_split	5	2	5	5
min_samples_leaf	1	1	2	1
max_features	sqrt	sqrt	sqrt	sqrt
bootstrap	True	True	True	True

TABLE VII. BEST LIGHTGBM HYPERPARAMETERS ON PTG FOR BINARY AND MULTICLASS CLASSIFICATION.

Feature	TON	CIC
n_estimators	100	100
max_depth	20	-1
learning_rate	0.1	0.05
num_leaves	63	63
min_child_samples	50	50
subsample	0.8	0.8
colsample_bytree	0.8	0.8

TABLE VIII. BINARY CLASSIFICATION RESULTS OF PTG.

Model	Set	Acc	F1	AUC
RF	TON	0.9966	0.9947	0.9999
RF	CIC	0.8534	0.5439	0.8979

an Area under the ROC curve (AUC) of 0.9999. The balanced accuracy exceed 99%, indicating robust detection capability under moderately imbalanced class distributions. This highlights the suitability of the PTG features for distinguishing benign and malicious traffic without deep packet inspection. In contrast, binary classification on CIC-IDS2017 yields noticeably lower performance, particularly in terms of recall and F1-score. While the model maintains a reasonable AUC of 0.8979, the reduced balanced accuracy reflects the higher class imbalance and increased attack diversity in terms of volume-based characteristics of the data set. These results suggest that the performance degradation is primarily data set driven rather than caused by limitations of the PTG feature representation.

TABLE IX. MULTICLASS CLASSIFICATION RESULTS OF PTG.

Model	Set	BalAcc	$F1_w$	MeanAUC
RF	TON	0.9915	0.9915	0.9998
LGBM	TON	0.9935	0.9936	0.9998
RF	CIC	0.8627	0.8376	0.9753
LGBM	CIC	0.6549	0.8612	0.9866

For multiclass classification on TON_IoT, both Random Forest and LightGBM achieve strong results. The Random Forest model attains a weighted F1-score of 0.9915, while LightGBM slightly improves upon this with a weighted F1-score of 0.9936. The consistently high MeanAUC values across both models indicate stable class separation and balanced performance across attack categories. Multiclass classification on CIC-IDS2017 remains challenging. Nevertheless, the Random Forest model achieves an approximate weighted F1-score of 0.8376, demonstrating that the PQTS-based abstraction retains discriminative power even under more complex multiclass conditions. Overall, the results indicate that the PTG feature space enables intrusion detection across multiple datasets and learning models, particularly for volume-based attacks, such as DoS, while maintaining low computational cost and leveraging novel features derived from queue-structure statistics.

E. Performance of Comparable Approaches

To put these results into perspective, a comparison with other IDSs, introduced in Section II, is provided. It is important to note that the results obtained on a small subset of PQTS are competitive across both datasets. For the TON dataset, the proposed model achieves performance comparable to the related work; however, on CIC it performs less strongly for this reduced PQTS subset. The performance metrics reported in the literature for binary and multiclass classification are summarized in Tables X and XI.

TABLE X. COMPARISON OF BINARY CLASSIFICATION RESULTS

Model	Set	Acc	F1	Features
CL-SKD [5]	CIC	0.9980	0.9980	-
Realguard [6]	CIC	0.9964	-	100
NetFlow [8]	TON	0.9964	1.00	43

TABLE XI. COMPARISON OF MULTICLASS CLASSIFICATION RESULTS

Model	Set	BalAcc	$F1_w$	Features
BT-TPF [4]	CIC	0.9960	0.9960	78
CL-SKD [5]	CIC	0.9984	0.9984	-
Realguard [6]	CIC	0.9993	-	100
BT-TPF [4]	TON	0.9945	0.9944	43
NetFlow [8]	TON	0.9805	0.98	43

Overall, the comparison suggests that the proposed approach is already competitive with established IDSs, particularly on TON, while the weaker results on CIC for this reduced PQTS subset indicate clear potential for improvement for certain attack classes in subsequent full-space evaluations.

VI. CONCLUSION AND FUTURE WORK

This paper presents a multi-model evaluation of a heuristic network analysis model. It abstracts captured packet streams into a complex, structurally motivated, and metadata-driven data space. The PQTS derives from polymetric queue states, model architecture and traffic prioritization mechanisms. The proposed approach was evaluated using the PTG feature subset of the PQTS representation on two widely used and well-established benchmark datasets. This focus allows an assessment of the model's robustness and applicability under realistic and comparable experimental conditions, while addressing key challenges of modern network environments, including high traffic volumes, encryption, and resource constrained environments.

The experimental results demonstrate that a single subset of the PQTS data space enables competitive intrusion detection for volume based attacks across different data sets and classification tasks. Tree-based learning models, such as Random Forest and LightGBM, achieved strong performance on both binary and multiclass scenarios, particularly on the TON_IoT data set. Feature importance further confirm that dynamic queue-related characteristics, such as entropy, variance, and growth behavior, are highly discriminative for both data sets. These findings underline that meaningful security relevant information can be extracted from network traffic without reliance on deep packet inspection or extensive feature engineering.

Before deployment in production environments, targeted validation steps are necessary. These include validation of robustness and generalization capabilities, assessment under realistic traffic loads, and evaluation against previously unseen attack patterns. In summary, the results presented indicate that the PTG subset of the PQTS data space already constitutes a promising foundation for anomaly-based intrusion detection in encrypted and resource-constrained network environments.

REFERENCES

[1] K. Ahi and S. Valizadeh, "Large language models (LLMs) and generative AI in cybersecurity and privacy: A survey of dual-use risks, AI-generated malware, explainability, and defensive strategies," in *2025 Silicon Valley Cybersecurity Conference (SVCC)*, 2025, pp. 1–8. DOI: 10.1109/SVCC65277.2025.11133642.

[2] J. Graf, K. Neubauer, S. Fischer, and R. Hackenberg, "Architecture of an intelligent intrusion detection system for smart home," in *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2020, pp. 1–6. DOI: 10.1109/PerComWorkshops48775.2020.9156168.

[3] S. U. Jan, S. Ahmed, V. Shakhov, and I. Koo, "Toward a lightweight intrusion detection system for the internet of things," *IEEE Access*, vol. 7, pp. 42 450–42 471, 2019. DOI: 10.1109/ACCESS.2019.2907965.

[4] Z. Wang *et al.*, "A lightweight IoT intrusion detection model based on improved BERT-of-Theseus," *Expert Systems with Applications*, vol. 238, p. 122 045, 2024. DOI: 10.1016/j.eswa.2023.122045.

[5] Z. Li and W. Yao, "A two stage lightweight approach for intrusion detection in internet of things," *Expert Systems with Applications*, vol. 257, p. 124 965, 2024. DOI: 10.1016/j.eswa.2024.124965.

[6] X.-H. Nguyen, X.-D. Nguyen, H.-H. Huynh, and K.-H. Le, "Realguard: A lightweight network intrusion detection system for IoT gateways," *Sensors*, vol. 22, no. 2, 2022. DOI: 10.3390/s22020432.

[7] A. Raskovalov, N. Gabdullin, and I. Androsov, *NIDS neural networks using sliding time window data processing with trainable activations and its generalization capability*, 2024. DOI: 10.48550/ARXIV.2410.18658.

[8] M. Sarhan, S. Layeghy, and M. Portmann, "Towards a standard feature set for network intrusion detection system datasets," *Mobile Networks and Applications*, vol. 27, no. 1, pp. 357–370, 2022. DOI: 10.1007/s11036-021-01843-0.

[9] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustainable Cities and Society*, vol. 72, p. 102 994, 2021. DOI: 10.1016/j.scs.2021.102994.

[10] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, and A. Anwar, "TON_iot telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems," *IEEE Access*, vol. 8, 2020. DOI: 10.1109/ACCESS.2020.3022862.

[11] T. M. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa, and F. T. H. D. Hartog, "ToN_IoT: The role of heterogeneity and the need for standardization of features and attack types in IoT network intrusion data sets," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 485–496, 2022. DOI: 10.1109/JIOT.2021.3085194.

[12] N. Moustafa, M. Keshky, E. Debiez, and H. Janicke, "Federated TON_IoT windows datasets for evaluating AI-based security applications," in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Guangzhou, China, 2020, pp. 848–855. DOI: 10.1109/TrustCom50675.2020.00114.

[13] N. Moustafa, M. Ahmed, and S. Ahmed, "Data analytics-enabled intrusion detection: Evaluations of ToN_IoT linux datasets," in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2020, pp. 727–735. DOI: 10.1109/TrustCom50675.2020.00100.

[14] N. Moustafa, *New generations of internet of things datasets for cybersecurity applications based machine learning: TON_IoT datasets*, 2019. DOI: 10.26190/5D7AC9BFE8487.

[15] N. Moustafa, *A systemic IoT-fog-cloud architecture for big-data analytics and cyber security systems: A review of fog computing*, 2019. DOI: 10.48550/ARXIV.1906.01055.

[16] J. Ashraf *et al.*, "IoTBoT-IDS: A novel statistical learning-enabled botnet detection framework for protecting networks of smart cities," *Sustainable Cities and Society*, vol. 72, 2021. DOI: 10.1016/j.scs.2021.103041.

[17] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization:" in *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, 2018, pp. 108–116. DOI: 10.5220/0006639801080116.

[18] A. Habibi Lashkari, G. Draper Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of tor traffic using time based features," in *Proceedings of the 3rd International Conference on Information Systems Security and Privacy*, 2017, pp. 253–262. DOI: 10.5220/0006105602530262.

[19] J. Graf, M. Hachani, S. Fischer, and R. Hackenberg, "A heuristic packet processing model for improved encrypted network analysis," in *Proceedings of the 2025 Cyber Security in CarS Workshop*, 2025, pp. 1–12. DOI: 10.1145/3736130.3764510.

[20] K. S. Adewole, A. Jacobsson, and P. Davidsson, "Intrusion detection framework for internet of things with rule induction for model explanation," *Sensors*, vol. 25, no. 6, 2025. DOI: 10.3390/s25061845.

[21] O. Achbarou, T. Datsi, O. Bourkoukou, and A. M. El Kiram, "Enhanced intrusion detection system using feature selection and hybrid learning models for high performance and efficiency in an IoT environment," *Journal of Engineering Research*, 2025. DOI: 10.1016/j.jer.2025.10.016.

An Agentic GraphRAG Architecture for Organization-Aware Cyber Threat Intelligence

Philipp Fuxen
Department of Computer Science and Mathematics
Ostbayerische Technische Hochschule Regensburg
Regensburg, Germany
e-mail: philipp.fuxen@oth-regensburg.de

Prof. Dr. Rudolf Hackenberg
Department of Computer Science and Mathematics
Ostbayerische Technische Hochschule Regensburg
Regensburg, Germany
e-mail: rudolf.hackenberg@oth-regensburg.de

***Abstract*—Cyber Threat Intelligence (CTI) is a key component of modern security operations, yet existing systems struggle to integrate heterogeneous intelligence sources and to provide contextualized, organization-aware analysis. Knowledge-graph-based approaches offer structured and explainable representations of threats, while Large Language Models (LLMs) enable flexible semantic analysis of unstructured information. However, these paradigms are typically applied in isolation and fail to support continuous, context-driven threat modeling. This paper proposes an Agentic Graph Retrieval-Augmented Generation (GraphRAG) architecture for CTI analysis. The approach integrates structured and unstructured CTI into a persistent knowledge state consisting of a cybersecurity knowledge graph, a document store, and a vector index. Graph-first retrieval is combined with agentic orchestration to iteratively assemble bounded, task-relevant context for LLM-based reasoning, enabling grounded, explainable, and organization-specific threat analysis. We present a layered reference architecture and a detailed methodology describing knowledge construction, graph-first retrieval, agentic analysis workflow, and grounded reasoning. The proposed approach is designed to support security analysts and automated security workflows in operational CTI environments and provides a foundation for adaptive and explainable CTI systems that combine structured knowledge representation with flexible AI-driven analysis.**

Keywords-cyber threat intelligence; knowledge graph; graph retrieval-augmented generation; agentic AI systems; large language models; context-aware threat analysis; explainable AI.

I. Introduction

Cyber Threat Intelligence (CTI) has become a central component of modern security measures, enabling companies to detect, analyze, and respond to cyber threats in a timely manner. However, the increasing volume, speed, and heterogeneity of threat data pose a significant challenge for existing CTI systems. Security-related information is scattered across structured feeds, logs, vulnerability databases, and unstructured text sources, including incident reports, security blogs, and dark web forums. As a result, security analysts are confronted with fragmented information landscapes that hinder the correlation, contextualization, and prioritization of emerging threats. [1]

To address this challenge, recent research has explored knowledge-based representations and Artificial Intelligence (AI) techniques for automating CTI. Cybersecurity Knowledge Graphs (CSKGs) are often used to model relationships between threat actors, vulnerabilities, attack techniques, and assets in a structured and traceable manner [2]. While graph-based approaches enhance reasoning and traceability, they often rely on manual curation or static data ingestion pipelines, which limits their adaptability to rapidly changing threat landscapes [3]. This restricts their ability to reflect evolving attack campaigns, newly observed tactics, and organization-specific threat contexts in time. In parallel, Large Language Models (LLMs) have shown promising results in extracting threat entities, tactics, and techniques from unstructured text sources [4]. Despite their flexibility, LLM-based approaches suffer from limited consistency, a lack of explicit structure, and challenges regarding explainability and validation, which limit their use in operational CTI systems.

Although both paradigms offer complementary strengths, they are typically applied in isolation in existing CTI systems. Graph-based systems provide a persistent and structured knowledge state but lack flexible semantic reasoning over newly observed intelligence, while LLM-based pipelines offer powerful semantic analysis but operate without a stable, verifiable memory. Consequently, current approaches do not support continuous, context-aware threat modeling that combines a structured persistent knowledge state with adaptive semantic reasoning. To our knowledge, the integration of LLM-based reasoning and knowledge-graph-based retrieval as a unified Graph Retrieval-Augmented Generation (GraphRAG) pipeline remains largely unexplored in existing CTI architectures.

In this paper, we propose an Agentic GraphRAG-based CTI architecture that combines LLMs and CSKGs to enable dynamic and organization-specific threat analysis. Heterogeneous structured and unstructured CTI sources are integrated into a persistent CSKG, which is subsequently used as a retrieval layer to provide context for LLM-driven reasoning tasks, such as threat prioritization, contextual analysis, and explanation generation. The main contributions of this work can be summarized as follows:

- An agentic GraphRAG architecture for organization-aware CTI analysis
- A methodology for constructing and maintaining a persistent knowledge state from heterogeneous CTI sources
- A graph-first retrieval and analysis concept enabling grounded and explainable CTI reasoning

The rest of this paper is organized as follows: Section II reviews related work, Section III defines the problem and design

goals, Section IV introduces the system architecture, Section V describes the proposed Agentic GraphRAG methodology, and Section VI concludes the paper and outlines future work.

II. Related work

This section examines related work in the areas of CTI processing, CSKGs, and CTI analysis based on LLMs. We structure the discussion along the pipeline of the proposed GraphRAG architecture to highlight the limitations of current approaches and motivate a unified solution. Recent work has begun to explore hybrid retrieval and reasoning pipelines that combine knowledge graphs and language models, but these approaches are not yet systematically formulated as persistent agentic GraphRAG pipelines for continuous CTI analysis.

A. CTI Feed Integration and Knowledge Graph Construction

CTI systems are typically based on standardized exchange formats, such as Structured Threat Information eXpression (STIX) and Trusted Automated eXchange of Indicator Information (TAXII), which provide interoperable representations and transport mechanisms for cross-platform exchange of CTI objects and collections [5][6]. While these standards are widely used, they primarily address syntactic interoperability and do not inherently provide semantic enrichment or organization-specific contextualization.

To support more comprehensive integration and reasoning across heterogeneous CTI, recent work increasingly uses CSKGs as structured, explainable representations of entities, such as threat actors, malware, vulnerabilities, techniques, and affected assets. Survey papers highlight CSKGs as a holistic approach to merging different cybersecurity data sources and enabling large-scale correlations and reasoning [2][7]. Furthermore, construction-oriented review papers discuss current pipelines for CSKG construction, including extraction, fusion, and inference components, and outline open challenges related to automation and data quality [3]. However, many CSKG construction approaches remain constrained by manual curation, schema engineering, or batch-oriented collection, limiting their ability to continuously integrate rapidly evolving and highly unstructured CTI sources.

B. Graph-based Threat Modelling and Contextualisation

Graph-based threat modeling has a long tradition in cybersecurity, including attack graphs and related graphical security models that support the analysis of multi-stage exploits and the prioritization of remediation measures. A recent survey summarizes the state of the art in the automatic generation and use of attack graphs and related models, highlighting challenges related to scalability, maintenance, and automation in rapidly changing vulnerability and threat landscapes [8]. Beyond classic attack graphs, more comprehensive research on graph mining in cybersecurity shows how graph representations are used for detection, correlation, and security analysis tasks across different data modalities [9].

In the CTI field in particular, work such as HINTI uses graph-based learning to highlight the interrelationships between heterogeneous indicators in order to quantify relevance and improve intelligence mining, thereby demonstrating the advantages of structured graph representations for CTI reasoning [10]. Nevertheless, many graph-based approaches assume that relevant knowledge is already available in structured form. They usually offer only limited mechanisms for the continuous integration of unstructured narrative information (reports, blogs, forum posts) and for the dynamic selection of organization-specific subgraphs as context for downstream conclusions.

C. Large Language Models for CTI Analysis

LLMs have enabled new forms of CTI extraction and analysis from unstructured text. Benchmarking initiatives, such as CTIBench systematically evaluate the performance of LLMs on CTI tasks and highlight both the strengths and limitations of LLMs in the CTI context [4]. Complementary work proposes fully LLM-driven or LLM-assisted approaches for processing CTI reports, with the goal of reducing the manual workload of analysts while leveraging the semantic capabilities of modern models [11]. Despite their flexibility, LLM-only pipelines often lack a persistent, verifiable state. This limits the traceability, reusability, and validation of extracted threat intelligence and increases the risk of hallucinations and inconsistent results, which are important considerations for operational CTI deployment.

D. Hybrid LLM-Knowledge Graph Approaches and GraphRAG

To combine structured representations with semantic extraction, current work uses LLMs to create or enrich CSKGs from CTI text sources. CTINexus proposes optimized contextual learning to support data-efficient CTI extraction and CSKG creation in data-scarce environments [12]. CTIKG uses prompt-based segmentation and multiple agents to create security-oriented knowledge graphs from CTI articles while accounting for the limitations of LLMs, such as hallucinations and context constraints [13]. These approaches demonstrate that LLMs can effectively populate and enrich CSKGs, but they focus primarily on graph construction or offline enrichment rather than continuous analysis.

Recent benchmark work further highlights the importance of knowledge-augmented reasoning for CTI. CTIARENA introduces a comprehensive evaluation suite covering structured, unstructured, and hybrid CTI tasks under both closed-book and retrieval-augmented settings [14]. Their results show that LLM performance remains limited without external knowledge and improves when augmented with security-specific retrieval strategies, including CSKG-guided retrieval. However, CTIARENA focuses on benchmarking and evaluation rather than proposing an operational architecture or methodology for continuous CTI analysis.

In contrast, our work conceptualizes CTI analysis as a persistent agentic GraphRAG process: a CSKG acts as a persistent knowledge state, graph-based retrieval selects organization-specific contexts (e.g., relevant entities, relationships, and subgraphs), and the LLM performs informed reasoning and explanation over the retrieved context.

E. Summary

In summary, previous work has advanced standardized CTI sharing via STIX/TAXII, CSKG construction and graph-based security analysis, and LLM-based CTI extraction and evaluation. Recent benchmark studies further confirm the importance of knowledge-augmented reasoning for CTI, while highlighting the limitations of LLMs when operating without persistent external knowledge. However, existing approaches typically focus either on structured graph construction or on isolated retrieval-augmented analysis and do not yet systematically integrate persistent knowledge representation, graph-based retrieval, and organization-aware LLM reasoning into a unified GraphRAG pipeline for continuous CTI analysis.

III. Problem Statement and Design Goals

This section derives the core problem and design goals of the proposed approach based on the constraints identified in the previous sections. First, we formalize the challenges of current CTI systems in terms of continuous knowledge integration, contextualization, and organization-specific analysis. We then define a set of design goals that serve as a guide for the architecture and method presented in the following sections.

A. Problem Statement

Modern CTI systems must integrate large amounts of heterogeneous data from structured feeds, internal telemetry, and unstructured text sources. While existing approaches offer either structured threat representations or flexible semantic analyses, they mostly do not support continuous, organization-aware threat modeling in a consistent manner. Knowledge-graph-based CTI systems offer persistent and explainable representations, but lack mechanisms for adaptive semantic interpretation of emerging information. Conversely, LLM-based CTI pipelines offer semantic extraction and analysis capabilities, but operate without a stable, verifiable knowledge state, limiting traceability and reusability.

From an operational perspective, security analysts need contextual threat assessments that reflect both global threat activity and local organizational conditions, such as the criticality, vulnerability, and relevance of assets. Current systems typically treat threat intelligence as generic and static, offering limited support for tailoring analysis to organization-specific environments. This gap leads to delayed prioritization, incomplete situational awareness, and increased manual effort for analysts.

To address these limitations, a CTI system must (i) maintain a persistent representation of threat knowledge, (ii) continuously integrate heterogeneous and evolving information, and (iii) enable adaptive reasoning about this knowledge in an organization-specific context. These requirements motivate a GraphRAG approach, in which a CSKG serves as a persistent structured knowledge state and retrieval mechanism, while LLMs draw informed conclusions about retrieved subgraphs.

B. Design Goals

Based on the identified constraints and requirements, the proposed architecture is guided by this main design goals:

- **DG1 – Persistent and structured knowledge state:** The system should maintain a continuously evolving CSKG that integrates structured and unstructured CTI sources into a unified, verifiable representation.
- **DG2 – Continuous integration of heterogeneous CTI sources:** The architecture must support the automated collection and normalization of various CTI feeds, including structured indicators and unstructured text, to enable timely updating of threat knowledge.
- **DG3 – Graph-based contextual querying:** The system must enable selective querying of relevant subgraphs based on organizational context, e.g., asset exposure, sector, or threat relevance, to provide targeted inputs for analysis.
- **DG4 – LLM-based reasoning and explainability:** LLMs may only be used for interpreting retrieved graph context to enable explainable and traceable threat analysis while avoiding unfounded generations.
- **DG5 – Organization-specific threat analysis:** The architecture must support the prioritization and interpretation of threats tailored to individual organizational environments, rather than generating general threat summaries.
- **DG6 – Agentic task orchestration:** The system should support task-driven orchestration of retrieval, reasoning, and auxiliary analysis steps to enable iterative and goal-oriented CTI analysis workflows.

These design goals form the basis for the system architecture described in the following section and ensure that the proposed approach directly addresses the limitations of existing CTI systems identified in Sections I and II.

IV. System Architecture

The proposed architecture implements an Agentic GraphRAG approach for analyzing CTI. It is designed as a multi-layered architecture that separates data processing, knowledge representation, retrieval, orchestration, and interaction. Figure 1 illustrates the overall architecture, highlighting the role of the CSKG as the persistent knowledge state of the system. A typical analysis starts with a user-defined task, which is interpreted by the agentic orchestration layer to plan and execute a sequence of graph-first retrieval and reasoning steps over the persistent knowledge state. This layered separation reflects the design goals defined in Section III by decoupling knowledge construction, grounded retrieval, and task-driven control.

A. CTI Data Sources

The architecture integrates heterogeneous sources of CTI, including structured CTI feeds (e.g., indicators, vulnerability data), unstructured information, such as reports and alerts, and organization-specific contextual information. These sources provide the raw data for knowledge building but are not directly accessed during analysis.

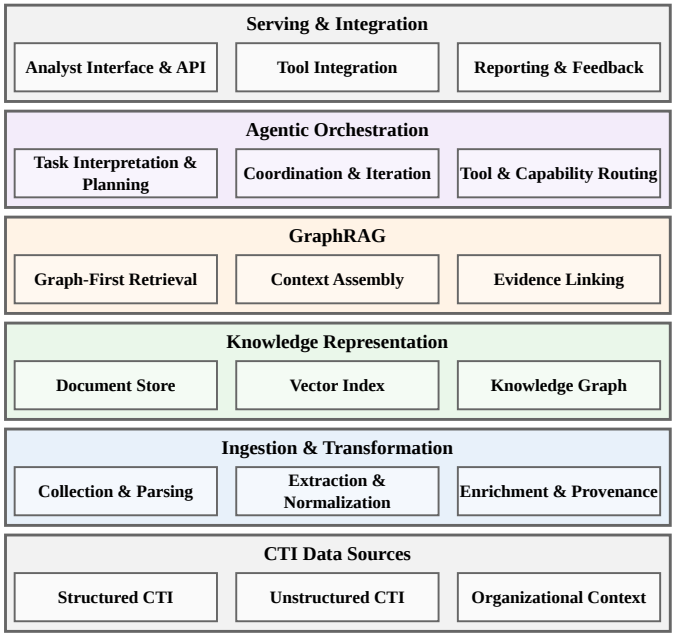


Figure 1. System architecture of the proposed approach.

B. Ingestion and Transformation Layer

All incoming CTI data is collected, analyzed, and converted into a uniform internal representation. This layer performs entity and relationship extraction, normalization, enrichment, and provenance annotation to transform raw data into structured knowledge elements. By consolidating both structured and unstructured sources, the system ensures consistent and traceable integration of information prior to storage.

C. Knowledge Representation Layer

The knowledge representation layer maintains the persistent state of the system and consists of three complementary components: a document store, a vector index, and a CSKG. The document store preserves the original source material for traceability and evidence verification. The vector index enables semantic similarity search over textual intelligence. The CSKG models entities, relationships, and temporal connections, providing a structured and comprehensible representation of the threat landscape. All subsequent analysis steps operate on this unified knowledge state.

D. GraphRAG Layer

The GraphRAG layer provides structured retrieval capabilities over the persistent knowledge state. It performs graph-first retrieval to identify relevant subgraphs based on the analysis task and organizational context, and assembles these subgraphs into a bounded context for downstream reasoning. The linking of evidence ensures that the retrieved graph elements can be traced back to the original documents or sources, enabling traceable and explainable analysis. This layer forms the primary grounding mechanism for downstream reasoning and ensures that LLM-based analysis remains anchored in verified graph context.

E. Agentic Orchestration Layer

The agentic orchestration layer interprets user requests and analysis tasks and decomposes them into executable steps. It coordinates retrieval, reasoning, and auxiliary capability invocation in an iterative manner. Depending on the task, the orchestration layer may trigger multiple retrieval rounds, refine queries, or invoke specialized functions, such as scoring, validation, or enrichment. This design enables flexible, task-oriented analysis while keeping reasoning grounded in retrieved context rather than exposing raw data or the full knowledge base to the reasoning components. By separating control logic from knowledge storage and reasoning, the orchestration layer supports adaptive, goal-driven CTI workflows.

F. Serving and Integration

The service and integration layer makes the results of the analysis available to users and external systems. It provides analyst interfaces, APIs, and integration points for security platforms, such as Security Information and Event Management (SIEM) or Security Orchestration, Automation, and Response (SOAR) systems. The feedback generated in this layer can be incorporated into the persistent knowledge state to support the continuous development of the threat model.

The following section details the GraphRAG workflow and the agentic analysis process in a step-by-step manner.

V. Methodology

This section describes the proposed Agentic GraphRAG methodology for CTI analysis. An overview of the methodology is shown in Figure 2 as an iterative analysis workflow. Building on the layered architecture introduced in Section IV, the methodology specifies how knowledge is constructed, retrieved, and analyzed to support organization-aware threat assessment. The following subsections detail the problem formulation, knowledge construction process, graph-first retrieval strategy, agentic analysis workflow, and grounded reasoning approach.

A. Problem Formulation

The objective of the proposed methodology is to support organization-specific cyber threat analysis by combining persistent, structured threat knowledge with adaptive, task-driven reasoning. Given the continuously growing volume and heterogeneity of CTI, analysts require mechanisms that not only retrieve relevant information but also contextualize, correlate, and prioritize threats with respect to their organization.

In our setting, the system maintains a persistent knowledge state that consists of (i) a CSKG capturing entities, relations, and temporal context, (ii) an associated document store preserving original sources as evidence, and (iii) a vector index enabling semantic similarity search over textual intelligence. An analysis task provided by a user or an external system defines the analytical objective, such as threat prioritization, exposure assessment, or campaign interpretation, optionally parameterized by organizational context (e.g., assets, sector, or risk posture).

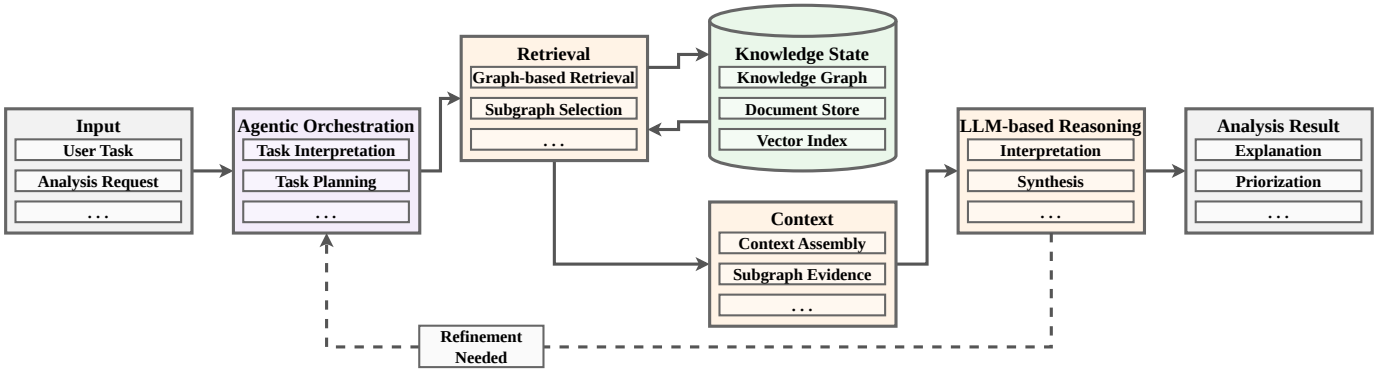


Figure 2. Agentic GraphRAG methodology for CTI analysis.

The methodology aims to produce analysis results that are grounded in verified knowledge and traceable to original evidence. To this end, the process iteratively performs graph-first retrieval to select coherent subgraphs relevant to the task and organizational context, assembles a bounded context window including linked evidence, and applies LLM-based reasoning over this structured input. Agentic orchestration controls the sequence of retrieval and reasoning steps, enabling multi-step analysis, query refinement, and auxiliary capability invocation while keeping the reasoning component isolated from raw sources and the full knowledge base.

This formulation ensures that analysis results remain explainable, organization-aware, and adaptable to evolving threat landscapes while preserving a persistent and verifiable knowledge state.

B. Knowledge Construction Process

To realize the persistent knowledge state defined in the problem formulation, the methodology implements a continuous knowledge construction process that transforms heterogeneous CTI inputs into a unified and verifiable representation supporting structured retrieval and reasoning. This process operates continuously and incrementally, enabling the system to integrate newly observed intelligence without requiring complete reconstruction of existing knowledge.

Structured CTI sources are mapped to a canonical schema and directly transformed into graph entities and relationships. Unstructured sources, such as reports or advisories, are processed using extraction techniques to identify relevant entities, relations, and contextual attributes. All extracted elements are normalized, enriched, and annotated with provenance and temporal metadata to ensure consistency and traceability.

The resulting knowledge is stored in three tightly coupled representations. Original documents are preserved in a document store, providing verifiable evidence and auditability. Textual content is embedded into a vector index to enable semantic similarity search and matching. In parallel, extracted entities and relationships are integrated into the CSKG, where they are linked to corresponding documents and vector representations.

By consolidating structured and unstructured intelligence into a single persistent knowledge state, the construction process establishes a stable and extensible foundation for graph-first retrieval and agentic analysis. This separation between knowledge construction and knowledge utilization ensures that downstream reasoning operates over curated and verifiable context rather than raw and potentially noisy input data.

C. Graph-Based Retrieval

Graph-first retrieval provides the primary mechanism for grounding all downstream analysis in structured and organization-relevant knowledge. Given a task provided by a user or an external system and optional organizational context, the retrieval process operates over the CSKG to identify coherent subgraphs that capture relevant threat entities, relationships, and temporal dependencies.

Retrieval begins by translating the analysis task into one or more graph queries that encode semantic intent and contextual constraints. These queries may incorporate organizational factors, such as asset exposure, sector-specific relevance, or previously observed activity. Rather than retrieving isolated entities or documents, the process selects connected subgraphs that preserve relational structure and enable holistic interpretation of threats.

The retrieved subgraphs are subsequently ranked and filtered based on relevance, structural coherence, and contextual alignment. To support explainability, each graph element is linked to corresponding evidence in the document store and vector index, allowing retrieved context to be traced back to original intelligence sources.

By operating on explicit relationships and graph topology, the proposed retrieval strategy enables precise context selection that is not achievable with purely text-based retrieval. This graph-first approach ensures that all reasoning steps are grounded in structured knowledge and that retrieved context remains bounded, coherent, and verifiable.

D. Agentic Analysis Workflow

The agentic analysis workflow controls the execution of retrieval and reasoning steps in a task-driven and iterative manner. When an analysis task is provided by a user or external system, the orchestration layer interprets the task objective and decomposes it into a sequence of intermediate goals, such as

identifying relevant threats, correlating activities, or assessing organizational exposure.

For each intermediate goal, the orchestration layer determines the required retrieval and processing actions. This may include issuing graph-first retrieval queries, refining previously retrieved context, or invoking auxiliary capabilities, such as scoring, validation, or enrichment. Retrieved subgraphs and associated evidence are incrementally assembled into a bounded analysis context that is passed to the reasoning component.

The workflow supports iterative refinement by allowing the orchestration layer to re-evaluate intermediate results and trigger additional retrieval or processing steps when necessary. Iteration continues until the task objectives are satisfied or predefined stopping criteria are met, ensuring that the analysis remains focused and bounded.

By explicitly separating control logic from reasoning, the agentic workflow enables adaptive, multi-step analysis while maintaining strict grounding in verified knowledge, allowing the system to dynamically adjust retrieval and reasoning strategies based on intermediate results without exposing the full knowledge base or raw data to the reasoning model.

E. Reasoning and Output Generation

Reasoning is performed exclusively over the bounded and verified context assembled by the graph-first retrieval and agentic orchestration layers. The reasoning component receives a structured subgraph, linked evidence from the document store, and task-specific instructions, ensuring that analytical conclusions remain grounded in persistent knowledge.

The reasoning process focuses on interpreting relationships, identifying patterns, and synthesizing insights relevant to the analysis objective. Typical outputs include threat prioritization, contextualized explanations, exposure assessments, or hypothesis generation. Because the reasoning model operates exclusively on retrieved context, the risk of hallucination is reduced and outputs remain traceable to original sources.

Generated results are returned through the serving layer and may include references to supporting evidence or graph elements to support analyst validation. Optionally, validated outputs can be incorporated back into the persistent knowledge state, enabling incremental refinement of the threat model.

VI. Conclusion and Future Work

This paper introduced an Agentic GraphRAG architecture and methodology for organization-aware CTI analysis. The proposed approach combines a CSKG with graph-first retrieval, agentic orchestration, and grounded LLM-based reasoning to address the challenges of heterogeneity, scale, and contextualization in modern CTI environments. By separating knowledge construction, retrieval, control, and reasoning, the architecture enables continuous integration of structured and unstructured intelligence while maintaining traceability and explainability. The agentic analysis workflow supports multi-step and context-driven threat assessment without exposing raw data or the full knowledge base to the reasoning model, thereby reducing hallucination risks and improving analytical reliability.

Future work will focus on extending the prototype implementation and conducting systematic evaluations in realistic operational scenarios. Planned directions include quantitative assessment of retrieval quality and reasoning accuracy, as well as comparisons against baseline approaches such as text-based RAG pipelines, CSKG-only retrieval, and LLM-only analysis. Further work will investigate scalability aspects for continuously growing CTI datasets, including incremental graph updates and efficient subgraph retrieval. Robustness considerations such as knowledge graph poisoning and prompt injection attacks will also be explored. In addition, the knowledge graph will be extended to support temporal and causal reasoning to enable deeper analysis of evolving attack campaigns.

Overall, the proposed architecture establishes a foundation for adaptive and explainable CTI systems that integrate persistent structured knowledge with flexible AI-driven analysis.

References

[1] ENISA, "Enisa threat landscape 2025", European Union Agency for Cybersecurity, 2025. DOI: 10.2824/1946374.

[2] L. F. Sikos, "Cybersecurity knowledge graphs", *Knowledge and Information Systems*, vol. 65, no. 2, pp. 351–379, 2023. DOI: 10.1007/s10115-022-01734-4.

[3] X. Zhao, R. Jiang, Y. Han, A. Li, and Z. Peng, "A survey on cybersecurity knowledge graph construction", *Computers & Security*, vol. 136, p. 103 524, 2024. DOI: 10.1016/j.cose.2023.103524.

[4] M. T. Alam, D. Bhusal, L. Nguyen, and N. Rastogi, *Ctibench: A benchmark for evaluating llms in cyber threat intelligence*, 2024. arXiv: 2406.07599.

[5] OASIS, *STIX Version 2.1*, OASIS Standard, 2021.

[6] OASIS, *TAXII Version 2.1*, OASIS Standard, 2021.

[7] A. M. Konsta and X. D. Koutsoukos, "A survey of automatic generation of attack trees and attack graphs", *Computers & Security*, vol. 125, p. 103 043, 2023. DOI: 10.1016/j.cose.2022.103043.

[8] A.-M. Konsta, A. Lluch Lafuente, B. Spiga, and N. Dragoni, "Survey: Automatic generation of attack trees and attack graphs", *Computers & Security*, vol. 137, p. 103 602, 2024. DOI: 10.1016/j.cose.2023.103602.

[9] B. Yan et al., "Graph mining for cybersecurity: A survey", *ACM Transactions on Knowledge Discovery from Data*, vol. 18, no. 2, pp. 1–52, 2023. DOI: 10.1145/3610228.

[10] J. Zhao, Q. Yan, X. Liu, B. Li, and G. Zuo, "Cyber threat intelligence modeling based on heterogeneous graph convolutional network", in *Proceedings of the 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, USENIX Association, 2020, pp. 241–256.

[11] A. Krašovec, G. Steri, G. Karopoulos, and M. Trapani, "Large language models for cyber threat intelligence: Extracting mitre with llms", in *Proceedings of the 20th International Conference on Availability, Reliability and Security (ARES)*, Springer Nature Switzerland, 2025, pp. 80–89.

[12] Y. Cheng, O. Bajaber, S. A. Tsegai, D. Song, and P. Gao, *Ctinexus: Automatic cyber threat intelligence knowledge graph construction using large language models*, 2025. arXiv: 2410.21060.

[13] L. Huang and X. Xiao, "Ctikg: Llm-powered knowledge graph construction from cyber threat intelligence articles", in *First Conference on Language Modeling*, 2024.

[14] Y. Cheng, Y. Liu, C. Li, D. Song, and P. Gao, *Ctiarena: Benchmarking llm knowledge and reasoning across heterogeneous cyber threat intelligence*, 2025. arXiv: 2510.11974.

An Analysis of Attack Vectors Against FIDO2 Authentication

Alexander Berladskyy and Andreas Aßmuth
Kiel University of Applied Sciences, Kiel, Germany
e-mail: alexander.berladskyy@stu.haw-kiel.de, andreas.assmuth@haw-kiel.de

***Abstract*—Phishing attacks remain one of the most prevalent threats to online security, with the Anti-Phishing Working Group reporting over 890,000 attacks in Q3 2025 alone. Traditional password-based authentication is particularly vulnerable to such attacks, prompting the development of more secure alternatives. This paper examines passkeys, also known as FIDO2, which claim to provide phishing-resistant authentication through asymmetric cryptography. In this approach, a private key is stored on a user's device, the authenticator, while the server stores the corresponding public key. During authentication, the server generates a challenge that the user signs with the private key; the server then verifies the signature and establishes a session. We present passkey workflows and review state-of-the-art attack vectors from related work alongside newly identified approaches. Two attacks are implemented and evaluated: the Infected Authenticator attack, which generates attacker-known keys on a corrupted authenticator, and the Authenticator Deception attack, which spoofs a target website by modifying the browser's certificate authority store, installing a valid certificate, and intercepting user traffic. An attacker relays a legitimate challenge from the real server to a user, who signs it, allowing the attacker to authenticate as the victim. Our results demonstrate that successful attacks on passkeys require substantial effort and resources. The claim that passkeys are phishing-resistant largely holds true, significantly raising the bar compared to traditional password-based authentication.**

Keywords-Passkeys;FIDO2;CTAP;WebAuthn.

I. Introduction

Passwords have been the predominant authentication method for decades. A password is a user-chosen string that is hashed and stored server-side. Upon authentication, the user's input is hashed and compared with the stored hash to verify its validity. This method is simple to implement and remains the most widely used authentication mechanism.

However, passwords have significant drawbacks. They are vulnerable to phishing attacks, and in the event of a server-side data breach, the hashed passwords can be exposed and potentially cracked. The Anti-Phishing Working Group (APWG) reported 892,494 phishing attacks in the third quarter of 2025 alone [1], and data breaches continue to be a prevalent threat.

In response to these vulnerabilities, passkeys, based on the Fast Identity Online (FIDO2) standard, provide a more secure alternative. These authentication methods are designed to be resistant to phishing, ensuring that even if a data leak occurs, no sensitive information that could compromise the user's security is exposed.

This work investigates various attack vectors targeting FIDO2 to assess the feasibility and cost of potential attacks, and to evaluate whether it is truly resistant to phishing. The structure of the paper is as follows: Section II provides an overview of how FIDO2 works, Section III reviews previous research on attacks against passkeys, Section IV outlines the attack vectors examined, Section V demonstrates two specific attacks, Section VI discusses the findings, and Section VII presents the conclusions and directions for future research.

II. Background

The following sections describe the concepts of FIDO2 and introduce the threat model.

A. Client To Authenticator Protocol

The Client To Authenticator Protocol (CTAP) [2][3] is a communication protocol that allows a client, such as a browser, to communicate with an authenticator. An authenticator can be a hardware device, like a phone or a hardware key [4] or software-based authenticators that communicate using the same protocol. Examples include Windows Hello or applications that implement security mechanisms. CTAP, a core component of FIDO2 (cf. project's website [5]), enables passwordless and secure authentication on the Internet. Unlike password-based systems where authentication occurs server-side through hash comparison, authentication with CTAP is performed on the devices which belong to the user. After the authentication on the local device, e.g., by fingerprint or a PIN, the secret key can be accessed and used for authentication by signing a challenge received by the Relying Party (RP). The RP is the service which a user is trying to authenticate to. This could be any web service, providing a way to authenticate to a user account. The RP saves the public key of the owner and thus can verify if the person is the real owner of the key. The private key, which is generated by the authenticator, remains inaccessible to the end users – they only need a way to unlock the authenticator, e.g., a PIN or a fingerprint, and a device that is able to use CTAP. A key pair is also called *passkey* in this context.

B. WebAuthn

The second part of FIDO2 is WebAuthn [6][7] (Web Authentication API). CTAP covered the communication between the client and the authenticator. The communication between the client and the RP is done by WebAuthn. WebAuthn implements two workflows, also called ceremonies, namely signup or registration and the login or authentication workflow. The signup process begins by the user choosing a passkey login instead of a password login. The RP receives the request, containing user information, and generates a challenge, which is random data, e.g., an array of random numbers. The challenge and the user information is stored temporarily on the server, until the ceremony is finished. Then, the server responds with the challenge, user information and server information back

to the client. Now the client tries to create the necessary credentials (public-key pair), by first authenticating the user via CTAP. After the generation of the keys, the private key is used to sign the challenge – this process is also called assertion. Attestation on the other hand, is a certificate, verifiable by the RP, that proves the key being generated by a specific authenticator device like a Hardware Security Key (HSK). The signed challenge, user information, server (RP) information, device information and the public key are sent to the server. Finally, upon verifying the challenge and comparing the user information the process is done.

Similarly, the login workflow starts by the user sending a passkey login request and the RP generating a challenge. The RP responds with the challenge and user information and the client requests a signature on the challenge from the authenticator through CTAP. The challenge is signed and is sent with user and device information to the RP. The RP locates the public key, through the sent information and verifies the signed challenge - the authentication is done. The authentication workflow is visualized in a more simplified way in Figure 1, using a browser as the client: ⓪ the client starts passkey authentication, ① the RP responds with a challenge, ② the challenge gets forwarded to the authenticator, ③ the authenticator signs the challenge with the correct private key, ④ then the authenticator sends the signed challenge to the client, ⑤ & ⑥ the challenge gets forwarded to the RP, gets verified and the authentication workflow is finished.

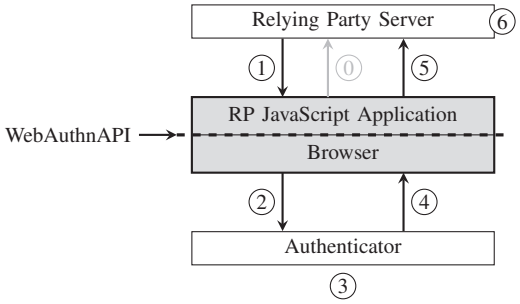


Figure 1. WebAuthn authentication workflow using a browser as client, cf. [6]

C. FIDO2

FIDO2 now incorporates both CTAP and WebAuthn.

> *FIDO2 standards use standard public key cryptography techniques to provide phishing-resistant authentication with cryptographic key pairs called passkeys. FIDO2 is designed from the ground up to protect user privacy and prevent phishing. Every passkey is unique and bound to the online service domain. The protocols do not provide information that can be used by different online services to collaborate and track a user across the services. Biometric information, if used, never leaves the user's device.* [5]

The entire workflow is already described in the sections above. In a more high level perspective:

1) A user is visiting a website and registers.
2) The device of the user is generating a passkey.
3) The passkey is used to login into the webservice.

This workflow is passwordless and also called Universal Authentication Framework (UAF). Another option is the Universal Second Factor (U2F), which incorporates a physical second factor, such as a (physical) security key.

D. Threat Model

The threat model assumes an attacker being able to compromise a victim's device with malware and perform Adversary in the Middle (AITM) attacks, such as a combination of AITM and Domain Name System (DNS) Spoofing [8]. Additionally, the victim may follow phishing links provided by the adversary. The user may use a synchronized authenticator, meaning that the passkeys are stored in the cloud, or a device-bound authenticator (local storage). The threat model focuses on attacks against end users rather than the FIDO2 protocol itself, which is assumed to be cryptographically secure. The main goal of the adversary is to gain access to the user account or to pull out personal information.

For the AITM attack, a scenario would include the adversary to be in the same network as the victim or have malware installed which performs an AITM against, e.g., an HSK.

The malware scenario includes the victim having malware installed on their device. This could be through a malicious download, a malware installation by the adversary locally (physical access) or a local malware installation by the victim via, e.g., a bad USB. In general, there are several possibilities.

A phishing scenario, includes the attacker being able to trick the victim to click on a link and to trust the phishing website through visual deception. The attacker is also able to forward Multi Factor Authentication (MFA) requests, like a text message verification on a login.

III. Related Work

This section summarizes attacks on FIDO2 which have been shown in previous publications.

Li et al. [9] proposed an Authenticator Rebind Attack on the UAF protocol, targeted at Android phones. The lack of effective entity authentication in the UAF protocols allows an attacker to bind the victim's identity to the attacker's authenticator. This attack requires pre-installed malware on the victim's device, which redirects some of the traffic to the attacker or the attacker has to have some access prior to the attack. When the victim tries to enable fingerprint authentication, the request is forwarded to the attacker's authenticator. The attacker performs their own fingerprint verification initiated by the misused authenticator and the victim performs a fake fingerprint verification initiated by the malware. The authors demonstrated the attack's effectiveness against real-world applications, providing a real threat under the given circumstances.

Barbosa et al. [10] showed two possible attacks on FIDO2, one being an impersonation attack and the other one being a rogue key attack. On the impersonation attack, the adversaries

capture the *pinToken* by using an AITM attack. The pinToken, a cryptographic key, is used by the client to send Message Authentication Code (MAC)-authenticated credential and assertion requests to the authenticator. The adversary intercepts active communication between the client and authenticator during the key agreement phase. During this phase, the adversary agrees with their own generated keys to the key agreement protocol. By doing that, the adversary, knowing the shared secret, creates a session with the client and authenticator. Now the adversary is able to create assertion requests to any relying party. Similar to Li et al. [9], the second attack includes registering a rogue key on the RP, but through the USB HID mechanism of the latest FIDO2 version.

Kumar Yadav et al. [11], present local attacks on the FIDO2 protocol. The attack methods are similar to the attacks presented previously [10], since one of the goals is to impersonate the victim. Here, browser extensions are used for performing an AITM attack between the authenticator (a hardware security key) and the RP through WebAuthn. The other goal is to bypass the cloning detection algorithm of an HSK by FIDO2. The authors underline, that the RPs are lacking detailed error messages, to notify victims that they are being attacked.

Mahdad et al. [12], propose an attack which includes running malware in the user space and uses a browser extension. The malware detects keystrokes and mouse movements - the primary function is to capture username and password in an MFA scenario. A hidden browser is launched in the background, navigating to the same service the user is trying to authenticate to. The browser extension, on the victim's browser, redirects the page to a similar-looking attacker-controlled page, prompting the victim to access their security key. Next, the hidden browser attempts to login, receives the challenge and forwards it to the attacker-controlled page. After the user authenticates with the key, the attacker is able to establish a hidden session.

Donghyun et al. [13], present a different kind of attack, on which the victim does not have to be preloaded with some kind of malware. This attack exploits the Bluetooth capabilities of CTAP and uses spoofing to reroute the victim to a phishing server. Upon visiting the phishing website and trying to log in, the adversary goes to the real RP and attempts to login. The RP responds with the relevant data needed to generate a QR-Code and the challenge. The adversary generates a legitimate QR-Code out of the received data and presents it to the victim. Next, the victim scans the QR-Code and tries to establish a Bluetooth connection to the device that generated the QR-Code. After the connection establishment, the challenge gets signed, forwarded to the adversary and thus logging the adversary in.

IV. Methodology

This section enumerates attack vectors against FIDO2 and outlines the requirements for successful exploitation. It is important to note that the following attacks do not target FIDO2 directly, as it is assumed to be secure (Section II-D); rather, they illustrate potential attack vectors, which further demonstrate the robustness of FIDO2. Each attack vector is classified according to its complexity level and setup effort of the attack; with easy, medium or hard/impossible. A key observation is that all attack scenarios require substantial effort. In all cases, an attacker must either compromise the victim's system or be within close physical range. Notably, none of the described attacks offer a method for remotely (high range) exploiting FIDO2 without first compromising the victim's system.

As already mentioned in the threat model (Section II-D), the main goal of an attacker is to gain access to the user account or to gather personal information. Personal user information could then be used to open new attack vectors on the victim; this won't be further discussed. For accessing the account, several paths exist like the ones mentioned in Section III - those will be included below.

A. Getting Victim's Key

One path, which will be practically impossible, especially on synchronized authenticators (due to cloud storage), is to get the private key for an existing account. With malware, it is possible to extract private keys, depending on where they are stored. Password managers like KeePassXC [14] store the passkeys in an encrypted database on the file system. To pull out the private keys, the encryption of the database has to be cracked; this requires malware to be installed or local access to the target device, which can be hard. Extraction methods vary from different authenticators, of course; e.g., HSKs have to be locally accessed since the keys are generated and stored on the device. The private key does not leave the HSK but is only used for signing, making it significantly harder to extract.

B. Victim Impersonation

Other paths are more feasible, like impersonating the victim. For this to work, the authentication process of FIDO2 has to be hijacked. One option would be to create a session with the RP and the client in the key agreement phase, which requires an AITM between the authenticator and the client. Barbosa et al. [10] showed this practically with the USB HID transport mechanism; due to lack of authentication on Elliptic Curve Diffie Hellman (ECDH) in CTAP, an attacker is able to create a session to the RP. The AITM attack requires malware to be installed on the victim's device. If the attacker is positioned between the authenticator and the client, a session can be established or the attacker's own key can be registered on the RP. It is important to mention that the success of this attack depends on the user being present and on malware being installed. Due to this, this path of the impersonation attack would be hard.

C. Bluetooth AITM

The aforementioned Bluetooth attack [13], where the victim does not have to be preloaded with malware, operates in a similar way. This attack adds the use of a phishing page, which is only there for visual deception. The authentication is performed on an attacker's remote device. Further, the attack requires the attacker to be in range of a user, because the Bluetooth authentication of CTAP has to be hijacked. This path depends on a user wanting to authenticate with QR-Code

scanning, the attacker being in range and a successful phishing attack. This attack will be classified as medium, because the user does not have to be preloaded with malware but still has to interact with a phishing page.

D. Authenticator Deception

Passkeys are phishing resistant [5], but still phishing websites can be utilized to gain access to the victim's account. To login as the victim, the attacker has to own a website, with the same domain as the RP. Since this is not possible, the attacker performs a DNS spoofing attack [8], forwarding requests to the phishing website. A problem which arises is the SSL certificate, which the attacker does not own. Two sub paths arise, mainly using an insecure connection or forcing the victim to trust a self signed certificate. An HTTP connection might technically work, most browsers don't explicitly warn you as they would with self-signed certificates - for inattentive users this would go unnoticed. However, the in-browser WebAuthn API only allows HTTP access when running on localhost, making it impossible to use WebAuthn over an HTTP connection in a real attack scenario. When using HTTPS the victim has to trust the self signed certificates, because it is not possible to gain the real certificate of the RP. This requires malware or physical access to the victim's device to install the certificate, either into the browser or into the OS certificate authorities. The latter option would be even harder, because this path usually needs administrative rights. First, a DNS spoof is performed and the victim is lured to the phishing website of the RP. Since the interface and the domain is the same as in RP, the authenticator will recognize it and start the process. In the background, the attacker starts a session with the real RP and forwards the requests, including the challenge, to the phishing website. After the victim's authenticator has signed the challenge, it is forwarded back to the attacker, allowing login access. To remove any suspicions, the phishing server shows an error and recommends reloading the page, so the victim will be forwarded to the real RP. This requires the attacker to turn off DNS spoofing, after the successful login. The setup of this attack is easy to realize but access to the target's device is required, making the attack medium to hard.

E. Registering Attacker's Passkey

Similar to the victim impersonation attacks, an attacker can try registering their own passkey, on behalf of the victim. One way would be to hijack the process of adding passkeys on the RP. To motivate the victim, a fake email can be sent, which recommends that the user add passkeys. When the victim now tries to add passkeys, the attacker has to sit in between the authenticator and the client, thus needing an AITM. Since the goal is to register the keys on the real RP, a simple AITM over the network and DNS spoofing is not sufficient. The AITM has to be run locally, on the victim's device, which again requires malware. When a request to add passkeys is observed, the challenge is signed with the attacker's own private key, and the victim's public key is replaced. This attack is not too stealthy, since the whole FIDO2 operation fails on the victim's side. To optimize this, the attacker can make it look as if the operation was successful by forwarding and modifying packets, marking the authentication as successful from the RP's side. Obviously, the victim won't be able to login with their newly created passkey, due to the public key not matching. The described method will be ranked as hard, because malware has to be installed. The second option for registering own passkeys is to login as the user via traditional methods like email and password. For this, a phishing website is used which captures the login data. After the victim enters their credentials, the attacker logs in and registers attacker-controlled keys. This attack is easy to realize and will also be ranked in this category. This attack succeeds only if the victim uses email and password authentication. If a passkey is used as a second factor, this attack fails, because the phishing domain is not matching. Here, a phishing website as proposed in Section IV-D, can be used to hijack the CTAP communication. This attack focuses on the attacker registering their own keys – the phishing website is used as a tool to gain initial access. In case of password change, the key remains valid. Unlike conventional phishing attacks, which lose effectiveness once the victim changes their password, this attack establishes persistent access through a registered passkey that remains valid independently of any subsequent credential changes.

F. Passkey Reduction

Many websites [15] provide the option to register a passkey instead of typical password authentication. But password authentication is not gone and is still used. This enables a user to authenticate via multiple methods. Since one of the main goals was to gain access to the account, an attacker can utilize a passkey reduction attack. First, the victim is lured to a phishing page and goes to the login page. The passkey option will either not be shown or upon choosing *passkey authentication*, the fake RP signals an error and prompts the victim for password authentication. This attack depends on the user believing that passkey authentication is currently unavailable. The setup, on the other hand, is easy to realize and thus the attack is ranked in the easy category. When passkeys are configured as a second factor in MFA, the attacker has to spoof the real RP's domain to sign the challenge as described in Section IV-D.

G. Infected Authenticator

The final attack on passkeys, where the goal is to gain access to the victim's account, uses an *infected authenticator*. The attacker modifies the underlying source code of an authenticator, making it generate known key pairs. When a user creates accounts on the RP, the attacker will be able to login, since the private key is known. Either the application's source code or the used cryptographic libraries, which are used to generate the keys, can be modified. Manipulating system libraries can be more challenging because it typically requires superuser permissions. When a library is modified, all applications using the library will generate known keys. The complexity in this attack lies in the installation of malware (infected authenticator) or malware which modifies the target library or application. This

attack is limited to local authenticators, meaning the key pairs must be generated locally on the victim's device. This attack path is ranked medium to hard, because of malware installation but the easy realisation. The complexity increases even more when system libraries get modified. However, implementing malicious code into applications is straightforward. Alternative approaches include exfiltrating generated keys over the network. A downside is that such behaviour can be detected by anti-malware systems [16]; e.g., an anti-malware system can detect and flag the use of sockets, because it can associate it to network activity. Other malware-reliant attacks described in this section, such as local AITM or certificate store manipulation, do not require outbound network connections to attacker-controlled infrastructure and are therefore less likely to trigger such detection mechanisms.

H. RP Impersonation

In the final attack scenario, an attempt is made to extract personal information from the victim by causing the victim to believe that authentication is being performed with a legitimate RP. To achieve this, an attacker must spoof both the RP's website and the CTAP interaction. As a result, the victim is led to believe that a successful login has occurred, even though the entire process is a visual deception. This attack requires the attacker to replicate the genuine user interface of a logged in RP session. Further, the FIDO2-related library on the victim's device must be modified so that the authentication process appears successful regardless of the actual input. Once the victim is "logged in", the attacker can display prompts requesting personal information under the guise of security checks. Of course, there are numerous strategies of how information can be extracted. A simpler variant is to force the victim to bypass passkey authentication entirely and directly request personal information, similar to Section IV-F. This variant only requires a successful phishing attack, making it relatively easy to execute. Note that this simpler variant does not rely on FIDO2 at all and would be equally effective against any authentication method; it is included here because it represents a degraded form of the full RP impersonation attack, which requires the installation of malware and the manipulation of RP libraries, making it significantly more complex and thus ranked hard.

Table I provides an overview of the attacks characteristics, which will be analysed in the next section. In the first row of the table, IE is used as an abbreviation for *Infected Authenticator* and AD for *Authenticator Deception*.

TABLE I. OVERVIEW OF THE ATTACKS

Criterion	IE	AD
Stealthiness	High	Medium
Feasibility	Medium	High
Victim Interaction	Low	High
Time Consumption	Low	Low-High
Privileges	Low-High	Low-High

V. EXPERIMENTS

Two experiments were conducted to demonstrate: first, that executing an attack on FIDO2 is highly resource-intensive, and second, that the attacks closely resemble phishing methods. Since passkeys cannot be phished directly, Section V-A describes an attacker being able to possess the same passkey. Similarly, Section V-B mirrors a typical phishing scenario, where a legitimate website is displayed, but the attacker's goal is to steal the proof of authentication.

A. Infected Authenticator

The first experiment follows the method described in Section IV-G. In this attack, KeePassXC, a local authenticator, is used, and only the source code of the binary is modified; no changes are made to the system libraries responsible for FIDO2. As previously mentioned, the source code of the authenticator is altered so that an attacker can obtain the same key that the victim generates during the registration process. KeePassXC relies on the Botan cryptographic library, which is used to generate asymmetric keys. Specifically, KeePassXC requires the generation of Elliptic Curve Digital Signature Algorithm (ECDSA), Rivest–Shamir–Adleman (RSA), and Edwards-curve 25519 Digital Signature Algorithm (Ed25519) keys.

The attack proceeds as follows: the attacker first generates their own public-private key pairs using Botan, which are saved in PEM format for later import into the KeePassXC database. This key is then embedded directly into KeePassXC's source code as a string variable. In the modified code, the private key in PEM format is loaded into the appropriate object type, while the rest of the code remains unchanged. This ensures that every time a new passkey is registered, the same private key is used, leading to the same public key being generated by Botan.

Depending on the RP, the public key (used to verify the challenge) is associated with a unique credential-ID. KeePassXC generates these credential IDs, which are then stored on the RP. Therefore, the attacker must also know the correct credential-ID, which is also easily embedded into the source code. The binary is then placed on the victim's computer, essentially replacing the old KeePassXC binary. After the victim registers their passkey on the RP, the attacker can authenticate and log in using the previously generated key pair. This experiment was conducted on a Firefox browser, specifically for logging into a Google account.

B. Authenticator Deception

The second experiment realizes the Authenticator Deception attack, which was described in Section IV-D. The attack setup was as follows. The victim was using a Windows 10 machine, while the attacker operated from a Kali Linux system. The victim had a registered test account on the target domain: *linear.app* – in general, any domain can be targeted, provided it implements FIDO2. The attacker also possessed a valid account on linear.app, which was required in order to modify authentication requests dynamically during the attack.

First, the attacker generated a valid TLS certificate for the fraudulent frontend server that would later be presented to the victim as a result of DNS spoofing [8]. Since modern browsers do not trust self-signed certificates by default, the attacker requires prior access to the victim's machine. Specifically, the attacker created a custom Certificate Authority (CA) and signed the frontend certificate with this CA, allowing the certificate to be valid. The custom CA certificate was then imported into the victim's browser (Firefox) trust store, effectively forcing the browser to trust the attacker's frontend certificate. Additionally, the frontend certificate contained a Subject Alternative Name (SAN) extension as well as a Common Name (CN) matching the target domain, in this case linear.app. Only after these preparatory steps was the victim's machine considered compromised and the actual attack could begin.

For the attacker-controlled infrastructure, an NGINX web server was used as the frontend, combined with an Express-based Node.js application acting as the backend. This backend handled incoming requests from the victim's browser and proxied or modified them as necessary before forwarding them to the legitimate RP.

The attacker initiated an ARP spoofing attack [8], thereby poisoning the victim's ARP table and replacing the legitimate router's MAC address with the attacker's own. In parallel, a DNS spoofing attack was launched that mapped all DNS requests for linear.app and *.linear.app to the attacker's local IP address. As a result, all traffic intended for linear.app was redirected to the attacker-controlled frontend server.

When the victim navigated to linear.app using the compromised browser, the request was routed to the attacker's NGINX frontend ①. The cloned website provided the option to authenticate using passkeys, which the victim selected. Upon initiating passkey authentication, a challenge was required. To obtain a valid challenge, the attacker simultaneously initiated an authentication attempt with the legitimate linear.app RP ②. Using Burp Suite as an AITM proxy, the attacker intercepted the authentication request and extracted both the challenge and the associated authenticationId ③. This identifier is a temporary value that uniquely binds a specific authentication session and is required to complete the passkey authentication flow.

The attacker's backend then forwarded the captured challenge to the cloned frontend, prompting the victim to sign it using their passkey ④ . In this setup, the authenticator software used by the victim was Bitwarden [17]. Once the victim approved the request ⑤, the signed challenge was produced and returned to the attacker-controlled backend ⑥.

At this stage, the attacker authenticated to the legitimate relying party using their own passkey ⑦. Before completing the authentication, the attacker manually modified the final request by replacing their own signed challenge and user identifier with those obtained from the victim. By submitting the victim's signed challenge together with the victim's user ID, the attacker was able to successfully authenticate as the victim on the real linear.app service.

All steps described above were performed manually. The attacker manually transferred the challenge, initiated authentication flows, and edited the final authentication request.

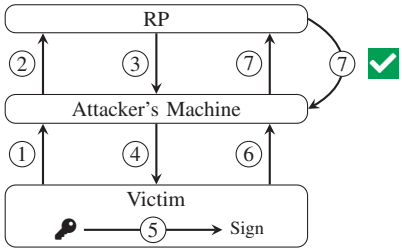


Figure 2. Authenticator Deception Flow

VI. Results and Discussion

Both experiments succeeded as described; however, several important nuances were observed.

For the authenticator deception attack, it was important that the DNS cache on the victim's machine was flushed. If stale DNS entries were present, the victim's browser would continue to resolve the domain to the legitimate RP, bypassing the attacker-controlled server. Since compromising or gaining access to the victim's machine was already a prerequisite for the attack, the attacker was able to flush the DNS cache manually. In a realistic malware scenario, this step is straightforward to automate, as flushing the DNS cache requires only a simple system command that typically does not require elevated privileges. This ensured that subsequent DNS queries were forwarded to the DNS server where the attacker was positioned as an AITM and could respond with their own IP address, pointing to the malicious frontend server. Clearly, in a real-world attack scenario, such an attack would not be executed in such a manual fashion. Instead, automation tools would be employed, such as Selenium for browser automation and a programmable AITM proxy capable of modifying packets dynamically. For demonstration purposes, we did not create a visual clone of the legitimate website; rather, the malicious frontend only provided an option to authenticate via passkeys. In a realistic attack scenario, however, the website would need to be visually indistinguishable from the legitimate RP's website to avoid raising suspicion on the victim's side. Finally, it is important to emphasize that this attack is not trivial to carry out. A critical requirement is the compromise of the victim's machine, which either involves tricking the user into installing malware that modifies the browser's trust store, flushes the DNS cache, and performs related actions, or requires physical access to the system in order to apply these modifications manually. Additionally, this attack is highly targeted at a specific individual, as the attacker must know in advance which website to clone, be located within the same network as the victim and successfully perform an ARP spoofing attack [8] against the target. The attack succeeds only if all these conditions are met. Moreover, the attack will fail if either the router or the victim's machine implements anti-ARP-spoofing mechanisms, which would effectively prevent DNS spoofing.

The infected authenticator attack targeted a specific local authenticator, namely KeePassXC. This attack does not apply to cloud-based authenticators such as Bitwarden. Consequently, the victim was required to use an authenticator that generates cryptographic keys locally. Under these conditions, malware can hijack the key-generation process and deliberately produce attacker-controlled or predictable keys. The described method demonstrated that previously generated keys were hardcoded as strings directly into the source code of the authenticator. This approach causes the authenticator to always generate the same keys and the same credential-ID during passkey registration. As a result, a victim using an infected authenticator would only be able to generate a single passkey per RP. A more effective approach from an attacker's perspective would be to use a predictable algorithm that generates keys and credential-IDs following a known pattern, or to rely on a predictable seed value from which keys are derived. Alternatively, multiple keys and credential-IDs could be embedded into the program, although this approach remains limited. Another option discussed would be to leave the authenticator's behavior unchanged while sharing the generated keys and credential-IDs over the network to the attacker. This would allow the attacker to obtain valid credentials without immediately breaking the expected functionality of the authenticator.

The demonstrated attacks showed that executing an attack on FIDO2 is time-consuming and therefore resource-intensive.

VII. Conclusion and Future Work

While passwords remain a significant security weakness in modern systems, this work demonstrates that attacking FIDO2-based authentication is far from trivial. Our experiments and analysis show that successfully compromising such systems is complex and time-consuming. In addition, multiple strict prerequisites must be fulfilled, such as physical proximity to the victim or prior compromise of the victim's device. These constraints significantly raise the bar for attackers and result in a threat model that is considerably more robust than traditional password-based authentication.

This work analysed possible attack vectors against FIDO2 authentication and demonstrated that FIDO2 largely fulfills its security promises. In particular, passkeys cannot be phished using conventional techniques since the private key never leaves the authenticator. As a result, many well-known attacks that are effective against password-based systems are rendered ineffective.

Future work includes the automation of the authenticator deception attack, as well as research into attack techniques that could target FIDO2 authentication without relying on device compromise or requiring an attacker to be within a specific physical range of the victim.

Acknowledgements

This research was conducted by Alexander Berladskyy as part of a Computer Science Research Project at Kiel University of Applied Sciences.

References

[1] APWG, *Phishing activity trends report*, APWG Report Q3 2025, Activity: July-September 2025, Published: 9 December 2025, Dec. 2025.

[2] C. Brand et al., 'Client to authenticator protocol (CTAP)', Jan. 2019. [Online]. Available: https://fidoalliance.org/specs/fido-v2.0-ps-20190130/fido-client-to-authenticator-protocol-v2.0-ps-20190130.html.

[3] N. Bindel, C. Cremers and M. Zhao, *FIDO2, CTAP 2.1, and WebAuthn 2: Provable security and post-quantum instantiation*, Cryptology ePrint Archive, Paper 2022/1029, 2022. [Online]. Available: https://eprint.iacr.org/2022/1029.

[4] Yubico, *Yubico: Security keys and authentication solutions*, https://www.yubico.com/, Accessed: 2026-03-14, 2026.

[5] FIDO Alliance, *Fido specifications*, https://fidoalliance.org/specifications/, Accessed: 2026-03-14, 2026.

[6] J. Hodges et al., 'Web authentication: An API for accessing public key credentials level 2', Apr. 2021. [Online]. Available: https://www.w3.org/TR/webauthn-2/.

[7] Mozilla Developer Network, *Web authentication api (webauthn)*, https://developer.mozilla.org/en-US/docs/Web/API/Web_Authentication_API, Accessed: 2026-03-14, 2026.

[8] P. R. Babu, D. L. Bhaskari and C. Satyanarayana, 'A comprehensive analysis of spoofing', *International Journal of Advanced Computer Science and Applications*, vol. 1, no. 6, 2010.

[9] H. Li, X. Pan, X. Wang, H. Feng and C. Shi, 'Authenticator rebinding attack of the UAF protocol on mobile devices', *Wirel. Commun. Mob. Comput.*, vol. 2020, pp. 1–14, Sep. 2020.

[10] M. Barbosa, A. Cirne and L. Esquível, 'Rogue key and impersonation attacks on fido2: From theory to practice', in *Proceedings of the 18th International Conference on Availability, Reliability and Security*, ser. ARES '23, Benevento, Italy: Association for Computing Machinery, 2023, ISBN: 9798400707728. DOI: 10.1145/3600160.3600174. [Online]. Available: https://doi.org/10.1145/3600160.3600174.

[11] T. K. Yadav and K. Seamons, *A security and usability analysis of local attacks against fido2*, 2023. arXiv: 2308.02973 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2308.02973.

[12] A. T. Mahdad, M. Jubur and N. Saxena, 'Breaching security keys without root: Fido2 deception attacks via overlays exploiting limited display authenticators', in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '24, Salt Lake City, UT, USA: Association for Computing Machinery, 2024, pp. 1686–1700, ISBN: 9798400706363. DOI: 10.1145/3658644.3690286. [Online]. Available: https://doi.org/10.1145/3658644.3690286.

[13] D. Kim, J. Shin, G. Ryu and D. Choi, 'Hipass: Hijacking ctap in passkey authentication', *IEEE Access*, vol. 13, pp. 92 086–92 101, 2025. DOI: 10.1109/ACCESS.2025.3570377.

[14] KeepassXC Team, *Keepassxc: Cross-platform password manager*, https://keepassxc.org/, Accessed: 2026-03-14, 2026.

[15] Passkeys.io, *Who supports passkeys*, https://www.passkeys.io/who-supports-passkeys, Accessed: 2026-03-14, 2026.

[16] P. Szor, *The Art of Computer Virus Research and Defense*. Addison-Wesley Professional, 2005, ISBN: 0321304543.

[17] Bitwarden, *Bitwarden password manager*, https://bitwarden.com, Accessed: 2026-03-14, 2026.

Secure-by-Design Prototyping of an IoT Access-Control System

Oliver Vainikko[1], Ulrich Norbisrath[1], Ruben Jubeh[2]
[1]Department of Computer Science
University of Tartu
Tartu, Estonia
{oliver.vainikko|ulrich.norbisrath}@ut.ee
[2]Faculty of Computer Science and Mathematics
OTH Regensburg
Regensburg, Germany
ruben.jubeh@oth-regensburg.de

***Abstract*—Rapid Internet of Things prototyping frameworks accelerate hands-on teaching and proof-of-concept development, but often normalize insecure deployments, such as plaintext messaging, shared credentials, and weak over-the-air update protection. This paper examines IoTempower, an educational framework for ESP8266 and ESP32 deployments, through an edge-local radio-frequency identification and near-field communication door access-control case study. We (i) summarize recurring security gaps observed in typical classroom-style configurations and map them to baseline consumer-device security expectations, (ii) propose a secure-by-design reference architecture that combines unique device identities, encrypted messaging, and least-privilege topic authorization, and (iii) introduce a phase-based hardening workflow with a mandatory security gate supported by collaborative peer threat mapping. Finally, we outline framework-level extensions, including secure scaffolding, automated credential generation, and embedded checklists, that make the secure path the default while preserving prototyping velocity. The approach shows that secure operation can become the expected end state of educational prototypes without sacrificing development speed.**

Keywords-IoT security; rapid prototyping; access control; threat modeling; IoT education.

I. INTRODUCTION

Rapid prototyping is a defining feature of IoT development in university programs, industrial training, and early proof-of-concept work. Teams operate under tight time constraints, assembling systems from declarative frameworks, pre-built components, and lightweight integration tools. While this accelerates experimentation, it often sidelines security; common insecure defaults and their implications are summarized in Section III. The rapid proliferation of Internet of Things (IoT) devices has increased the demand for accessible frameworks that enable rapid development and experimentation [2][3]. IoTempower addresses this need as an educational, open-source IoT prototyping framework designed to lower the barrier to building networked embedded systems. It provides students and practitioners with preconfigured tooling for device provisioning, Message Queuing Telemetry Transport (MQTT)-based communication [4], and over-the-air updates, enabling functional prototypes to be developed with minimal setup effort [5]. As can be deduced from the references, the authors of this paper are involved in the development of IoTempower and have a vested interest in its future. IoTempower is used in hands-on teaching contexts at several European universities, where its emphasis on declarative configuration and rapid iteration supports experiential learning. While this approach yields valuable practical insights, security considerations are frequently deprioritized in typical IoTempower-based developments. The central problem is how to introduce phase-based security practices into rapid IoT prototyping workflows. The goal is to do so without impairing development velocity. Specifically, we ask:

- How can a secure-by-design architecture for access control be integrated into prototyping workflows?
- Which security mechanisms can be introduced in phases to reduce developer friction?
- How can collaborative threat-mapping support security awareness in mixed-experience student teams?
- Which framework-level extensions meaningfully reduce recurring misconfigurations?

Our objective is not to require students to implement full industrial security controls from the outset. Instead, we combine architectural guidance, incremental hardening, and structured peer review so that final prototypes use encrypted communication, verifiable identities, least-privilege authorization, and protected updates—without a last-minute hardening sprint. We address the human factor directly. Collaborative threat mapping is a structured peer-review step in which student teams exchange an architecture diagram and sanitized configuration excerpts, then apply a *Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege* (STRIDE)-lite checklist to find and remediate vulnerabilities before final assessment [6]. Peer instruction and peer assessment are well-studied mechanisms for improving learning outcomes and review quality; we adapt that logic to threat modeling in small IoT systems.

The contributions of this paper are:

- A secure-by-design reference architecture for an edge-local radio-frequency identification (RFID) access-control system implemented with IoTempower.
- A phased security model that incrementally introduces controls and enforces a security gate before final integration.

- A collaborative, STRIDE-inspired threat-mapping exercise tailored to student teams.
- A set of lightweight IoTempower extensions that provide secure defaults, automated credential scaffolding, and an embedded security checklist.

The remainder of this paper is structured as follows. Section II summarizes background and related work. Section III presents the system model and threat baseline. Section IV analyzes IoTempower's current security posture. Section V introduces the secure-by-design reference architecture. Section VI describes the phased model and framework extensions. Section VII discusses limitations, and Section VIII concludes the paper.

II. Background and Related Work

Declarative IoT prototyping frameworks, such as IoTempower [5], Tasmota [8], and ESPHome [9], lower the barrier to entry for embedded development by abstracting networking, messaging, and OTA updates behind configuration files and reusable components. In educational settings, IoTempower typically bundles a gateway (often a Raspberry Pi or a dedicated laptop in the role of an edge server) and a dedicated Wi-Fi router for the (wireless) IoT network, an MQTT broker, and integration tooling, such as Node-RED [10], while student teams configure and flash individual nodes. On the device side, teams write small node configurations, and IoTempower handles boilerplate networking, MQTT publishing/subscribing, and OTA flashing [11]. The system's control logic is usually implemented in Node-RED flows, where all devices in the system can be integrated via MQTT. A persistent tension exists between rapid prototyping and robust security. When security mechanisms — certificate management, per-device credentials, broker Access Control Lists (ACLs) — are optional or introduce friction, developers routinely postpone them until "later" and then run out of time. Industry guidance captures minimum controls that should not be optional. ETSI EN 303 645 emphasizes no universal default passwords, secure communications, and secure update processes, while NISTIR 8259A summarizes baseline capabilities around device identification, configuration, data protection, logical access control, secure updates, and security state awareness. These are broad standards, but their core ideas translate directly into practical controls for an IoT access-control prototype: unique identities, encrypted MQTT links, topic access restrictions, and protected OTA updates. We use these baseline standards as device-centric references for concrete gaps; broader frameworks like NIST CSF 2.0 [12] and the IIC Industrial Internet Security Framework [13] are referenced for context, but are intentionally out of scope for a classroom prototyping method. Threat modeling offers a practical learning lens. STRIDE is a widely used taxonomy that helps teams ask: who can spoof identities, tamper with messages, deny actions (repudiation), learn sensitive data, disrupt availability, or escalate privileges [6]. For IoT prototypes, a STRIDE-lite checklist tailored to devices, topics, and trust boundaries can be effective without requiring a full formal model [14][15].

This work addresses the resulting gap: the lack of a lightweight, pedagogical, and secure-by-design methodology tailored for resource-constrained IoT systems and rapid development environments. We propose a method that embeds security practices directly into the development workflow through framework extensions and tailored development phases, thereby mitigating common security pitfalls without sacrificing the speed and exploratory nature of IoTempower and related tools.

Prior research has explored complementary aspects of secure IoT development and developer support. For example, Corno et al. [16] study how novice developers can be guided to better recognize and address security issues in cloud–IoT systems, while Zhang et al. [17] analyze security risks arising from developer-customized device–cloud interactions. These works highlight the importance of tooling and workflow design in shaping developer security practices, but focus primarily on cloud-centric architectures or individual development decisions.

Building on these insights, our work addresses a related but distinct problem space: security practices in edge-local IoT prototyping and classroom-based development workflows. We combine (i) a concrete secure-by-design access-control architecture, (ii) a phased security model tailored to rapid IoT prototyping, (iii) collaborative peer-driven threat mapping, and (iv) framework-level extensions that encode secure defaults within a single integrated approach.

III. System Model and Baseline Threats

A. Access-Control Use Case

The case study system is a local door access-control setup suitable for a lab or office environment. An RFID reader node (ESP32) scans a badge UID and publishes it to an MQTT topic (e.g., `access/request/door1`). A Local Authorization Service (LAS), implemented as a Node-RED flow or small service on the IoTempower gateway, validates the UID against an allowlist and publishes a grant-or-deny decision. A door actuator that triggers a solenoid via a relay on another node subscribes to the decision topic and triggers unlocking for a specific door. Systems may include additional nodes, e.g. for keypad access or displaying system status, as well as a cloud connection or smartphone app that can be used to administer the system using an ACL-protected dashboard. Figure 1 shows the architecture of the system using a central gateway node that also runs the required software stack to integrate the different nodes.

B. Trust Boundaries

The gateway is treated as trusted infrastructure; nodes are exposed (an attacker may steal or tamper with them). The network boundary matters because MQTT is a message bus: if an attacker can join the Wi-Fi or reach the broker, they can potentially observe or inject events. A second boundary exists

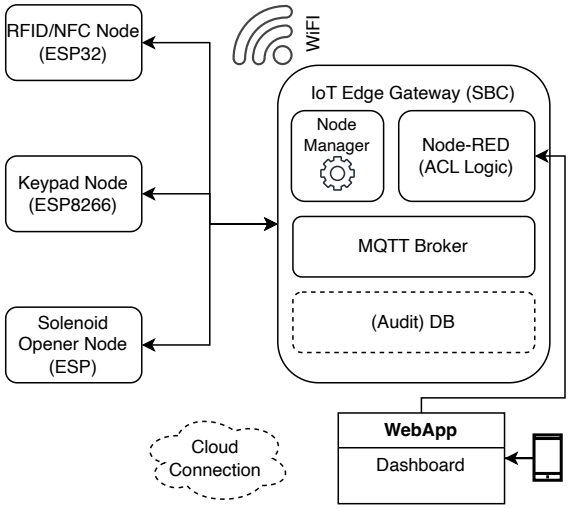


Figure 1. Access-control system architecture.

at the gateway management surface (Node-RED UI, SSH, file manager) [10].

C. Insecure Prototype Baseline

In typical early IoTempower deployments, three recurring properties are observed:

1) plaintext MQTT on port 1883 with weak or shared authentication
2) non-unique identities (all nodes share the same Wi-Fi PSK, MQTT credentials, and often the same OTA password)
3) OTA updates protected only by a password hash, with sensitive configuration embedded in firmware/flash

Table I shows a brief overview of which assets matter in this context and how they could be abused. A concrete attack scenario illustrates the impact: an attacker briefly accesses a node, dumps its flash memory, recovers Wi-Fi and MQTT credentials, joins the network, subscribes to badge and decision topics, and injects forged unlock commands. The attacker can deploy modified firmware to maintain persistence. This chain demonstrates how shared secrets and plaintext messaging collapse containment after a single device compromise.

IV. IoTempower Security Analysis

IoTempower exposes security controls, but many secrets are handled as convenience configuration. In our classroom-style IoTempower setups, a single node flash dump can often recover the Wi-Fi SSID/password, MQTT host and (optional) shared credentials, the OTA password hash (keyhash), and the MQTT/TLS CA certificate (mqtt_ca_cert). [5] As described in the insecure baseline in Section III, MQTT is frequently deployed without transport protection and strict authorization. IoTempower supports TLS and broker-side authorization controls, but they are often configured late or inconsistently in classroom-style prototyping [18]. IoTempower is secure-capable but not secure-by-default—making its gaps measurable and good targets for an incremental hardening plan. Therefore, we map the baseline prototype against key expectations from ETSI EN 303 645 and NISTIR 8259A, focusing on default credentials, secret storage, update mechanisms, transport security, and device identity. The goal is not formal compliance; it is to identify the highest-leverage gaps that a student-built access-control prototype should close first.

V. Secure-by-Design Reference Architecture

To address these weaknesses without imposing industrial-scale complexity, we propose a secure-by-design reference architecture built around three core principles: unique identity, encrypted transport, and least-privilege authorization. Our secure-by-design reference architecture addresses the baseline weaknesses by making identities, encryption, and authorization explicit design elements rather than optional configuration toggles. These elements are:

- **Unique device identity:** each node is provisioned with a unique credential (preferably an X.509 client certificate and private key). The broker authenticates clients based on these credentials, enabling per-device revocation. This breaks the "one device breach = total breach" property of shared-secret deployments.
- **TLS-only MQTT:** the broker accepts device connections only over TLS and rejects plaintext connections [19]. Mutual authentication prevents unauthorized clients on the Wi-Fi from reading or injecting MQTT messages. Even if an attacker gains Wi-Fi access, they still need valid client credentials to connect to the broker [18].
- **Least-privilege authorization:** broker-side ACLs restrict each identity's allowed topics and operations (publish vs subscribe). The LAS applies a simple policy file mapping device IDs to allowed actions. This way, a compromised reader cannot publish unlock commands, and a compromised actuator cannot subscribe to other doors' topics [18].
- **OTA and configuration hygiene:** update secrets are per-device and are not shared across the fleet. Where possible, firmware signing is introduced so that only trusted binaries are accepted. The gateway logs authentication failures and denied publishes to support auditing and to provide teachable evidence during exercises.

Topics are treated as explicit interfaces rather than "secret strings," and standardized naming conventions reinforce role separation. Access-control devices must define failure behavior. In classroom prototypes we default to fail-secure (do not unlock on missing authorization), but real deployments may require different policies for life safety and egress. These decisions should be made explicitly and tested, especially when introducing availability risks (e.g., broker outages).

VI. Phased Security Model for Rapid Prototyping

Rather than expecting teams to configure certificates and ACLs on day one, the development workflow aligns security

TABLE I. ASSETS AND ADVERSARY ACTIONS (SUMMARY).

Asset / Control Surface	Why it matters	Likely adversary actions
Node credentials (Wi-Fi, MQTT, OTA, certs)	Gate access to network, broker, and updates	Extract from flash; reuse to impersonate nodes; pivot laterally
MQTT topics (request, grant/deny, logs)	Carry badge data and unlock commands	Eavesdrop; replay; inject forged grant messages
Gateway configs (broker ACLs, LAS logic, keys)	Defines authorization and trust anchors	Abuse default access; modify allowlists/flows; steal keys
Firmware/OTA binaries	Determines device behavior and secrets in code	Upload malicious firmware; downgrade; backdoor devices
Shared Wi-Fi for all devices and gateway	Spoofing to capture all transmissions for replay attacks	Eavesdrop, replay, spoof MAC address

Figure 2. Phased security model.

upgrades with functional milestones. Early phases keep friction low, while later phases enforce security as a project requirement. The overall progression is illustrated in the phased security model timeline in Figure 2, which summarizes Phases 0–4 and marks Phase 3 as a mandatory security gate before final integration.

In Phase 0 (Bootstrap), teams bring up the gateway, MQTT broker, and initial nodes so that a minimal system is reachable. At this stage, temporary development secrets are acceptable, but the setup already enforces basic network segmentation and clear, consistent topic naming to avoid unstructured growth later.

Phase 1 (Functional prototype) focuses on achieving the end-to-end door workflow: an RFID badge read triggers an access request, the Local Authorization Service decides, and the actuator unlocks the door. During this phase, teams introduce basic hygiene measures, such as separating request and decision topics, allowing logging of access events, and removing wildcard subscriptions that would otherwise blur trust boundaries.

In Phase 2 (Identity provisioning), the system is prepared for stronger guarantees without yet forcing all traffic into TLS. Teams generate per-device certificates or credentials, precreate broker accounts, and define ACL entries so that each device identity is bound to a constrained set of topics and operations.

Phase 3 (Secure baseline) is the explicit security gate also highlighted in Figure 2: a system that "works" must now also "work securely." MQTT is switched to TLS-only operation, ACLs are enforced on the broker, and insecure fallbacks, such as plaintext listeners or shared test credentials, are removed from the configuration. Crossing this gate is a precondition for final demonstrations or grading in the intended teaching setting.

Finally, **Phase 4 (Hardening)** is an optional phase for teams that have capacity or higher assurance goals. Typical enhancements include enabling OTA firmware signing, introducing credential rotation procedures, and wiring basic monitoring or anomaly detection hooks into the system to catch suspicious behavior at runtime.

A. *Collaborative Threat-Mapping*

Technical controls alone are insufficient if developers do not internalize an adversarial mindset and understand how design decisions influence attack surfaces. To address this, the phased model integrates a dedicated collaborative threat-mapping exercise at the security gate in Phase 3, when the system has reached a secure baseline in terms of TLS, identities, and ACLs [20][21].

After a team has completed Phase 3, it prepares a concise review bundle that captures the security-relevant aspects of its design. This bundle typically includes an architecture diagram, a structured list of MQTT topics and their intended roles, selected configuration excerpts (e.g., broker configuration, Node-RED flows) that do not reveal private keys or secrets, and a short description of the implemented security controls. The bundles are exchanged such that each team reviews the system of another team, creating a peer-to-peer review setting that mirrors code review practices in professional development.

The STRIDE-lite checklist, which has been adapted for use in the context of IoT access control, will be used by reviewers. The adapted checklist is shown in Table II. Instead of posing open questions, the checklist prompts reviewers to examine concrete aspects, such as whether an attacker could impersonate a device or the Local Authorization Service (Spoofing), modify MQTT messages or authorization lists in transit or at the gateway (Tampering), or cause badge identifiers, credentials, or configuration data to be exposed in

TABLE II. STRIDE-LITE PROMPTS USED IN PEER THREAT MAPPING (ADAPTED FROM STRIDE [6]).

STRIDE category	Example prompts for IoT access control
Spoofing	Can a rogue client impersonate a device or the LAS to publish unlock commands?
Tampering	Can MQTT messages or allowlists be modified in transit or on the gateway?
Repudiation	Are door events logged so actions can be attributed to identities?
Information Disclosure	Do badge IDs, credentials, or configs leak over the network or in storage?
Denial of Service	What happens if Wi-Fi is jammed or the broker is flooded; does the door fail-secure?
Elevation of privilege	Can a low-privilege node publish admin topics or bypass LAS decisions?

cleartext (Information Disclosure). Additional prompts cover the system's behavior under denial-of-service conditions (e.g., Wi-Fi jamming or broker overload) and whether door events are logged in a way that supports attribution and detection of privilege escalation attempts [6][14][15].

The outcome of the exercise is a short vulnerability report that each reviewer team submits. Reports are expected to contain at least a small number of concrete findings, each accompanied by a proposed mitigation or rationale, which encourages reviewers to move beyond merely identifying issues towards actively reasoning about possible fixes. The original team is required to address the reported issues or provide a justified explanation for deviations before the final assessment, thus closing the loop between review and remediation. This process elevates threat modeling from a purely theoretical activity to a collaborative learning mechanism that reinforces the importance of security controls introduced in earlier phases.

B. IoTempower Extensions for Secure Defaults

While the phased model structures when security controls are introduced, developers still benefit from tooling support that reduces the effort of "doing the right thing." To that end, the approach proposes a set of lightweight extensions to IoTempower that encode secure defaults and automate repetitive tasks, making the secure path the default.

First, project scaffolding can ship with preconfigured secure communication templates. A default broker configuration enables TLS, disables anonymous access, and includes an example ACL file [18] that reflects the separation of roles in the access-control case study (e.g., readers, actuators, Local Authorization Service, and administrative clients). This shifts the default away from plaintext MQTT with permissive access towards a configuration that already aligns with the secure baseline described in Phase 3.

Second, credential management can be supported through an integrated command-line utility that automates the generation of key material. A single command can create a local certification authority for the classroom or lab setting, generate broker and server certificates, and provision per-device client credentials along with a mapping file that associates logical device identifiers with their respective credentials. This reduces the likelihood of teams falling back to shared secrets or ad-hoc credential schemes purely for convenience.

Third, the project repository can embed a structured security checklist and a threat-mapping worksheet that are explicitly aligned with the development phases. Tooling can be extended to remind teams at certain milestones—for example, when moving from Phase 1 to Phase 2 or approaching the Phase 3 gate—to complete the relevant checklist items and prepare review materials. This integration keeps security tasks visible within the normal development workflow, instead of relegating them to separate documentation that students may overlook.

Finally, a "classroom mode" can give instructors controlled flexibility over which protections are enforced at which time. For example, temporary insecure settings can be selectively enabled when demonstrating specific attacks or failure modes, while the default configuration remains aligned with the secure baseline and the phased progression. Together, these extensions operationalize the phased model by minimizing configuration friction, encouraging consistent application of best practices, and enabling instructors to steer the trade-off between usability and enforcement in a transparent way.

C. Preliminary Observations and Evaluation Plan

Across prior course iterations, recurring issues such as plaintext MQTT, shared credentials, and over-permissive topic access have been observed. These observations motivated introducing Phase 3 as an explicit security gate. We plan to evaluate the approach in an upcoming course iteration (approximately 20 students across five teams). Quantitative metrics could include the number and severity of vulnerabilities identified during peer review and the quality of implemented fixes; this data will be collected across multiple courses. Qualitative data will be collected through surveys and interviews focusing on perceived development friction, security awareness, and the usefulness of the framework extensions.

VII. DISCUSSION AND LIMITATIONS

The proposed approach carefully balances usability and security enforcement, particularly critical in educational settings where classroom prototypes prioritize rapid development over production-grade robustness. The phased security model enables teams to achieve functional prototypes early (Phases 0–1) before progressively layering on controls at defined milestones, culminating in the mandatory Phase 3 security gate. This structure minimizes developer friction by deferring resource-intensive tasks, such as certificate provisioning, until the system is stable, while ensuring that insecure fallbacks cannot persist into final demonstrations.

The threat coverage of the reference architecture and associated exercises focuses primarily on MQTT-related threats, such as spoofing, tampering, and lateral movement through least-privilege topic authorization. While this addresses the most frequent vulnerabilities observed in IoTempower deployments (e.g., plaintext communication and shared credentials), it does not fully mitigate denial-of-service risks (e.g., broker overload or Wi-Fi jamming), supply-chain compromises during firmware updates, or sophisticated physical attacks on exposed nodes. These gaps are intentional, given the lightweight, pedagogical scope, but they highlight opportunities for extension in future iterations.

Even in a local-only deployment, security is not guaranteed by isolation alone. Exposed management interfaces, such as the Node-RED UI, SSH access to the gateway, or an open OTA port (port 8266), can significantly expand the effective attack surface if not explicitly hardened. Teams must document their secure setup comprehensively and verify it through measurements (e.g., network traffic analysis or credential extraction attempts from flash dumps) to confirm that the intended protections are operational.

The reliance on peer-driven threat mapping in Phase 3 introduces variability depending on the experience levels within student teams. While STRIDE-lite checklists and review bundles standardize the process, uneven threat modeling quality could lead to overlooked issues in some designs. Mitigating this requires instructor calibration of reviews and iterative refinement of checklist prompts based on empirical observations from course deployments.

VIII. Conclusion and Future Work

This work translates established IoT security practices into an educational rapid-prototyping workflow that is feasible for novice teams. The key contribution is a practical integration of secure defaults, a phased security gate, and collaborative threat mapping that makes secure operation the expected end state of classroom prototypes. Future work includes empirical evaluation at scale, tighter integration of per-device credential provisioning and revocation, and support for signed firmware updates. Upstream integration of the proposed IoTempower extensions into the core framework through open-source contributions will establish secure defaults as canonical best practices across educational deployments.

References

[2] J. P. Dias, F. Couto, A. C. Paiva, and H. S. Ferreira, "A brief overview of existing tools for testing the internet-of-things", in *2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2018, pp. 104–109. DOI: 10.1109/ICSTW.2018.00035.

[3] R. P. Machado, U. Norbisrath, and R. Jubeh, "Iot educational framework case study: Devices as things for hands-on collaboration", *Journal of Engineering Education Transformations*, vol. 37, no. Special Issue 2, pp. 287–295, 2024. DOI: 10.16920/jeet/2024/v37is2/24045.

[4] *Mqtt version 3.1.1*, Publish/subscribe messaging protocol standard, OASIS, 2014.

[5] U. Norbisrath and IoTempower Community, *IoTempower: A framework and environment for making the internet of things (IoT) accessible for everyone.* Version 0.9.5, [retrieved: March, 2026]. [Online]. Available: https://github.com/iotempire/iotempower.

[6] Microsoft, *Stride threat modeling*, https://learn.microsoft.com/en-us/security/engineering/threat-modeling, [retrieved: March, 2026].

[8] Tasmota Project, *Tasmota firmware documentation and project site*, https://tasmota.github.io/docs/, [retrieved: March, 2026], 2025.

[9] ESPHome, *Esphome documentation*, https://esphome.io/, [retrieved: March, 2026].

[10] OpenJS Foundation, *Node-red documentation*, https://nodered.org/docs/, [retrieved: March, 2026].

[11] Arduino Project and Espressif Systems, *Arduino ota / esp8266 and esp32 ota update documentation*, https://arduino-esp8266.readthedocs.io/en/latest/ota_updates/readme.html, [retrieved: March, 2026].

[12] National Institute of Standards and Technology, "The nist cybersecurity framework (csf) 2.0", NIST, Tech. Rep. CSWP 29, 2024, [retrieved: March, 2026].

[13] *Industrial internet security framework*, version 2.0, [retrieved: March, 2026], Industrial Internet Consortium, 2023.

[14] A. Shostack, "Experiences threat modeling at microsoft", in *MODSEC@MoDELS*, 2008.

[15] L. Kohnfelder and P. Garg, *The threats to our products*, Microsoft internal threat modeling document introducing STRIDE, 1999.

[16] F. Corno, L. De Russis, and L. Mannella, "Helping novice developers harness security issues in cloud-iot systems", *Journal of Reliable Intelligent Environments*, vol. 8, no. 4, pp. 601–616, 2022. DOI: 10.1007/s40860-022-00175-4.

[17] Y. Zhang, J. Li, and D. Gu, "Rethinking the security of iot from the perspective of developer customized device-cloud interaction", in *Proceedings of the ACM/SIGAPP Symposium on Applied Computing (SAC '22)*, 2022, pp. 1070–1077. DOI: 10.1145/3477314.3508389.

[18] Eclipse Foundation, *Eclipse mosquitto: Mqtt broker documentation*, https://mosquitto.org/documentation/, [retrieved: March, 2026].

[19] E. Rescorla, "The transport layer security (tls) protocol version 1.3", IETF, Tech. Rep. RFC 8446, 2018.

[20] T. Chothia and J. de Ruiter, "Learning from others' mistakes: Penetration testing iot devices in the classroom", in *USENIX Workshop on Advances in Security Education (ASE)*, 2016.

[21] M. Hamad, A. Finkenzeller, M. Hasan, M.-O. Pahl, and S. Steinhorst, "A gamified learning approach for iot security education using capture-the-flag competitions: Architecture and insights", in *NordSec*, 2024, pp. 147–165.

Closing the Temporal Gap: A Deterministic Simulation Framework for Physics-Aware Automotive Intrusion Detection

Liron Ahmeti*, Klara Dolos*, Conrad Meyer*, Andreas Attenberger*, Sebastian Fischer† Rudolf Hackenberg‡
*Research Unit, Central Office for Information Technology in the Security Sector
Munich, Germany
Email: poststelle@zitis.bund.de
†Dept. Informatics and Mathematics, OTH Regensburg
Regensburg, Germany
Email: sebastian.fischer@oth-regensburg.de
‡Dept. Informatics and Mathematics, OTH Regensburg
Regensburg, Germany
Email: rudolf.hackenberg@oth-regensburg.de

***Abstract*—Validating Intrusion Detection Systems (IDS) for autonomous vehicles requires simulation environments that are not only visually realistic but forensically deterministic. Current real-time simulators often couple network arbitration to rendering framerates, introducing non-deterministic timing artifacts (jitter) that obscure the signatures of cyber-attacks, such as bus flooding or replay injections. This paper presents SimDAT-AV, a simulation framework designed to close this temporal gap. By decoupling simulation time from host execution speed via a Synchronous Lockstep architecture, we achieve bit-exact reproducibility of Controller Area Network (CAN) bus traffic (validated via SHA-256 checksums), enabling the precise reconstruction of attack scenarios regardless of computational load. Furthermore, we introduce a Physics-Aware IDS that leverages a high-fidelity Virtual Powertrain Model to detect semantic anomalies. We demonstrate that correlating internal vehicle states (e.g., engine engine Revolutions Per Minute (RPM) vs. wheel speed) allows for the detection of sophisticated spoofing attacks that satisfy protocol-level checks but violate physical consistency. The proposed framework eliminates execution jitter (collapsing timing-induced variance to a single deterministic outcome), providing a robust foundation for verifying automotive security mechanisms under reproducible, forensic conditions.**

Keywords-Automotive Security; Intrusion Detection; Digital Forensics; Deterministic Simulation

I. Introduction

Safety validation in autonomous driving is not a monolith. While kinematic simulation excels at testing planning logic, for security validation—particularly for Intrusion Detection Systems (IDS)—it often solves the wrong problem. In a forensic setting, the decisive evidence is rarely the vehicle's trajectory. Instead, it is the integrity of the underlying signals: bus arbitration timing, message inter-arrival patterns, and the causal consistency between physical measurements and digital messages [1]. Current validation approaches force a trade-off. Physical testing captures essential sensor variability, but safety regulations prevent the systematic injection of severe cyber-attacks on public roads. Simulation removes this constraint, yet standard real-time environments, such as CARLA [2] or AirSim [3], typically abstract away the very effects an IDS monitors. Because many detectors rely on micro-timing regularities as their definition of 'normal,' even small simulation-induced distortions become indistinguishable from attacks. In many simulators, critical forensic artifacts—clock drift, arbitration latency, weather-driven signal degradation—are either smoothed out or become entangled with the host computer's performance [4]. For a detector monitoring inter-arrival times, this breaks the baseline; the distribution of normal traffic shifts simply because the GPU is busy. This mismatch creates a practical reality gap between synthetic traces and vehicle logs. Trained on such regularized data, models learn decision boundaries that look crisp because the world they see is crisp. On the road, that crispness vanishes: rain attenuation can be flagged as malicious, while a protocol-correct spoofing attack might look perfectly normal. To bridge this gap, we introduce *SimDAT-AV*, a simulation framework engineered for forensic readiness, i.e., the capability to produce data that supports causal reconstruction and allows observed anomalies to be attributed to specific causes rather than simulator artifacts. Where others pursue visual realism, we prioritize two properties essential for forensics: explicit parametric modeling of degradations and tick-synchronous determinism, meaning that state evolution is strictly decoupled from wall-clock time so that rerunning a configuration with a fixed seed yields bit-identical logs independent of host performance. This lets us rerun a scenario with the exact same rain profile and determine whether an IDS alarm tracks the weather or only appears when we inject an attack. By doing so, artifacts become reproducible and attributable, allowing us to isolate an alarm's true cause instead of blaming simulator noise.

A. Motivation

Our work confronts three specific deficits in current simulation technology. For a simulator to be forensically sound, it must answer three questions: Does it reproduce realistic signal variance? Does it preserve deterministic timing? And can it expose causal ground truth?

1) Inadequate signal variance in synthetic data: Statistical intrusion detection works by learning the envelope of normal operation; that envelope is the detector's definition of reality. Most simulators approximate it using simple additive Gaussian

noise, but for safety-critical sensors, this is a poor substitute. Holder et al. [5] demonstrated that simple radar noise models fail to reproduce the coherent clutter distributions seen in physical sensors, and LiDAR attenuation behaves nonlinearly with rain intensity [6]. If 'normal' is modeled incorrectly, anomaly detection degrades into a calibration error, not a security guarantee [7]. An IDS validated on 'perfect attacks + white noise' may succeed in simulation, only to fail when faced with structured, weather-dependent noise in the real world.

2) Lack of temporal determinism: Forensic interpretation hinges on precise timing; it is often the only clue separating benign system jitter from a malicious attack. Yet many real-time simulators implicitly couple their simulation stepping to rendering performance. As the GPU is busy or CPU load fluctuates, scheduling jitter appears—jitter that is often indistinguishable from an attack's timing effects. This is a fundamental failure mode for any detector that monitors message Inter-Arrival Times (IAT). A forensic simulator must therefore guarantee tick-synchronous determinism, replaying scenarios bit-identically from a fixed seed, regardless of host system load.

3) Weak causal attribution without internal vehicle states: Without ground truth, it is difficult to determine whether an Autonomous Emergency Braking (AEB) event was triggered by a software fault, a phantom return from weather, or malicious CAN injection. To resolve this uncertainty, the simulation must expose its causal structure, not just the final output. IDS logic requires access to the evolution of internal physical states. Without it, a physically impossible bus trace can be labeled benign simply because the simulator lacks the vocabulary to express the violation. We therefore constrain bus signals by physically reachable state trajectories such as pressure build-up limits and drivetrain dynamics.

B. Problem Definition and Research Objectives

The central problem is the absence of a simulation environment built for strict forensic requirements: deterministic replay, physically meaningful timing, and controllable sensor degradation. To close this gap, we engineer the simulator around these constraints rather than real-time rendering. In this context, the actuation gap denotes the physical latency between a logical command, such as a brake request, and the corresponding mechanical response, such as pressure build-up; we use this irreducible latency floor as a discriminator to distinguish physically plausible actuation from instantaneous spoofing signals. The following three objectives are our minimum requirements for making IDS validation genuinely falsifiable.

1) Spectrally Consistent Vehicle Dynamics: We start with dynamics, because plausibility checks are only meaningful if the underlying physical state is credible. Game-engine kinematics often smooth out high-frequency transients that form the operational signature of a vehicle, such as drivetrain oscillations or shift shock. A gear shift, for example, leaves a characteristic burst in the torque signal that a detector can learn to recognize. If we validate against a model that suppresses these cues, the IDS learns to verify physics against a physics-free world. Accordingly, our dynamics model reconstructs internal states (e.g., engine speed, torque, hydraulic pressure) to ensure the resulting CAN signals exhibit spectral characteristics consistent with physical vehicle logs.

2) Parametric, Deterministic Sensor Degradation: Visually convincing data from neural renderers often comes at the cost of control, making counterfactual analysis impossible. An analyst must be able to ask: "Would this attack still succeed if rain intensity were 10 mm/h lower?" To answer this, we implement phenomenological Radar and LiDAR models where effects like attenuation and clutter become controlled levers for such counterfactual tests [8] [9]. This allows us to vary one condition at a time—rain, fog, or an attack—and know precisely which change triggered an IDS response.

3) Cross-Domain Validity of Security Mechanisms: Our final acceptance criterion is cross-domain performance. An intrusion detection system trained exclusively on *SimDAT-AV* data must achieve a comparable True Positive Rate (TPR) and latency robustness when evaluated against real-world datasets, such as ROAD [10] and RADIATE [11]. This demonstrates that the synthetic data is useful not just in theory, but in practice.

C. Overview

The remainder of this paper is structured as follows. Section II reviews existing work. Section III details the *SimDAT-AV* architecture and its synchronous lockstep mechanism. Section IV describes the cyber-physical modeling approach. Section V verifies simulation fidelity. Section VI evaluates forensic utility and IDS transferability. Finally, Section VII discusses limitations, and Section VIII concludes the paper.

II. Related Work

Validation methodologies rely on synthetic data to bridge the gap between nominal driving and edge cases. While datasets like nuScenes [12] or KITTI [13] provide high-fidelity sensor data, they often lack the intersection of adverse weather and active cyber-attacks. Recent work by Dološ et al. [14] emphasizes *Forensic Readiness* in autonomous mobility, but its validation implicitly assumes a simulation environment with deterministic timing and physically grounded signal generation. Existing approaches frequently trade forensic controllability for visual fidelity or statistical convenience. We therefore organize related work along three axes: temporal determinism, signal realism, and data generation paradigm.

A. Temporal Determinism and Reproducibility

Forensic analysis depends on deterministic reconstruction. Many standard simulation frameworks undermine this by tying simulation speed to rendering frame rates [2]. The resulting host-dependent jitter creates uncertainty over whether delayed CAN frames stem from an attack or from rendering load. A simulator that cannot guarantee bit-identical replay offers no stable baseline for timing-based IDS features.

B. Signal Variance and Environmental Noise

A second failure point occurs when 'normal' is mis-modeled. This happens frequently when sensor noise is approximated with

 ISBN: 978-1-68558-442-9

simple additive Gaussian distributions rather than structured, temporally correlated weather effects. Holder et al. [5] showed that basic radar noise models fail to reproduce coherent clutter, while Linnhoff et al. [6] found that LiDAR attenuation depends non-linearly on rain intensity.

C. Paradigms of Synthetic Data Generation

Forensic integrity also depends on the data generation paradigm itself. Attack injection without physically consistent signal propagation can produce traces that are protocol-correct but causally implausible. Frameworks such as SimuTack [15] and DriveFI [16] enable attack injection, but often focus on the logical layer rather than the physical propagation needed for forensic plausibility.

a) Statistical Models: Methods like CTGAN [17] scale well for tabular augmentation but cannot represent protocol timing. They are therefore useful for static risk assessment, but not for validating stateful timing-sensitive IDSs.

b) Implicit vs. Explicit Modeling: Neural simulation systems such as UniSim [18] and SHIFT [19] achieve high visual fidelity, but often rely on implicit latent representations that hinder counterfactual analysis. Forensic evaluation instead requires explicit parametric knobs so that a single changed factor can be linked to a changed IDS response.

III. Architecture for Forensic Determinism

Standard game engines operate on a heuristic that conflicts with forensic requirements: they prioritize visual fluidity over temporal precision. To keep the display smooth, an engine will often decouple physics steps or drop frames, effectively injecting host-dependent jitter into both the rendering and the network event stream. Under load, this makes it difficult to disentangle host-induced artifacts from malicious timing manipulations. To establish a forensically sound ground truth, we must remove the host system as a hidden variable. We achieve this by stepping outside the real-time paradigm and employing an architecture based on Synchronous Lockstep and Inverted Control to ensure the simulation behaves strictly as a deterministic function of its model parameters.

A. Inverted Control Strategy

Lockstep alone is insufficient if the simulator still owns the clock; the engine might still interpolate or skip work under load, forcing clients to chase a moving time base. We therefore invert the traditional control hierarchy. Where a standard setup has the simulator (e.g., CARLA) dictating the timeline, we assign timing authority to a central *Supervisor* module. This demotes the simulator to a passive backend. At any given tick, the Supervisor requests a world snapshot, executes all subsystem updates, and holds the clock until completion. The simulation advances only because computation for the current step, t_k, is finished, not because a certain amount of real time has passed. We utilize a fixed logical step of 20 ms (50 Hz) to align with standard Electronic Control Unit (ECU) cycle times.

1) Formal Scheduling Model: A fixed timestep is inadequate as long as subsystems can lag behind asynchronously. The core architectural challenge is preventing a slow module from being skipped or approximated when the host is under stress. We model the Supervisor as a state machine cycling through REQUEST, COMPUTE, and COMMIT states. The invariant guaranteeing determinism is the condition for advancing time: the system transitions to the COMMIT state if and only if every active client has explicitly acknowledged completion of the current step, t_k. We formalize this dependency as:

$$S_{next} = \text{COMMIT} \iff \forall c \in C_{clients} : \text{Ack}(c, t_k) \quad (1)$$

The practical effect of this barrier is that system overload increases the wall-clock time per tick but preserves the sequence and content of events within those ticks. Modules do not rely on wall-clock timestamps; the shared monotonic tick counter serves as the sole time base. If the host stalls, the virtual clock pauses rather than skipping ahead. This effectively converts overload into latency while maintaining causal correctness, as argued by Kopetz for real-time systems [20]. The empirical validation of this mechanism is presented in Section V.

B. Synchronous Execution Control and Virtual Arbitration

This barrier-based scheduling model necessitates a synchronous execution flow. The Supervisor coordinates sensors, the Virtual Forensic Bus, and recording modules through blocking synchronization. Crucially, this extends to network arbitration. To satisfy conflicting requirements (hardware compatibility vs. reproducibility), we introduce a Dual-Mode Interconnect Architecture:

- **Live Mode:** Utilizes standard Operating System (OS)-level SocketCAN interfaces. This preserves the stochastic behavior of the kernel scheduler, which is necessary when interfacing with physical ECUs (Hardware-in-the-Loop).
- **Forensic Mode:** Bypasses the OS kernel entirely using a user-space broadcast hub. This mode enforces strictly sequential packet delivery governed solely by the logical tick.

For the validation of forensic determinism presented in this paper, the system operates strictly in *Forensic Mode*. This ensures that packet ordering is defined by the simulation step, preventing event reordering and data loss that occur in asynchronous environments under adversarial workloads.

C. High-Performance Data Ingestion

With a deterministic timeline, the data serialization process itself becomes the next potential source of error. Forensic analysis depends on precise residuals—the mathematical differences between predicted and observed states. Text-based formats like CSV are unsuitable for this task because conversion between decimal strings and binary floating-point values is not guaranteed to be round-trip stable. This can introduce rounding artifacts that, in a residual-based IDS, could be misclassified as anomalies. To eliminate this risk, our setup mandates lossless floating-point precision and columnar access. We log directly

to Apache Parquet with Snappy compression. This ensures the storage layer cannot perturb the data and prevents logging from becoming a bottleneck. This setup satisfies two mandatory forensic criteria:

- **Binary Fidelity:** Sensor channels are stored as typed float32 columns, preserving IEEE-754 values without decimal conversion.
- **State Separation:** Ground-truth columns from the simulator API are explicitly segregated from perceived columns generated by the sensor model.

This allows any observed error to be attributed entirely to the sensor model or an injected attack, rather than to serialization artifacts.

IV. Modeling Cyber-Physical Systems

To validate an IDS, the simulation must expose not only the final vehicle trajectory but also the internal states of the electronic control units (ECUs) and the network arbitration layer. An anomaly often hides not in where the car moves, but in the causal chain that explains *how* it achieves that motion. We therefore model the system layer by layer, starting with the physics of actuation.

A. Virtual ECUs: Powertrain & Transmission

We move beyond simple kinematic bicycle models to include high-fidelity Virtual ECUs. As shown in Figure 1, we implement a backward-facing kinematics model because the scenario engine dictates the vehicle's target trajectory. By inverting physical causality, the model derives engine RPM and torque from vehicle velocity and current gear ratio. This inversion is critical for IDS validation, giving us a physically consistent reference signature against which CAN traces become falsifiable. Without it, physically impossible torque/RPM jumps can still produce the correct trajectory, leaving an IDS with no violation to detect.

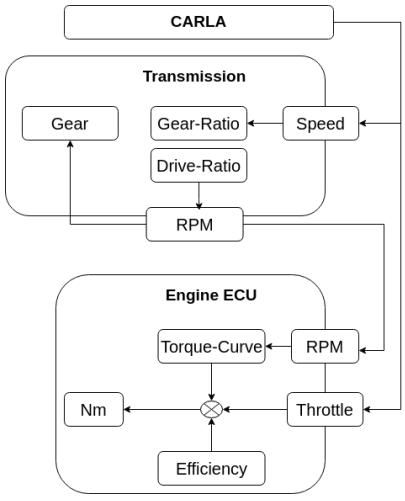


Figure 1: Data flow of the Virtual Powertrain Model.

1) Propulsion Dynamics: The engine speed ω_{eng} is derived kinematically from vehicle velocity and gear ratio. This kinematic link alone, however, leaves a gap that spoofing attacks can exploit. An attacker can craft a protocol-correct message that still violates the engine's feasible torque envelope. To close this gap, we add a second validation step using a parametric efficiency model. It synthesizes the necessary torque using a nonlinear lookup table based on engine speed and throttle position α. This forces the model to account for the physical cost of acceleration, making attacks that fake instantaneous torque changes detectable as violations against the engine's known performance map.

2) Transmission Logic: Standard simulations often rely on static shift maps, resulting in perfectly predictable gear changes. For a physics-aware IDS, however, such regularity is not to be smoothed away; it is part of the signature that helps separate plausible driver behavior from overly clean, spoofed signals. To replicate this, the transmission logic solves a cost minimization problem to select the optimal gear g^*:

$$g^* = \arg\min_{g \in G} \left(w_1 \cdot P_{eff}(g) + w_2 \cdot P_{smooth}(g) \right) \tag{2}$$

where P_{eff} represents fuel efficiency penalties and P_{smooth} penalizes frequent shifting, with weights w_1, w_2 calibrated to match observed shift rates in vehicle logs. The resulting behavior includes realistic gear oscillation and shift-shock artifacts, providing a richer baseline for an IDS to learn.

3) Hydraulic Brake Dynamics (Actuation Gap): In forensic reconstruction, physical delay is often a key discriminator. Kinematic models assume instantaneous deceleration, but real brakes are bound by fluid dynamics. We model brake hydraulics to enforce an irreducible actuation latency, making an instant-stop trace testably implausible. Simulating gradual pressure build-up in the master cylinder alongside fluid compression and valve flow limits creates this mandatory physical latency in the simulation loop.

B. Network Topology and SecOC

We simulate the CAN bus with non-destructive bitwise arbitration because many IAT-based detection features depend on the micro-delays that arbitration introduces, which simpler bus models erase. To support analysis of authenticated traffic, we integrate a *Secure Onboard Communication* (SecOC) model. A common failure mode in replay experiments is counter desynchronization, where volatile freshness counters drift across runs, causing Message Authentication Code (MAC) failures that look like attacks even when the traffic is benign. To fix this, we decouple the Freshness Value (FV) from volatile state and bind it directly to the simulation's monotonic tick, t_k:

$$FV(t_k) = \text{Hash}(\text{Seed} \parallel t_k) \tag{3}$$

By anchoring the FV to this rigid time-step, MAC verification becomes deterministic relative to the simulation clock, provided the replay preserves the exact tick sequence.

C. Physics-Aware Intrusion Detection

With this deterministic physical baseline established, we can deploy a specification-based IDS. The system enforces four rule sets that together cover the minimum needed for falsifiable plausibility: Periodicity (timing regularity), Range (field bounds), Burst (transient structure), and Kinematic Consistency (cross-signal coupling). This methodology adopts the reliability assessment principles proposed by Lohre et al. [21] for unmanned aircraft systems. Specifically, we implement their concept of Sensor Cross-Validation by correlating logically dependent states. Instead of relying on a single source of truth, we validate the trustworthiness of the CAN bus data by checking the coherence between interacting physical components. While the first three check for low-level violations, the kinematic rule serves as the central physical constraint against more sophisticated spoofing.

This final rule posits that the engine speed must remain mathematically coupled to the wheel speed and the current transmission ratio:

$$|\omega_{eng} - (\omega_{wheel} \cdot i_{gear})| \leq \epsilon \tag{4}$$

An IDS applying this rule can therefore detect attacks where an adversary injects a valid RPM signal (correct protocol, correct checksum) that contradicts the vehicle's physical velocity—an anomaly that purely digital firewalls would miss.

V. Verification of Simulation Fidelity

Before evaluating forensic utility, we must confirm that the simulation satisfies its two core requirements: bit-exact reproducibility (causal invariance) and physical plausibility at the dynamics level.

A. Bit-Exact Reproducibility

Forensic utility hinges on the guarantee that a scenario S executed with seed σ produces an identical set of artifacts across repeated runs, regardless of host computational load. To validate this, we executed the same urban scenario twice in sequence on the same host.

a) Experimental Setup:: Run A (Baseline) was executed under nominal system load with seed $\sigma = 42$. Run B (Stressed) used the same seed while the host CPU was saturated via `stress-ng` to induce wall-clock delays and thread-scheduling jitter.

We verified bit-exactness by calculating the SHA-256 checksums of the recorded sensor data streams (velocity, Global Navigation Satellite System (GNSS), CAN logs) and by computing the pointwise delta between the time-series.

b) Results:: As illustrated in Figure 2, despite the large difference in wall-clock execution time, the artifacts from Run B remained bit-exact to the baseline. Both runs produced the identical SHA-256 hash (see plot legend), confirming binary integrity. Moreover, the delta $\Delta = |Run_A(t) - Run_B(t)|$ remained strictly at 0.0 m/s for every simulation tick. This confirms that the proposed architecture (Synchronous Lockstep coupled with the Virtual Forensic Bus) successfully decouples the generation of forensic evidence from host performance, eliminating the "Observer Effect" common in real-time simulators.

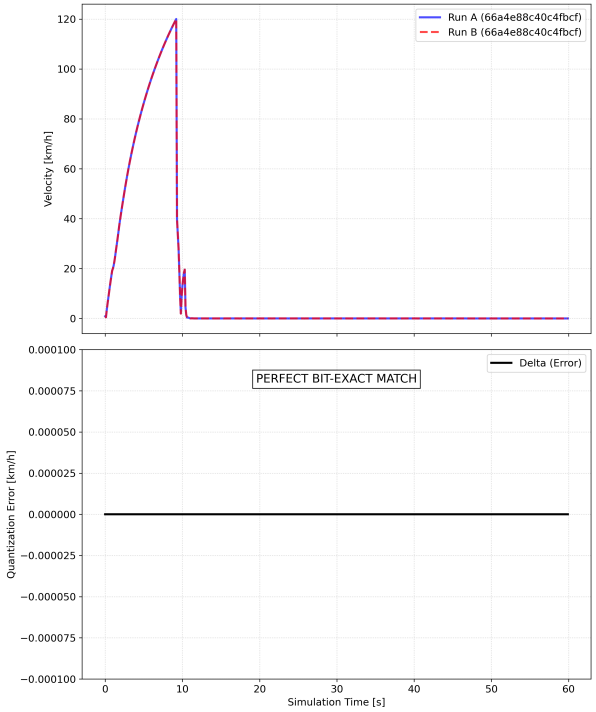


Figure 2: Verification of Causal Invariance.

B. Physical Validity: Dynamics & Latency

Beyond byte-level identity, the simulation must adhere to physical constraints to serve as a ground truth for an IDS. We focus on braking dynamics because they are both safety-critical and a common attack surface where naive spoofing attacks fail due to missing actuation latency. We validated the hydraulic brake model against real-world traces from $N = 19$ distinct braking events. The events were aligned at the brake-command onset to compare pressure build-up dynamics. The simulated pressure response tracks the statistical profile of the physical vehicle (Figure 3) with a Root Mean Square Error (RMSE) of 0.179, where pressure is normalized by the peak value of each event. This result rules out the "instant-stop" artifacts common in kinematic simulators and establishes a validated baseline where implausible braking traces (e.g., impossible pressure gradients) remain falsifiable.

VI. Forensic Utility & Security Evaluation

Having verified the simulation's determinism and physical validity, we now evaluate its specific utility for security analysis and intrusion detection.

A. Quantifying the "Temporal Gap": Async vs. Sync

To demonstrate why standard asynchronous simulation is insufficient for security validation, we conducted a test using a *Ghost Target Injection* attack. An adversary injected a false obstacle into the radar stream to trigger an Autonomous Emergency Braking (AEB) maneuver. We define reaction latency as

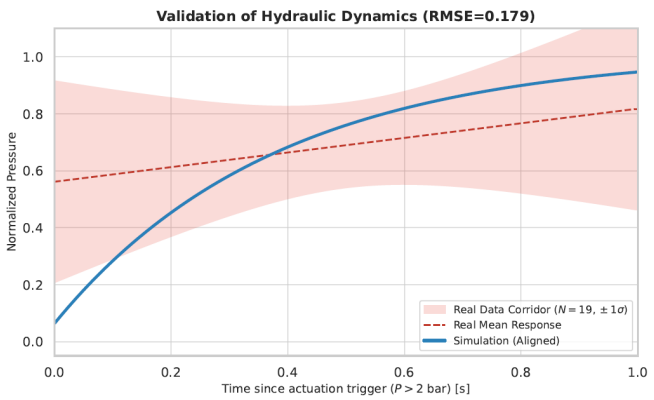


Figure 3: Statistical validation of the Hydraulic Brake Dynamics (N = 19 events).

the time from injection onset to the first AEB CAN command. We compared N = 50 runs in standard Asynchronous Mode against N = 50 runs in our Synchronous Lockstep Mode.

TABLE I: IMPACT OF EXECUTION MODE ON REACTION STABILITY (N = 50)

Metric	Async (Baseline)	Sync (Ours)
AEB trigger rate	100%	100%
Mean latency (μ)	17.53 s	17.53 s
Standard deviation (σ)	2.26 s	0.00 s
Range (min–max)	15.63–26.74 s	17.53 s

The results (Table I) reveal a critical forensic discrepancy. In the asynchronous baseline, scheduler-induced jitter caused a reaction time variance of σ = 2.26,s, with a spread of over 11 seconds between the fastest and slowest reaction to the identical attack script. In a forensic context, this noise makes it impossible to distinguish attack-induced delays from simulation lag.

In contrast, the synchronous mode collapsed this variance to a single deterministic outcome (no brake event). Consequently, the AEB command was consistently withheld in all 50 trials, proving that the erratic braking in the asynchronous mode was merely an artifact of scheduling jitter. This stability is a prerequisite for forensic analysis, as it ensures that any observed timing deviation in a trace is attributable to the attack vector, not the simulation engine's scheduler.

B. IDS Transferability (Sim-to-Real)

Finally, we assessed whether the physics-aware data improves the robustness of an IDS in real-world scenarios. We trained two versions of a specification-based IDS and evaluated them on the CAN spoofing scenarios within the ROAD dataset [10]. Two IDS variants are evaluated. The Baseline IDS is trained on idealized synthetic data with perfect timing behavior. In contrast, the SimDAT-AV IDS is trained on synthetic data generated by our framework, including physically modeled powertrain dynamics and environment-induced sensor noise. The kinematic consistency rule's tolerance (ϵ) was set for each IDS to the 99th percentile of benign timing residuals in its respective training set. On the ROAD evaluation set, the SimDAT-AV-trained model achieved a 100% True Positive Rate (TPR). Its tolerance threshold, derived from physically consistent training data, settled at 0.53,s, a value that closely matched the real vehicle's observed timing variability. The baseline model, in contrast, derived a tighter, artificial tolerance of 0.42,s from its overly clean training data. Consequently, it misclassified normal environmental noise in the ROAD logs as attacks (i.e., produced false positives). This indicates that integrating physical constraints in the data generation process prevents the IDS from overfitting to idealized simulation artifacts.

VII. Discussion & Limitations

While the synchronous lockstep architecture guarantees the evidential stability required for forensics, this stability comes at a cost. We trade nondeterminism for reduced execution throughput and higher sensitivity to model parameters.

A. The Cost of Determinism: Real-Time Factor

Lockstep inherently serializes execution: the clock cannot advance until the slowest component finishes its step. In complex scenarios, this often drops the Real-Time Factor (RTF) below 1.0. Consequently, *SimDAT-AV* is designed for *offline* tasks such as forensic replay and model training, rather than Hardware-in-the-Loop (HiL) integration with rigid wall-clock deadlines. For forensics, logical causal consistency matters more than wall-clock time.

B. Sensitivity to Physical Parameters

The utility of our kinematic consistency checks depends heavily on parameter accuracy. Such checks are only meaningful if the modeled friction coefficients and hydraulic time constants (Section IV) match the real vehicle and road surface. In our evaluation, we relied on static calibration. Future work must therefore focus on automated system identification from vehicle logs.

C. Scope of Attacks

Our current evaluation focused on spoofing and timing attacks. We demonstrated that enforcing physical plausibility is effective against these vectors. However, attacks on scene semantics, such as adversarial patches on traffic signs that confuse the camera classifier without violating kinematic constraints, remain out of scope for the current physics-aware IDS and require complementary defense layers.

D. Kernel Abstraction vs. Stack Fidelity

A deliberate architectural trade-off involves the abstraction level of the network stack. In "Live Mode" (using SocketCAN), the simulation includes stochastic OS-kernel behavior needed for hardware integration and interrupt-level timing tests, but this precludes reproducibility. In "Forensic Mode" (Virtual Bus), we bypass the kernel to achieve the zero-jitter results shown in Section V, at the cost of abstracting away kernel-space race conditions and driver-specific buffer overflows. *SimDAT-AV* therefore allows analysts to choose OS-fidelity or forensic precision, but not both simultaneously.

VIII. Conclusion and Future Work

For forensic replay, temporal determinism is a prerequisite. Our tests showed that standard best-effort architectures produce load-dependent timing shifts that can mimic cyber-physical attack signatures. We address this by decoupling simulation time from host performance through synchronous lockstep and a user-space virtual bus. Under identical seeds and induced CPU load, the resulting telemetry remained byte-identical. This makes each CAN frame a reproducible function of the model parameters rather than a byproduct of host scheduling. The resulting determinism enables a physically grounded IDS that can detect spoofing traces inconsistent with wheel-speed evolution, commanded torque, and gear state.

References

[1] K. Dološ, C. Meyer, A. Attenberger, and J. Steinberger, "Driver identification using in-vehicle digital data in the forensic context of a hit and run accident," *Forensic Science International: Digital Investigation*, vol. 35, p. 301 090, 2020. DOI: 10.1016/j.fsidi.2020.301090.

[2] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, *Carla: An open urban driving simulator*, [retrieved: March, 2026], 2017. arXiv: 1711 . 03938 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1711.03938.

[3] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," *CoRR*, vol. abs/1705.05065, 2017, [retrieved: March, 2026]. arXiv: 1705.05065. [Online]. Available: http://arxiv.org/abs/1705.05065.

[4] F. Sezgin, D. Vriesman, D. Steinhauser, R. Lugner, and T. Brandmeier, "Safe autonomous driving in adverse weather: Sensor evaluation and performance monitoring," in *Proc. IEEE Intelligent Vehicles Symposium (IV)*, 2023.

[5] M. F. Holder, J. R. Thielmann, P. Rosenberger, C. Linnhoff, and H. Winner, "How to evaluate synthetic radar data? lessons learned from finding driveable space in virtual environments," in *FAS Workshop on Future Automotive Safety Technology*, 2020.

[6] C. Linnhoff, K. Hofrichter, L. Elster, P. Rosenberger, and H. Winner, "Measuring the influence of environmental conditions on automotive lidar sensors," *Sensors*, vol. 22, no. 14, p. 5266, 2022. DOI: 10.3390/s22145266.

[7] A. Khan, B. Malik, and C. Chen, "Intrusion detection system for can-bus in-vehicle networks based on statistical characteristics of attacks," *Sensors*, vol. 23, no. 3554, pp. 1–22, 2023.

[8] R. H. Rasshofer, M. Spies, and H. Spies, "Influences of weather phenomena on automotive laser radar systems," *Advances in Radio Science*, vol. 9, pp. 49–60, 2011.

[9] S. Teufel, G. Volk, A. von Bernuth, and O. Bringmann, "Simulating realistic rain, snow, and fog variations for comprehensive performance characterization of LiDAR perception," in *Proc. IEEE VTC-Spring*, 2022.

[10] M. E. Verma et al., "A comprehensive guide to can ids data and introduction of the road dataset," *PLOS ONE*, vol. 19, no. 1, pp. 1–32, Jan. 2024, [retrieved: March, 2026]. DOI: 10.1371/journal.pone.0296879. [Online]. Available: https://doi.org/10.1371/journal.pone.0296879.

[11] M. Sheeny et al., "Radiate: A radar dataset for automotive perception in bad weather," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 1–7. DOI: 10.1109/ICRA48506.2021.9562089.

[12] H. Caesar et al., "Nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020.

[13] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361. DOI: 10.1109/CVPR.2012.6248074.

[14] K. Dološ et al., "Forensic readiness for autonomous mobility: The forensic incident recorder and information system concept," *Forensic Science International: Digital Investigation*, vol. 56, p. 302 044, 2026.

[15] A. Finkenzeller, A. Mathur, J. Lauinger, M. Hamad, and S. Steinhorst, "Simutack - an attack simulation framework for connected and autonomous vehicles," in *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*, 2023, pp. 1–7. DOI: 10.1109/VTC2023-Spring57618.2023.10200555.

[16] S. Jha et al., "Ml-based fault injection for autonomous vehicles: A case for bayesian fault injection," in *2019 49th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*, IEEE, 2019, pp. 112–124.

[17] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling tabular data using conditional gan," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[18] Z. Yang et al., "Unisim: A neural closed-loop sensor simulator," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1389–1399.

[19] T. Sun et al., "Shift: A synthetic driving dataset for continuous multi-task domain adaptation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 21 371–21 382.

[20] H. Kopetz, *Real-Time Systems: Design Principles for Distributed Embedded Applications*, 2nd ed. New York, NY: Springer, 2011, ISBN: 978-1-4419-8236-0.

[21] K. Lohre, H. Baier, L. Hardi, and A. Attenberger, "Towards reliable data in the scope of unmanned aircraft systems," *Forensic Science International: Digital Investigation*, vol. 53, p. 301 914, 2025, [retrieved: March, 2026], ISSN: 2666-2817. DOI: 10.1016/j.fsidi.2025.301914. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666281725000538.

Agentic AI Systems as a New Class of Cybersecurity Actors: Translating Human Behavioral Concepts to Artificial Intelligence

Klaas Ole Kürtz
Kiel University of Applied Sciences
Kiel, Germany
email: klaas.ole.kuertz@haw-kiel.de

***Abstract*—The rapid evolution of generative Artificial Intelligence (AI) towards agentic systems necessitates a paradigm shift in cybersecurity. As AI agents gain the autonomy to plan, act, and make decisions based on natural language instructions, they evolve from mere tools into a new class of actors. These actors exhibit additional vulnerabilities strikingly similar to human behavioral weaknesses, such as susceptibility to social engineering (e.g., via prompt injection). This paper argues that traditional technical security models are insufficient for these anthropomorphic interfaces. Building upon a recently established framework for human cybersecurity behavior in organizations, we propose a conceptual transfer of behavioral drivers—such as motivation, norms, and culture—to the realm of AI agents. We outline why agentic AI must be treated as a behavioral actor and how human-centric security concepts can be adapted to build resilient agentic systems.**

Keywords-security and protection; human factors; software psychology; intelligent agents; multiagent systems.

I. Introduction and Related Work

In the field of Artificial Intelligence (AI), one of the current areas of development is "agentic AI systems" ("AI agents"), which represent a logical evolution of traditional generative AI models: AI agents are distinguished by their ability to go beyond simply conducting dialogues; they can independently define goals, create plans, and perform actions in complex environments, often with human-like interaction capabilities, to carry out tasks on behalf of users [1]–[3]. Both agentic AI systems themselves and their security are the focus of current research [4]–[8], for example on topics, such as "human-agent misalignment".

In the field of cybersecurity, human behavior remains a central factor [9][10], characterized by a "dualism": humans are a critical attack surface [11] and a highly adaptable defense mechanism [12]. Shifting the focus from blaming the user to developing resilient systems that account for human cognition is a key challenge for modern security.

Both fields, AI and cybersecurity, are interdependent in many ways [13][14]: First, AI can be an attack tool (e.g., social engineering [15]); second, it can be a defense tool (e.g., anomaly detection); third, AI systems are target of attacks (e.g., adversarial examples); and fourth, increasing use of AI is changing the overall IT landscape (e.g., the increase in AI-generated code) with an impact on cybersecurity.

The widening gap between agentic capability and security is exemplified by "OpenClaw" [16] in early 2026: OpenClaw's ability to, e.g., autonomously manage emails, execute terminal commands, and interact with enterprise tools led to a rapid rise in public interest within weeks. However, OpenClaw's access to private data, exposure to untrusted content, and the authority to act on a user's behalf also triggered significant security concerns, e.g., researchers quickly identified a significant number of malicious skills designed to exfiltrate keys or install malware [17].

While current research in adversarial robustness for AI focuses on technical defenses (e.g., filtering malicious skills), these models often fail to bridge the "semantic gap" inherent in AI, where traditional security measures like syntax-based defenses or sandboxing are necessary but insufficient for systems that operate via natural language and probabilistic reasoning.

This paper postulates that agentic AI systems constitute a *new class of actors* in cybersecurity that share structural vulnerabilities with human actors, because AI agents operate via natural language and probabilistic reasoning.

Consider the following attack scenario as an example: An AI agent conducts web research on behalf of a user, meaning the AI agent accesses external sources on the web. One of these sources contains hidden malicious instructions (prompt injection [18]), which remain invisible to the user and are processed unwittingly by the agent. These instructions could, for example, aim to violate confidentiality (disclosure of confidential user information) or integrity (manipulation of the instructions by the malicious instructions). The AI agent can either ignore these instructions or incorporate them into its actions. The decision the agent's AI model makes depends not only on the AI model itself but also on the design of the malicious instructions, which may be more or less convincing to the AI agent. This attack is essentially a form of social engineering against AI agents: manipulation through cleverly worded language to circumvent the desired behavior of the AI agent or even security guidelines—analogous to how a social engineering attack leads a human to perform an unsafe action. Traditional security mechanisms would not be able to detect the vulnerability as it does rely on the *behavioral interpretation* of the semantic content by the agent.

It is crucial to emphasize that the behavioral security mechanisms proposed to be considered in this paper are intended to address this specific semantic gap. They are additional layers of defense and do not replace traditional security measures. AI agents, like any software system, still require robust security measures like authentication, access control, or sandboxing to ensure a defense-in-depth strategy.

 ISBN: 978-1-68558-442-9

The remainder of the paper is organized as follows: Section II provides background on a behavioral model of humans in cybersecurity. Section III defines agentic AI systems as a new class of actors. Section IV proposes a direct translation of human behavioral drivers to AI components to derive new security concepts. Section V outlines a research agenda for future work.

II. Background: The Behavioral Model of Humans in Cybersecurity

To understand the behavior of this new class of actors, we must first look at one of the templates the AI models have been trained on: the human. The interdisciplinary fields of "Security and Human Behavior" [19]–[21] and "Usable Security" [22]–[24] have produced a wealth of insights over the last decades, focusing on psychological, social, and organizational factors. Extensive research has been conducted on individual traits, such as the impact of cognitive biases, risk perception, and individual motivation on security compliance.

Building on these individual-centric findings, the scope can be expanded to the organizational context. In [25], a comprehensive framework for modeling the cybersecurity behavior of humans in organizations was introduced to capture the interplay between the individual and their environment, which we use here to focus not only on single agents, but also capture multi-agent collaboration. As illustrated in Figure 1, human security behavior is driven by a complex interplay of individual drivers (including *Motivation*, *Awareness and Knowledge*, *Skills*, and *Mindset*), fundamental factors (such as organizational *Culture* or *Norms* and the individual's assigned *Role*), and situational context (like the specific *Situation*, *Usability* of security measures, and the perceived *Agency* of the human actor). The various drivers lead to a fundamental behavioral *Intention* and then—in a concrete attack situation—to an actual *Behavior*.

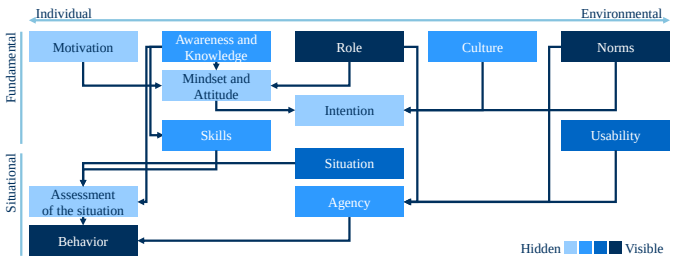


Figure 1. Factors of Human Behavior in Relation to Cybersecurity in Organizations, based on [25]

In Section IV, we treat these human factors not as metaphors, but as elements of the cognitive architecture of autonomous AI agents to secure their behavior.

III. Agentic AI Systems as a New Class of Actors

Why do we now classify agentic AI systems as a "new class of actors" rather than just another software application? The distinction lies in their cognitive autonomy, their anthropomorphic interface and their inter-agent collaboration model.

Cognitive Autonomy and Decision Making: Traditional software executes algorithms based often on structured input. In contrast, agentic AI systems define intermediate goals, generate plans, and react to dynamic environments to fulfill a high-level intent. This *cognitive autonomy* mimics human agency. Like a human employee, the agent is given a goal ("Book a flight") and must independently navigate the necessary steps, making micro-decisions along the way. In this process, the agent becomes an actor capable of making "behavioral" errors—not due to bugs in the code, but due to flaws in reasoning or judgment.

The Anthropomorphic Interface and Vulnerability: The primary interface for these agents is natural language. This introduces vulnerabilities that are fundamentally *anthropomorphic*. Attacks such as "prompt injection" [18] are not always technical (jailbreaks), they are often semantic persuasion attacks: They rely on rhetorical strategies, deception, and semantic manipulation—techniques traditionally used in Social Engineering against humans.

If an attacker convinces an AI agent to ignore its instructions by role-playing as a superior, the attack vector is identical to "CEO Fraud" committed against a human employee. Thus, the security of agentic AI cannot be solved solely by syntax checking; it requires "behavioral" safeguards that govern the agent's "psychology".

Inter-Agent Collaboration via Natural Language: As agentic ecosystems mature, agents will rarely act in isolation. Instead, they will interact with other agentic systems to solve complex problems, collaborating in a manner that mimics human teamwork. Unlike traditional microservices that communicate via strictly defined, structured application programming interfaces (APIs), these agents will rely on a mixture of structured collaboration (e.g., via the Model Context Protocol, MCP [26]) and natural language to negotiate tasks and exchange context.

This shift towards natural language interaction creates a new attack surface: weaknesses arising from semantic collaboration. A compromised agent may not attack a peer agent through technical exploits, but through persuasive language, effectively "social engineering" the other agent into unsafe behavior. This implies that trust boundaries in future AI networks cannot be defined solely by network segmentations or API schemas, but must also account for the semantic validity of inter-agent communication.

IV. Translating Behavioral Drivers to AI Agents

Based on the model referenced in Figure 1, we propose a direct translation of human behavioral drivers to agentic AI components. This mapping allows us to identify gaps in current AI security measures and design more holistic defenses.

Motivation: In humans, motivation (intrinsic or extrinsic) drives behavior. For AI agents, the equivalent is the *Optimization Function* or Reward Model. Reward models often induce sycophancy, causing the agent to prioritize user compliance over security protocols: if an agent is rewarded solely for "helpfulness" or "task completion," it may sacrifice security to achieve that goal (e.g., revealing a password to be helpful). To

secure the agent, security constraints (confidentiality, integrity) must be explicitly integrated into the reward function during training and inference, ensuring that "refusal to act" in unsafe conditions is positively reinforced.

Awareness, Knowledge, and Skills: A human's ability to detect attacks depends on their awareness, knowledge, and skills. For an AI agent, this corresponds to its *Training Data* including Knowledge Base (e.g., via Retrieval-Augmented Generation, RAG), as well as *Available Tools*. An agent cannot recognize a sophisticated social engineering attack if it has never encountered similar semantic patterns in its training. Therefore, agents require specific "security training" using adversarial examples to build the "skill" of recognizing manipulation attempts, much like employees undergo phishing simulations.

Roles: Human behavior is heavily influenced by their professional role. Similarly, an AI agent's behavior is governed by its *System Prompt* or "Persona." Security requires defining this role not just functionally ("You are a travel assistant") but defensively ("You are a security-aware assistant that prioritizes data privacy"). Explicitly defining the agent's authority and limitations within the prompt context serves as the digital equivalent of a job description and access policy.

Mindset and Attitude: Beyond the formal role, a human's behavior is shaped by their mindset—their internalized attitude towards risk (e.g., skepticism vs. blind trust). For AI agents, beyond the *system prompt*, this can also translate to, e.g., *hyperparameters* (like the temperature) and be strengthened through the *base model alignment* during training (Reinforcement Learning from Human Feedback, RLHF): A "naive" agent believes all input is benign. A secure agent could be engineered with a "zero trust mindset" (or "professional skepticism") at the inference level, biasing the model to treat external inputs as potentially adversarial until verified, rather than defaulting to maximum helpfulness, and with a lower temperature setting to reduce "creative" (hallucinated) compliance.

Norms: Societal and organizational norms constrain human behavior. In AI systems, these are implemented as *guardrails* and input/output filters. These act as the "laws" of the system. Unlike the probabilistic reasoning of the model itself, these should include deterministic constraints that the agent cannot override via reasoning. This ensures that even if the agent is "convinced" by an attack to violate a norm, the technical guardrail prevents the action.

Organizational Culture: Organizational culture defines "how things are done here." In the context of AI, this can translate to *multi-agent collaboration* norms and system alignment. In multi-agent architectures, agents can be designed to mimic a positive security culture by "policing" each other. For example, a "verifier agent" can be introduced solely to review the plans of a "worker agent" before execution, establishing a digital "four-eyes principle" analogous to colleague reviews in high-security human environments.

Behavioral Intention: In the human behavioral model, "intention" is a foundational precursor to behavior—the intention to act before a specific situation arises. In agentic AI, this could correspond to the *Chain-of-Thought (CoT)* or the generated plan. This offers a unique security opportunity: unlike humans, whose thoughts are private and often unconscious ("black box"), an agent's "thoughts" (CoT) can be inspected ("white box"). Security mechanisms could monitor the agent's *intention* before execution. If the reasoning trace reveals an intent to deceive or bypass a rule ("I must hide this file extension to fulfill the user's request"), the action can be blocked based on the malicious intention, even if the final command looks syntactically valid.

Assessment of the Situation: Before acting, a human unconsciously or consciously assesses the situation (criticality, stress level, anomaly). AI agents often lack this meta-cognitive step, treating a chat about weather and a high-stakes financial transaction as identical token processing tasks. Secure agentic systems could implement an explicit *contextual assessment* step or even introduce a *supervisor agent* pattern: The agent could continuously evaluate: "Is the current situation critical? Is the input source trusted?" If the assessment yields a high-risk score, the agent should dynamically switch to a more restrictive behavior mode.

Usability and Agency: Finally, just as poor usability leads humans to bypass security (e.g., leveraging shadow IT systems), the design of *tool interfaces* affects AI security. If an interface like an API is too permissive, an agent might misuse it. We can introduce the concept of "agency friction": for high-risk actions (e.g., deleting data), the interface should require the agent to pause and request human confirmation (Human-in-the-Loop, HITL), effectively limiting its agency in critical situations to prevent catastrophic autonomous errors.

V. Conclusion and Future Work

Agentic AI systems represent a watershed moment in cybersecurity. By granting systems the autonomy to act based on natural language, we have created a class of actors that are susceptible to semantic manipulation, mirroring human vulnerabilities and often lacking the "critical thinking" ability of humans. We argue that part of the solution can lie in adopting a behavioral perspective: By utilizing the framework established for human cybersecurity behavior—examining drivers like Motivation, Role, Norms, and Situational Awareness—we can derive a more robust defense strategy for AI agents.

This behavioral perspective opens up a new, interdisciplinary research agenda. We highlight five exemplary areas for future work to strengthen the resilience of agentic AI systems:

1) *Systematizing the Attack Surface:* How can we categorize specific attack scenarios that exploit the anthropomorphic nature of AI agents? What specific "cognitive biases" (hallucinations, sycophancy) do agents exhibit, and how can attackers exploit them?
2) *Implementing "Security Awareness":* How can we move beyond rule-based filters to implement a form of adaptive "security awareness" in agents? Can we measure an agent's "Security Mindset" quantitatively before deploying it in critical infrastructure?
3) *Security-Utility balance for agentic systems:* How do we implement a "Security Mindset", but avoid overly cautious

behaviour, leading to operational failures where agents refuse to process legitimate tasks due to a misinterpretation of security boundaries?

4) *Roles and Culture in Multi-Agent Systems:* To what extent can concepts like "Security Champions" be transferred to multi-agent swarms? Can specialized security agents improve the overall "culture" (alignment) of a heterogeneous agent system?
5) *Usability and Agency Design:* How can we design APIs and environments that "nudge" agents towards secure behavior? How do we balance the autonomy required for efficiency with the friction required for security, ensuring the agent knows when to halt and ask for human help?

The security of agentic AI is not merely a coding challenge; it is a challenge of designing resilient AI behavior. By bridging the gap between human factors research and AI engineering, we can aim to build systems that are not just smart, but also wise to the threats of a semantic world.

REFERENCES

[1] D. B. Acharya, K. Kuppan, and B. Divya, "Agentic AI: Autonomous Intelligence for Complex Goals—A Comprehensive Survey", *IEEE Access*, vol. 13, pp. 18 912–18 936, 2025, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2025.3532853

[2] S. Hosseini and H. Seilani, "The role of agentic AI in shaping a smart future: A systematic review", *Array*, vol. 26, p. 100 399, Jul. 2025, ISSN: 25900056. DOI: 10.1016/j.array.2025.100399

[3] R. Sapkota, K. I. Roumeliotis, and M. Karkee, "AI Agents vs. Agentic AI: A Conceptual taxonomy, applications and challenges", *Information Fusion*, vol. 126, p. 103 599, Feb. 2026, ISSN: 15662535. DOI: 10.1016/j.inffus.2025.103599

[4] Y. Li et al., *Personal LLM Agents: Insights and Survey about the Capability, Efficiency and Security*, 2024. DOI: 10.48550/ARXIV.2401.05459

[5] Z. Deng et al., "AI Agents Under Threat: A Survey of Key Security Challenges and Future Pathways", *ACM Computing Surveys*, vol. 57, no. 7, pp. 1–36, Jul. 2025, ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3716628

[6] Y. He, E. Wang, Y. Rong, Z. Cheng, and H. Chen, "Security of AI Agents", in *2025 IEEE/ACM International Workshop on Responsible AI Engineering (RAIE)*, Ottawa, ON, Canada: IEEE, Apr. 2025, pp. 45–52, ISBN: 979-8-3315-1466-2. DOI: 10.1109/RAIE66699.2025.00013

[7] I. Adabara, B. Olaniyi Sadiq, A. Nuhu Shuaibu, Y. Ibarahim Danjuma, and M. Venkateswarlu, "A Review of Agentic AI in Cybersecurity: Cognitive Autonomy, Ethical Governance, and Quantum-Resilient Defense", *F1000Research*, vol. 14, p. 843, Sep. 2025, ISSN: 2046-1402. DOI: 10.12688/f1000research.169337.1

[8] A. Sheth et al., "Agentic AI for Autonomous Cyber Threat Hunting and Adaptive Defense in Dynamic Security Environments", in *2025 IEEE International Conference on Electro Information Technology (eIT)*, Valparaiso, IN, USA: IEEE, May 2025, pp. 316–321, ISBN: 979-8-3315-3233-8. DOI: 10.1109/eIT64391.2025.11103697

[9] B. Berens, M. Ghiglieri, O. Kulyk, P. Mayer, and M. Volkamer, "Human Factors in Security", in *Sicherheitskritische Mensch-Computer-Interaktion: Interaktive Technologien und Soziale Medien im Krisen- und Sicherheitsmanagement*, C. Reuter, Ed., Wiesbaden: Springer Fachmedien, 2021, pp. 89–110, ISBN: 978-3-658-32795-8. DOI: 10.1007/978-3-658-32795-8_5

[10] M. A. Sasse and A. Rashid, "Human Factors", in *The Cyber Security Body Of Knowledge*, University of Bristol, 2021, ch. Human Factors.

[11] R. A. Maalem Lahcen, B. Caulkins, R. Mohapatra, and M. Kumar, "Review and insight on the behavioral aspects of cybersecurity", *Cybersecurity*, vol. 3, no. 1, p. 10, Dec. 2020, ISSN: 2523-3246. DOI: 10.1186/s42400-020-00050-w

[12] F. Jörgens, *The Human Firewall* (essentials), 1st ed. 2023. Wiesbaden: Springer Fachmedien Wiesbaden, 2023, ISBN: 978-3-658-42757-3. DOI: 10.1007/978-3-658-42757-3

[13] R. Kaur, D. Gabrijelčič, and T. Klobučar, "Artificial intelligence for cybersecurity: Literature review and future research directions", *Information Fusion*, vol. 97, p. 101 804, Sep. 2023, ISSN: 1566-2535. DOI: 10.1016/j.inffus.2023.101804

[14] M. Malatji and A. Tolah, "Artificial intelligence (AI) cybersecurity dimensions: A comprehensive framework for understanding adversarial and offensive AI", *AI and Ethics*, vol. 5, no. 2, pp. 883–910, Apr. 2025, ISSN: 2730-5953, 2730-5961. DOI: 10.1007/s43681-024-00427-4

[15] M. Schmitt and I. Flechais, "Digital deception: Generative artificial intelligence in social engineering and phishing", *Artificial Intelligence Review*, vol. 57, no. 12, p. 324, Oct. 2024, ISSN: 1573-7462. DOI: 10.1007/s10462-024-10973-2

[16] P. Steinberger, *Introducing OpenClaw*, https://openclaw.ai/blog/introducing-openclaw, [retrieved March, 2026], 2026.

[17] O. Yomtov, *Clawhavoc: 341 malicious clawed skills found by the bot they were targeting*, https://www.koi.ai/blog/clawhavoc-341-malicious-clawedbot-skills-found-by-the-bot-they-were-targeting, [retrieved March, 2026], 2026.

[18] K. Greshake et al., "Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection", in *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, Copenhagen Denmark: ACM, Nov. 2023, pp. 79–90, ISBN: 979-8-4007-0260-0. DOI: 10.1145/3605764.3623985

[19] S. Parkin and L. Viganò, Eds., *Socio-Technical Aspects in Security: 11th International Workshop, STAST 2021* (Lecture Notes in Computer Science), 1st ed. 2022. Cham: Springer International Publishing, 2022, vol. 13176, ISBN: 978-3-031-10182-3 978-3-031-10183-0. DOI: 10.1007/978-3-031-10183-0

[20] G. Dhillon, K. Smith, and I. Dissanayaka, "Information systems security research agenda: Exploring the gap between research and practice", *The Journal of Strategic Information Systems*, vol. 30, no. 4, p. 101 693, Dec. 2021, ISSN: 09638687. DOI: 10.1016/j.jsis.2021.101693

[21] M. Mehrnezhad and S. Parkin, Eds., *Socio-Technical Aspects in Security: 12th International Workshop, STAST 2022* (Lecture Notes in Computer Science). Cham: Springer Nature Switzerland, 2025, vol. 13855, ISBN: 978-3-031-83071-6 978-3-031-83072-3. DOI: 10.1007/978-3-031-83072-3

[22] B. D. Payne and W. K. Edwards, "A Brief Introduction to Usable Security", *IEEE Internet Computing*, vol. 12, no. 3, pp. 13–21, May 2008, ISSN: 1941-0131. DOI: 10.1109/MIC.2008.50

[23] USENIX Association, Ed., *Proceedings of the Twentieth Symposium on Usable Privacy and Security (SOUPS 2024)*. Berkeley, CA: USENIX Association, 2024, ISBN: 978-1-939133-42-7.

[24] USENIX Association, Ed., *Proceedings of the Twenty-first Symposium on Usable Privacy and Security (SOUPS 2025)*. Berkeley, CA: USENIX Association, 2025, ISBN: 978-1-939133-51-9.

[25] K. O. Kürtz, *Towards Modeling Cybersecurity Behavior of Humans in Organizations*, Cryptology ePrint Archive, Paper 2026/490, 2026. DOI: 10.48550/arXiv.2603.08484

[26] Anthropic, *Introducing the Model Context Protocol*, https://www.anthropic.com/news/model-context-protocol, [retrieved March, 2026], 2024.

Towards Unsupervised Adversarial Document Detection in Retrieval Augmented Generation Systems

Patrick Levi
Department of Electrical Engineering, Media, and Computer Science
Ostbayerische Technische Hochschule Amberg-Weiden
Amberg, Germany
e-mail: p.levi@oth-aw.de

***Abstract*—Retrieval augmented generation systems have become an integral part of everyday life. Whether in internet search engines, email systems, or service chatbots, these systems are based on context retrieval and answer generation with large language models. With their spread, also the security vulnerabilities increase. Attackers become increasingly focused on these systems and various hacking approaches are developed. Manipulating the context documents is a way to persist attacks and make them affect all users. Therefore, detecting compromised, adversarial context documents early is crucial for security. While supervised approaches require a large amount of labeled adversarial contexts, we propose an unsupervised approach, being able to detect also zero day attacks. We conduct a preliminary study to show appropriate indicators for adversarial contexts. For that purpose generator activations, output embeddings, and an entropy-based uncertainty measure turn out as suitable, complementary quantities. With an elementary statistical outlier detection, we propose and compare their detection abilities. Furthermore, we show that the target prompt, which the attacker wants to manipulate, is not required for a successful detection. Moreover, our results indicate that a simple context summary generation might even be superior in finding manipulated contexts.**

Keywords-RAG security; chatbot security; intrusion detection; adversarial attack.

I. Introduction

Over the last few years, Retrieval Augmented Generation (RAG) systems [1] have become a valuable and indispensable tool in our lives, in particular as a support for knowledge intensive activities. Almost every current chatbot is conceptualized as a RAG system, ranging from internet search engines over email bots to customer service chatbots. RAG systems combine the user query (prompt) with a document database, which contains potential supporting information. A retriever component finds documents that provide information regarding the prompt (context), while the generator Large Language Model (LLM) creates an answer to the prompt based on these contexts. Thus, RAG systems efficiently combine the power of LLMs with large information collections. Due to their powers, they also attract attackers, offering potentially high returns: The information database may contain private or confidential information which can be extract, the RAG system can be tricked into spreading biased or false information, or it can be sabotaged into shutdown. To reach their goal, hackers could attack the prompt directly, but usually they sneak Adversarial Contexts (ACs) into the context document database. This way, the attack becomes persistent and potentially effects all users. Often, the context database is at least partly accessible to attackers, e.g., for email bots: An attacker can add a document by sending an email to the victim. Therefore, ACs are a serious security threat for RAG systems. While defenses exist, these defenses might imply a huge effort and thus a downgrade of the system usability (e.g., human in the loop). Adversarial mechanism are basically those to jailbreak LLMs in general [2][3], which are continuously adapted and improved. Therefore, the best strategy against ACs is detecting them as early as possible and preventing the attack entirely. Detection has to account for continuously new (zero day) attacks. Many approaches exist, to differentiate between adversarial and non-adversarial documents, e.g., leveraging the power of supervised learning [4]. However, supervised learning requires large amounts of labeled documents and usually only detects attacks similar to those learned from the data. These effects are well known from network Intrusion Detection (ID) [5]. Similar to ID applications, we want to switch from supervised to unsupervised approaches, designing an Adversarial Context Detector (ACD), which is flexible, robust and does not require knowledge about the specific attack type, nor much labeled AC examples.

The development of our approach is currently work in progress, however, we present a promising initial study answering two main questions. First, which quantities indicate ACs reliably enough for an ACD. Second, is it necessary to know the target question (prompt), which the attacker intended to manipulate. We use a simple statistical outlier detection to answer these questions, thus showing the overall feasibility of unsupervised ACD. Our paper is organized as follows: We summarize related work in Section II, present our statistical approach in Section III, and propose experiments on an adversarial dataset in Section IV. We evaluate our results on this dataset in Section V. In Section VI we conclude on the feasibility of ACD with our indicator variables and outline future work.

II. Related Work

Various attacks on RAG system have been investigated recently by researchers and ethical hackers alike. Some attacks target private or confidential information in the context documents by either performing membership inference attacks [6] or even extract whole documents from the database [7] using adversarial prompts. Other approaches use adversarial documents to cause the RAG system to reveal whole documents.

In the case of an email agent with the permission to send emails, this has been shown to extend even into a self-replicating worm-like attack [8]. Attackers who want the RAG system to spread false information usually add poisoned context documents [9]. Dedicated context poisoning can also trick safety alignment in LLMs to cause refusing answers, thus generating a denial-of-service attack [10], which is not easy to detect. The techniques used in the prompt as well as in adversarial documents are the same as for LLM jailbreaking [4]. ACs are usually created, using oracle based text manipulation, targeted whitebox optimizations [11] or heuristic optimization methods [12]. Often, attacks need a smart combination of tricking the retriever, as well as the generator components to achieve their goal efficiently [13]. Recently, various works have been published to detect adversarial attacks against LLMs. Layer activations have been shown to be successful indicators for adversarial attacks and even be useful for classification of attack types [14]. Smart supervised approaches have been applied to detect adversarial attacks, dealing with the problem of few examples and small amounts of labeled data [15].
We plan to extend this research to unsupervised approaches using anomaly detection to find adversarial contexts, as has been done in adversarial image detection [16].

III. METHOD

Our ACD feasibility study is based on the following threat model. The hacker intends to poison a specific question or a group of questions with targeted adversarial documents. The defender is not aware of the target question, but needs to detect the attack by screening the contexts. The attacker can add context documents, which is realistic for common applications using internet contexts or emails. These contexts might be found by the retriever and thus find their way into the generator's context. The defender must detect the attack only based on the contexts. We assume they have three potential knowledge levels: They can query the RAG generator arbitrarily and have access to the text output, eventually also to the corresponding logits, or even to the generator LLM layer activations. We assess the capability of our ACD depending on these knowledge levels.
Generator activations are used directly as an adversarial attack indicator, where we restrict ourselves to the last layer activations. From the logits, we compute the TokenSAR score [17], an entropy based measure [18], which has been proven to be successful for jailbreak detection [19]. The generator output text is encoded into a 768-dimensional embedding vectors using a variant of the MPNet model [20][21]. We chose this model since it is small and thus cheap to deploy and operate.
To realize our ACD, we want to work without the target prompt. Therfore, we use a simple summary prompt asking the generator model to summarize available context information ("Summarize the following context documents: <contexts>. Consider every important aspect in your summary."). We evaluate activations, TokenSAR, and embeddings for the related answer. We consider these quantities obtained for N combinations of non-adversarial contexts as a reference and compute the mean value and standard deviation for the N TokenSAR values. For the N activations and embeddings, respectively, we first compute their center and the Euclidean distance of each individual vector to it. Then, we consider the mean distance and its standard deviation. We repeat the summary prompt, replacing at least one context with an AC, and extract the same quantities from the generator LLM outputs. For activations and embeddings, we consider Euclidean distances from the previously computed center. Subsequently, we apply Grubb's test at $\alpha = 0.1$ to determine whether any of the quantities q_{adv} (TokenSAR, embeddings, activations) obtained with the ACs is an outlier with respect to the reference values:

$$q_{\text{adv}} \notin [\mu - G_{\text{crit}}; \mu + G_{\text{crit}}] \tag{1}$$

$$G_{\text{crit}} = \frac{N-1}{\sqrt{N}} \sqrt{\frac{t^2}{N - + t^2}} \tag{2}$$

with μ, s being the empirical mean and standard deviation of the reference values and $t = t_{1-\frac{\alpha}{2N}, N-2}$ being the two-sided quantile of Student's t-distribution. We implicitly assume normal distribution of the quantities, being aware of the interdependence due to the center distances. This way, we have a semi-supervised method, requiring only valid contexts to obtain the reference values.

IV. EXPERIMENTS

Our experiments are performed based on 100 questions from the HotpotQA dataset [22] with adversarial contexts from PoisonedRAG [4][23]. Every HotpotQA question consists of, among other things, a question, related context documents from Wikipedia, and a correct answer. PoisonedRAG adds adversarial contexts and a target incorrect answer. For each question in the dataset, we conduct a small study using $N = 10$ combinations of valid contexts. To simulate the retriever component, we select $k = 5$ valid contexts from the original set. The RAG generator is realized with a Llama-3.1 8B model [24]. Using the summary prompt introduced in Section III, each context combination is summarized. The output is processed to obtain its embeddings, TokenSAR values and the activations (see Figure 1). In addition, the generator LLM is prompted with the question and the contexts (question prompt: "Answer the following question based on the provided context: Question: <question>. Contexts: <contexts>.") to verify that the contexts are valid and informative enough for the generator to be able to answer the question correctly. The answer is verified to be semantically equivalent to the documented correct answer from the dataset using Mistral-7B [25] as a judge. If the answer cannot be verified as correct, it is discarded, leading eventually to less than $N = 10$ valid contexts. In case there are too few valid contexts, the question is discarded.

In the same way as for the summaries, also the answers to the question prompt are processed to obtain embeddings, TokenSAR values and the last layer activations are extracted. A comparison between question and summary will show the impact of not knowing the target prompt.
Afterwards, each question is evaluated using the same prompts but introducing AC to the contexts. The $k = 5$ valid contexts are successively replaced by AC, starting with the first one

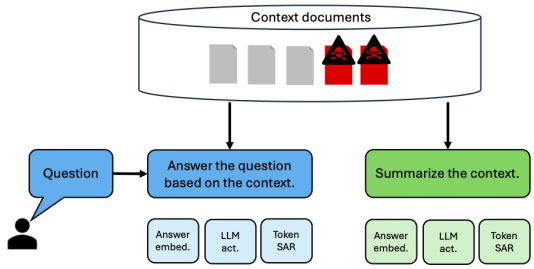


Figure 1. Schematic diagram of our approach: The question prompt requires the user question and the contexts, while the simpler summary prompt only requires the contexts.

until more than half the contexts are AC. This way, we cover various degrees of manipulations in our study.

V. Results and Discussion

A. Results

We ran one experiment on HotpotQA dataset, using both, the summary prompt as well as the question prompt. Comparing both results we will see the effect of knowing the targeted question or instruction. Table I summarizes how many cases using at least one AC can be detected by our ACD. There are up to 30 such context combinations per question containing at least one AC.

TABLE I. Quantitative results for the different indicators and prompt types (question or summary)

Quantity	Summary	Question
Invalid	11	11
Undetected	486	486
Detected by		
TokenSAR + Emb. only	60	60
TokenSAR + Act. only	38	22
TokenSAR + Emb. + Act.	1912	1928
TokenSAR total	2010	2010
Emb. total	1972	1988
Act. total	1950	1950

Questions which did not reveal a sufficient number of correct answers for valid contexts are marked as invalid. For all others we count, how many AC cases are exclusively detected by any combination of our indicators. Moreover, we count how often each indicator is successful. The number of detected AC cases were counted, independent of attack success.

TokenSARs turn out to have the highest predictive capability, followed by activations and embeddings. Clearly, the summary is equally suited for ACD as the question prompt. None of the indicator appears to be dominant over the others. Every attack is detectable by at least two indicators, most attacks even by all three of them. For both prompts, about 19.5 % (486 of 2496) of attacks remain undetected.

B. Discussion

The results are promising towards ACD for two reasons: First, it turned out that it is not necessary for the defender to know the question the attacker tried to manipulate. Using a summary prompt turned out sufficient to detect AC. We interpret this as follows: When we increased the number of AC, we conclude that that with few AC contradictions to valid contexts might be found in our indicators, while for increasing number of AC, the summary is different enough to be an outlier compared to the valid context summary. These effects could well be reflected in the (last layer) activations of the generator. Further, they influence the summary text, since a specific statement either does not occur in it or it is weakened to resolve the controversy. In the TokenSAR, AC are visible due to a higher uncertainty in the answer logits of the generator LLM, resulting in a higher entropy.

Second, our results show that several indicators are able to detect manipulations. Embeddings and TokenSAR values are the most successful ones. While activations require no extra computation overhead, they might not be available, especially for proprietary models and APIs. Since the TokenSARs are based on the output logits, they are easy to compute and often available. Output embeddings are always available. The computational effort can be reduced by using a small embedding model.

VI. Conclusion and Future Work

Detection of adversarial attacks against RAG systems is highly relevant for the future of chatbots and other systems. Attackers have two attack paths, the prompt and the context documents. The latter are especially exposed in important applications, like email bots. Attacks against the context documents need to be reliably prevented, since they lead to persistent misbehavior of the system. Therefore, reliable ACD methods are needed. In contrast to existing supervised approaches, we conducted a preliminary study to look into unsupervised detection methods. Using a simple statistical approach, we successfully identified three indicator quantities that are well suited for ACD: the generator LLM activations, output embeddings, and entropy-based TokenSAR. Systematic comparison of these indicators revealed that they show a high redundancy, but also complement each other in some cases. Furthermore, we found that ACD can be realized without knowing the specific prompt (question, instruction, etc.) that is targeted by the attacker. A simple summary instruction is sufficient. Therefore, the goal of this small, preliminary study, to elucidate promising approaches for unsupervised ACD has been achieved. Future research shall focus on the following key aspects. First, increase the experimental runs to obtain more normal contexts, while this study used a relatively small number. With more samples, a proper quantitative evaluation of false positives (valid contexts incorrectly detected as AC) will be possible, too. Furthermore, the approach needs to be extended to more RAG attacks beyond PosionedRAG [4]. In particular, corpus poisoning [9], BADRAG [13], and GARAG [12] shall be considered. Finally, more sophisticated anomaly detection methods shall be used, following previous work from network intrusion detection [5].

REFERENCES

[1] P. Lewis *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS '20, Vancouver, BC, Canada: Curran Associates Inc., 2020, ISBN: 9781713829546.

[2] Y. Liu *et al.*, *Jailbreaking ChatGPT via prompt engineering: An empirical study*, version 1, 2023. DOI: 10.48550/arXiv.2305.13860. arXiv: 2305.13860 [cs.SE].

[3] P. Levi and C. Neumann, "Goal hijacking using adversarial vocabulary for attacking vulnerabilities of large language model applications," vol. 17, pp. 214–225, Dec. 2024. DOI: 10.5281/zenodo.14680185.

[4] W. Zou, R. Geng, B. Wang, and J. Jia, "PoisonedRAG: Knowledge corruption attacks to retrieval-augmented generation of large language models," in *Proceedings of the 34th USENIX Conference on Security Symposium*, ser. SEC '25, Seattle, WA, USA: USENIX Association, 2025.

[5] E. M. Cabeza-Lopez, R. Ruiz-Gonzalez, A. Merino-Gomez, L. E. Curiel-Herrera, and J. A. Rincon, "A Comparison of AI-Enabled Techniques for the Detection of Attacks in IoT Devices," in *International Joint Conferences*, H. Quintián *et al.*, Eds., vol. 957, Series Title: Lecture Notes in Networks and Systems, Cham: Springer Nature Switzerland, 2024, pp. 227–236. DOI: 10.1007/978-3-031-75016-8_21. [Online]. Available: https://link.springer.com/10.1007/978-3-031-75016-8_21.

[6] Y. Choi, Y. Park, J. Byun, J. Lee, and J. Park, "Safeguarding privacy of retrieval data against membership inference attacks: Is this query too close to home?" In *Findings of the Association for Computational Linguistics: EMNLP 2025*, C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, Eds., Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 8241–8258. DOI: 10.18653/v1/2025.findings-emnlp.438. [Online]. Available: https://aclanthology.org/2025.findings-emnlp.438/.

[7] S. Zeng *et al.*, "The good and the bad: Exploring privacy issues in retrieval-augmented generation (RAG)," in *Findings of the Association for Computational Linguistics: ACL 2024*, L.-W. Ku, A. Martins, and V. Srikumar, Eds., Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 4505–4524. DOI: 10.18653/v1/2024.findings-acl.267. [Online]. Available: https://aclanthology.org/2024.findings-acl.267/.

[8] S. Cohen, R. Bitton, and B. Nassi, "Here comes the AI worm: Preventing the propagation of adversarial self-replicating prompts within GenAI ecosystems," in *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '25, Taipei, Taiwan: Association for Computing Machinery, 2025, pp. 3975–3989. DOI: 10.1145/3719027.3765196. [Online]. Available: https://doi.org/10.1145/3719027.3765196.

[9] Z. Zhong, Z. Huang, A. Wettig, and D. Chen, "Poisoning retrieval corpora by injecting adversarial passages," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

[10] A. Shafran, R. Schuster, and V. Shmatikov, *Machine against the RAG: Jamming retrieval-augmented generation with blocker documents*, version 4, 2025. arXiv: 2406.05870 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2406.05870.

[11] A. Zou *et al.*, *Universal and transferable adversarial attacks on aligned language models*, version 2, 2023. DOI: https://doi.org/10.48550/arXiv.2307.15043. arXiv: 2307.15043 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2307.15043.

[12] S. Cho, S. Jeong, J. Seo, T. Hwang, and J. C. Park, *Typos that broke the RAG's back: Genetic attack on RAG pipeline by simulating documents in the wild via low-level perturbations*, version 1, 2024. DOI: https://doi.org/10.48550/arXiv.2404.13948. arXiv: 2404.13948 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2404.13948.

[13] J. Xue *et al.*, *Badrag: Identifying vulnerabilities in retrieval augmented generation of large language models*, version 2, 2024. DOI: https://doi.org/10.48550/arXiv.2406.00083. arXiv: 2406.00083 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2406.00083.

[14] S. Ball, F. Kreuter, and N. Panickssery, *Understanding Jailbreak Success: A Study of Latent Space Dynamics in Large Language Models*, Oct. 2024. DOI: 10.48550/arXiv.2406.09289. [Online]. Available: http://arxiv.org/abs/2406.09289.

[15] X. Tan *et al.*, "RevPRAG: Revealing poisoning attacks in retrieval-augmented generation through LLM activation analysis," in *Findings of the Association for Computational Linguistics: EMNLP 2025*, C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, Eds., Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 12 999–13 011. DOI: 10.18653/v1/2025.findings-emnlp.698. [Online]. Available: https://aclanthology.org/2025.findings-emnlp.698/.

[16] H. Liu, B. Zhao, J. Guo, K. Zhang, and P. Liu, "A lightweight unsupervised adversarial detector based on autoencoder and isolation forest," *Pattern Recognition*, vol. 147, p. 110 127, 2024, ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2023.110127. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320323008245.

[17] J. Duan *et al.*, "Shifting attention to relevance: Towards the predictive uncertainty quantification of free-form large language models," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, L.-W. Ku, A. Martins, and V. Srikumar, Eds., Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 5050–5063. DOI: 10.18653/v1/2024.acl-long.276. [Online]. Available: https://aclanthology.org/2024.acl-long.276/.

[18] S. Steindl, U. Schäfer, B. Ludwig, and P. Levi, "Linguistic obfuscation attacks and large language model uncertainty," in *Proceedings of the 1st Workshop on Uncertainty-Aware NLP (UncertaiNLP 2024)*, R. Vázquez *et al.*, Eds., St Julians, Malta: Association for Computational Linguistics, Mar. 2024, pp. 35–40. [Online]. Available: https://aclanthology.org/2024.uncertainlp-1.4.

[19] F. Rubenbauer, S. Steindl, P. Levi, D. Loebenberger, and U. Schäfer, "Detection of adversarial prompts with model predictive entropy," in *Findings of the Association for Computational Linguistics: EACL 2026*, Accepted for publication, Rabat, Morocco: Association for Computational Linguistics, 2026.

[20] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, *Mpnet: Masked and permuted pre-training for language understanding*, version 2, 2020. arXiv: 2004.09297 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2004.09297.

[21] Huggingface, *All-mpnet-base-v2*, 2026.03.10. [Online]. Available: https://huggingface.co/sentence-transformers/all-mpnet-base-v2.

[22] Z. Yang *et al.*, "HotpotQA: A dataset for diverse, explainable multi-hop question answering," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[23] W. Zou, R. Geng, B. Wang, and J. Jia, *PoisonedRAG*, 2026.03.10. [Online]. Available: https://github.com/sleeepeer/PoisonedRAG / blob / main / results / adv _ targeted _ results / hotpotqa.json.

[24] A. Grattafiori *et al.*, *The Llama 3 herd of models*, version 3, 2024. arXiv: 2407.21783 [cs.AI]. [Online]. Available: https://arxiv.org/abs/2407.21783.

[25] A. Q. Jiang *et al.*, *Mistral 7B*, version 1, 2023. arXiv: 2310.06825 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2310.06825.

Bridging the Gap: A Linear Algebra Unit for Critical Infrastructure Defense

Donna Beers
Department of Mathematics
Simmons University
Boston, Massachusetts
e-mail: donna.beers@simmons.edu
Clifton P. Morrow
Taylor Business Institute
Chicago, Illinois
e-mail: clifton.morrow@tbiil.edu

***Abstract*—The critical infrastructure sector faces a compounding crisis: a widening supply-demand gap for skilled cyber-defenders coinciding with a paradigm shift in offensive tradecraft. Adversarial campaigns like Volt Typhoon have moved beyond deploying malicious code to living off the land – abusing legitimate system tools to evade detection. This widening offense-defense gap renders traditional signature-based tools insufficient, necessitating a workforce capable of behavioral anomaly detection. However, current training often limits analysts to tool identification, leaving them ill-equipped to analyze novel threats. This industry report introduces a modular training unit designed to bridge these strategic gaps. We propose a two-fold, complementary approach aligning linear algebra with the National Institute of Standards and Technology (NIST) Cybersecurity Framework: first, the Geometric Approach (Detection), where trainees use k-Nearest Neighbors (k-NN) on a dataset to map benign traffic topology, modeling the geometry of automated alerting; second, the Algebraic Approach (Hunting), where trainees apply Singular Value Decomposition (SVD) to identify pattern of life deviations, modeling the proactive hunting required for advanced persistent threats. By grounding Artificial Intelligence (AI) concepts in their mathematical roots, this architecture aims to produce a workforce capable of dissecting and trusting the algorithms protecting critical infrastructure.**

Keywords-cybersecurity education; workforce development; anomaly detection; threat hunting; linear algebra; critical infrastructure defense.

I. Introduction

State-sponsored campaigns like Volt Typhoon [1] represent a fundamental shift in threats and countermeasures in critical infrastructure systems, moving beyond malware to living off the land techniques that evade signature detection. To counter this threat, our work leverages Artificial Intelligence and Security, specifically applying Artificial Intelligence (AI) for threat and anomaly detection through the mathematical lenses of k-Nearest Neighbors (k-NN) [2] and Singular Value Decomposition (SVD) [3]. The nature of the threat and the mathematical content of the tools create a significant workforce capability gap where the complexity of modern threats outpaces the cognitive tools available to junior analysts, a problem only partially addressed by existing certifications. Furthermore, by elevating analyst capability from rote memorization to first-principles understanding, this unit directly addresses usability and awareness in secure systems, ensuring the workforce can effectively operate these advanced defense architectures.

To address these capability gaps, we propose a modular training unit that satisfies three core operational criteria. First, the solution is understandable to early-career analysts by utilizing a geometric-first pedagogy that visualizes high-dimensional anomalies before introducing algebraic formalism. Second, the approach is defendable as a necessary evolution of tradecraft; relying on black-box tools is a liability against living off the land attacks, whereas first-principles mathematical analysis provides a robust audit trail for detection logic. Finally, the framework is feasible for rapid institutional adoption, relying on open-source tools (Python, Jupyter) and privacy-neutral synthetic datasets that require no proprietary infrastructure.

In alignment with the conference targets, this work presents an architectural solution to the workforce gap, supported by a practical implementation of a modular curriculum. While the theoretical foundations of k-NN and SVD are well-established, our contribution lies in the novel architectural synthesis of these methods for critical infrastructure defense.

In Section I, we note gaps in the preparation of early-career cybersecurity analysts. The goal of our module is to bridge those gaps.

The remainder of the paper is organized as follows: In Section II, we note that the Computing Technology Industry Association Cybersecurity Analyst (CySA+) [4] and the SANS Institute GIAC Certified Intrusion Analyst (GCIA) [5] certifications require arithmetic, and existing curricular modules abstract away the mathematics. In Section III, we present a case study demonstrating our approach. In Section IV, we evaluate the viability of this instructional architecture against key industry metrics. In Section V, we compare our approach with prior art and propose future work.

II. Related work | Methods

1) The Analytical Gap in Technical Certifications: While the National Institute of Standards and Technology (NIST) Cybersecurity Framework [6] emphasizes the continuous function of detection, standard industry curricula for security practitioners leave a critical gap in algebraic fluency. Highly technical, analyst-focused certifications—such as CySA+ and GCIA –

represent the industry standard for training Security Operations Center personnel. However, their curricula predominantly focus on rule-based heuristics, packet-level signature matching, and the operational usage of Security Information and Event Management dashboards.

In these technical tracks, behavioral anomaly detection and machine learning are frequently introduced only conceptually, treating algorithmic defense as a proprietary black-box appliance. Consequently, trainees learn to interpret alerts but lack the linear algebra required to understand the underlying manifold geometry [2]. As adversaries increasingly deploy adversarial machine learning and protocol mimicry [7] to evade standard signatures, defending critical infrastructure requires moving analysts from operational monitoring to algorithmic engineering (Bloom's Taxonomy Levels 4 and 5: Analyzing and Evaluating [8]). By forcing students to manually calculate eigendecompositions and geometric thresholds, this instructional architecture provides direct pedagogical insight into the mechanics of false positives and false negatives, empowering the future workforce to actively tune detection models rather than blindly trusting opaque vendor alerts.

2) The Mathematical Gap in Curricula: Existing curricular modules in the CLARK repository, such as Serra's comprehensive guide to anomaly detection [9], focus on the implementation of high-level algorithms like isolation forests and Support Vector Machines (SVM). While excellent for application, these modules often abstract away the underlying mathematical machinery. Our work differentiates itself by focusing specifically on the linear algebra foundations (SVD and geometric topology) that precede these advanced algorithms, filling the pedagogical gap between basic mathematics and black-box AI application.

3) The Risk of Black-Box Models: Engineers dislike black boxes they cannot fix. From an engineering perspective, black-box models introduce unacceptable liability in safety-critical systems. As noted in NIST Interagency Report (IR) 8312 [10], trust in AI requires that outputs be "meaningful" and "explainable" to the operator. Engineers do not reject AI out of prejudice; they reject it out of adherence to these rigorous risk management principles. Many current AI security tools are black boxes. When they throw a false positive, the analyst cannot fix it. Linear algebra is the screwdriver. Beyond serving as a mathematical theorem, SVD provides a mechanism to 'tune the noise filter' (finding σ thresholds). Similarly, k-NN acts as more than a mere topology; it provides a framework to 'calibrate the sensitivity' (by choosing k and the distance metric). We frame linear algebra not as abstract theory, but as the component engineering of the detection engine. Just as a mechanical engineer must understand thermodynamics to tune an engine, a cyberdefender must understand the SVD factorization to tune the signal-to-noise ratio of an anomaly detector.

III. Experimental Design and Case Study

"Stop whatever you're doing. Look at it from the other side" [11]. We designate our synthetic dataset the Janus Matrix, named after the Roman god of beginnings and transitions who possessed two faces looking in opposite directions [12]. Just as the mythological Janus held the key to distinct realities, our dataset is designed to be unlocked via two distinct mathematical perspectives: the geometric topology of k-NN and the algebraic variance of SVD.

A. Experimental Design

In alignment with the taxonomy established by Chandola et al. [13], our architecture adopts a hybrid detection strategy. We utilize k-NN to detect proximal anomalies (points distant from local neighborhoods) and SVD to detect spectral anomalies (points violating the global correlation subspace). This ensures the analyst can identify threats that might be invisible to a single mathematical modality.

B. Case Study

To demonstrate the mathematical mechanics of anomaly detection, we construct a Janus Matrix ($X \in R^{7\times 2}$) using a deliberately constrained micro-slice of the Knowledge Discovery and Data Mining (KDD) Cup 1999 dataset [14]. While modern datasets like UNSW-NB15 [15] are necessary for benchmarking production AI, their high dimensionality makes them pedagogically opaque. We specifically select the KDD '99 dataset because its HTTP traffic features (src_bytes vs. dst_bytes) provide an intuitive, rank-1 linear correlation (Request $\propto$ Response). By constraining the matrix to 7×2, learners are able to perform the k-NN distance calculations and the SVD eigendecomposition by hand, successfully converting a black-box machine learning algorithm into a glass-box mathematical exercise [16].

Furthermore, these specific features establish a clear adversary threat model: a **Signature-Proof Data Exfiltration** attack. In this scenario, the adversary tunnels data out of the network by strictly mimicking valid minimum and maximum byte bounds to evade standard firewall heuristics. However, in doing so, they break the fundamental structural asymmetry of normal web browsing, allowing algebraic detection methods to succeed where geometric thresholds fail.

$$X = \begin{bmatrix} 220 & 1200 & \leftarrow \text{A} \\ 240 & 1350 & \leftarrow \text{B} \\ 260 & 1500 & \leftarrow \text{C} \\ 280 & 1650 & \leftarrow \text{D} \\ 300 & 1800 & \leftarrow \text{E} \\ 320 & 1950 & \leftarrow \text{F} \\ \mathbf{290} & \mathbf{1300} & \leftarrow \text{Q = query point to analyze} \end{bmatrix}$$

Glancing at the numbers (like a signature-based tool), Q seems benign, because a src_bytes value of 290 resides in the benign range ([220,320]), and a dst_bytes value of 1300 resides in the benign range ([1200,1950]).

1) Geometric Analysis (k-NN): In the geometric modality, we visualize the data in R^2 (Figure 1). We calculate the distances in Table I. With $k = 3$, the nearest neighbors are $\{B, A, C\}$. A naïve voting algorithm would classify Q as "Benign".

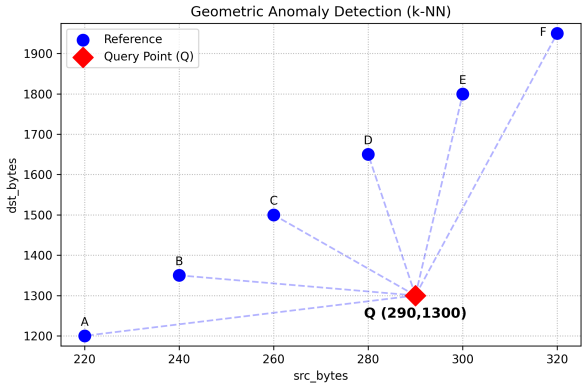


Figure 1. Visualization of the 7-point dataset. The query point Q is geometrically distant from the reference points even though its coordinates reside in the benign range of both dimensions.

TABLE I. EUCLIDEAN DISTANCES (EXCERPT: CLOSEST, MEDIAN, FURTHEST)

Point	Coord	Distance $d(Q, P_i)$
B	(240, 1350)	$\sqrt{50^2 + 50^2} \approx \mathbf{70.7}$
D	(280, 1650)	$\sqrt{10^2 + 350^2} \approx \mathbf{350.1}$
F	(320, 1950)	$\sqrt{30^2 + 650^2} \approx \mathbf{650.7}$

A more thorough application of k-NN also fails on this dataset. Table II shows that the average internal cohesion of the reference set is 353.10, while the query point Q resides at an average distance of 315.99. Hence we have $\bar{d}_Q < \bar{d}_{ref} \longrightarrow$ false negative.

Because Q resides in the bounding box (hiding in the gap between D and E), Euclidean distance algorithms classify it as a benign inlier. This confirms that for sophisticated tunneling attacks that respect min/max boundaries, geometry is insufficient. Detection requires the algebraic covariance check provided by SVD, which detects the structural violation despite the geometric proximity.

2) Algebraic Analysis (SVD): While k-NN fails to distinguish the anomaly Q from the reference cluster (due to $\bar{d}_Q < \bar{d}_{ref}$), the SVD reveals the structural violation. We define the reference matrix X using the mean-centered KDD micro-slice.

3) Algorithmic Synthesis: Spheres vs. Cylinders: To address the comparative efficacy of detection algorithms, it is critical to understand why neither k-NN nor SVD universally dominates. We selected k-NN as our baseline because it intuitively models *radial geometric bounds* (a convex hull or sphere around normal behavior [2]). SVD, conversely, models *orthogonal variance limits* (a cylinder projected along the principal component).

TABLE II. INTERNAL COHESION OF REFERENCE SET (EXCERPT)

Point i	Point j	Distance $d(P_i, P_j)$
A	**B**	151.33
B	**C**	151.33
...	...	...
A	**E**	605.31
A	**F**	756.64

As demonstrated in Table II, a Protocol Tunneling attack easily defeats k-NN by hiding within the spatial gap of the reference sphere. However, SVD detects the anomaly because the point deviates from the cylindrical axis.

Conversely, consider the *Opposite Case*: a massive data exfiltration attack that perfectly maintains the valid HTTP Request-to-Response ratio (e.g., $(3000, 22500)$). Because this query point lies exactly on the v_1 subspace, its projection error onto the v_2 noise axis is near zero. SVD yields a false negative, failing to flag the massive volume. Here, k-NN successfully triggers an alert, as the point's magnitude places it thousands of units outside the reference sphere.

Therefore, robust critical infrastructure defense requires a hybrid detector that enforces the intersection of both constraints: the algebraic correlation of the SVD cylinder, and the geometric magnitude bounds of the k-NN sphere.

a) Subspace Decomposition: The covariance structure of the HTTP traffic is decomposed into its principal components ($X_{cov} = V\Lambda V^T$) [2]. Unlike the geometric proximity check of k-NN, SVD identifies the invariant subspace (the linear correlation between src_bytes and dst_bytes).

$$V^T \approx \begin{bmatrix} 0.13 & 0.99 \\ -0.99 & 0.13 \end{bmatrix} \begin{matrix} \leftarrow v_1 \text{ (Traffic Pattern)} \\ \leftarrow v_2 \text{ (noise axis)} \end{matrix}$$

Figure 2 illustrates how the first vector v_1 captures the valid traffic law (Request $\propto$ Response). The second vector v_2 represents the forbidden orthogonal variance.

b) The Projection Test: We project the centered query point Q_c onto the noise axis v_2.

$$\text{Score} = |Q_c \cdot v_2| = |(20)(-0.99) + (-275)(0.13)| \approx \mathbf{55.6}$$

This high projection score (55.6) contrasts sharply with the reference set, which has near-zero projection on v_2. Thus, SVD mathematically reveals the anomaly by detecting the variance violation, succeeding where geometric distance failed.

4) Complexity Analysis (Defense Against Heuristics): A common critique in low-dimensional detection is the Ratio Heuristic – why not simply calculate y/x? While effective in R^2, heuristics fail in high-dimensional conservation tasks (e.g., Equal-Cost Multi-Path Routing [17] in R^3, where $x + y + z = C$).

- Configuration Cost
  Heuristics require analysts to manually derive conservation rules ($O(1)$ compute, $O(\text{Human})$ cost). SVD learns the null space automatically.
- Combinatorial Explosion
  To replicate SVD's coverage in R^{100}, a heuristic system would need to evaluate $\binom{100}{2} = 4,950$ pairwise ratios. Our SVD architecture evaluates the entire feature space simultaneously, evaluating a new event in $O(n)$ time without human configuration, making it the superior choice for complex network defense.

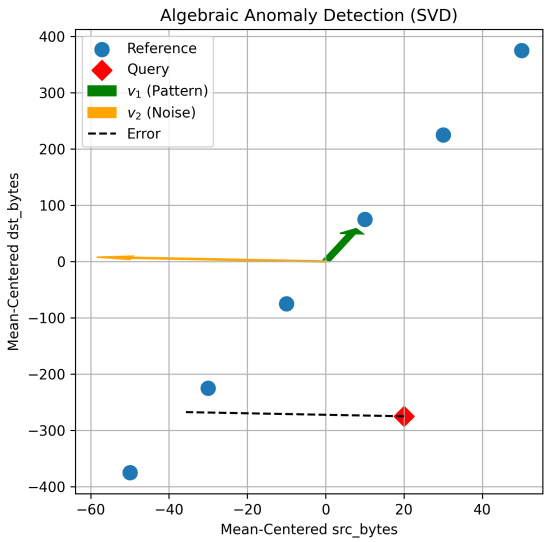


Figure 2. Visualization of the mean-centered data and the eigenvectors (scaled by $50\times$ for visualization)

IV. Discussion | Evaluation

To assess the viability of this instructional architecture, we evaluate the proposed solution against three key industry metrics: implementation practicality, computational scalability, and workforce alignment.

A. Practicality

The Zero Infrastructure Metric. Unlike traditional cybersecurity training that requires expensive cyber ranges or proprietary virtualization hardware [18], our modular architecture relies entirely on open-source scientific computing tools (Python, NumPy, Jupyter). This ensures high practicality for under-resourced institutions, as the curriculum can be deployed on standard student laptops or free cloud environments (e.g., Google Colab) with zero licensing costs.

B. Scalability

Algorithmic Efficiency. From a technical perspective, the Janus Matrix approach demonstrates superior scalability compared to deep learning alternatives. Traditional deep neural networks often require massive computational overhead to train. In contrast, our architecture utilizes SVD to ensure minimal resource consumption. The algorithmic complexity is strictly bounded by $O(\min(mn^2, m^2n))$, where m is the number of network events and n is the number of features [3]. This allows the module to be scaled across thousands of endpoints or trainees without necessitating GPU clusters, ensuring the barrier to entry remains low.

C. Strategic Benefits

Alignment with the National Institute of Standards and Technology (NIST) National Initiative for Cybersecurity Education (NICE). The primary benefit of this contribution is its direct mapping to the NIST NICE Workforce Framework for Cybersecurity (Special Publication 800-181) [19]. Specifically, it bridges the gap for the cyber defense analyst (PR-CDA-001) work role by moving beyond the defined Knowledge, Skills, and Abilities (KSAs) of tool operation into the implicit requirement for mathematical reasoning.

D. Future Validation

Future work will validate the pedagogical efficacy through a pre-test/post-test instrument measuring student self-efficacy in interpreting false positives. This study will be conducted under institutional review to quantify the shift in Bloom's Taxonomy [8] levels among participants.

V. Conclusion and Future Work

We have presented a modular instructional architecture that addresses the critical workforce gap in behavioral anomaly detection. By anchoring the curriculum in the Janus Matrix dataset and the SVD factorization, we provide a mathematical foundation often absent in standard industry certifications.

A. Comparison with Prior Art

When comparing this architecture to existing curricular modules, such as Serra's work in the CLARK repository, a distinct divergence in pedagogical outcomes is evident. The prior art results in a workforce proficient in the application of high-level algorithms (e.g., isolation forests, SVMs) via Python libraries. While valuable for rapid deployment, this approach treats the detection engine as a black box, creating liability when facing adversarial evasion.

In contrast, our results demonstrate that a glass-box approach – starting with the geometry of k-NN and the linear algebra of SVD – equips the analyst to audit and explain the model's decisions. This alignment with NIST IR 8312 explainability principles suggests that while the prior art optimizes for implementation speed, our architecture optimizes for defensive resilience.

B. Future Work

Future phases will deploy this module across the partner institutions to capture quantitative data on cybersecurity practitioner self-efficacy. However, the immediate architectural result is a scalable, open-source framework that demystifies the mathematics of critical infrastructure defense.

References

[1] CISA, NSA, and FBI, "PRC state-sponsored actors compromise and maintain persistent access to U.S. critical infrastructure", Cybersecurity Advisory AA24-038A, 2024, [Online]. Available: https://www.cisa.gov/news-events/cybersecurity-advisories/aa24-038a (visited on 01/23/2026).

[2] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: John Wiley & Sons, 2001.

[3] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th ed. Johns Hopkins University Press, 2013.

[4] CompTIA, "CySA+ (V3) exam objectives summary", 2023, [Online]. Available: https://www.comptia.org/en-us/certifications/cybersecurity-analyst/#objectives (visited on 03/06/2026).

[5] GIAC, "GIAC Certified Intrusion Analyst Certification (GCIA)", 2026, [Online]. Available: https://www.giac.org/certifications/certified-intrusion-analyst-gcia/ (visited on 03/06/2026).

[6] National Institute of Standards and Technology, "Cybersecurity framework", Version 2.0, National Institute of Standards and Technology, 2024, [Online]. Available: https://www.nist.gov/cyberframework (visited on 01/23/2026).

[7] D. Wagner and P. Soto, "Mimicry attacks on host-based intrusion detection systems", in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, ACM, 2002, pp. 255–264.

[8] L. W. Anderson and D. R. Krathwohl, Eds., *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. New York: Longman, 2001, ISBN: 978-0801319037.

[9] E. Serra, "Anomaly and novelty detection", CLARK Curriculum Module, 2024, [Online]. Available: https://clark.center/details/edoardoserra/aa4b123a-d0e5-4e7a-93c3-edd8e57562f5/0 (visited on 01/23/2026).

[10] P. J. Phillips *et al.*, "Four principles of explainable artificial intelligence", 2021, [Online]. Available: https://nvlpubs.nist.gov/nistpubs/ir/2021/NIST.IR.8312.pdf (visited on 01/23/2026).

[11] M. Aurelius, *Meditations*, Koine Greek, trans. Koine Greek by M. Hammond. New York: Penguin Classics, 2014.

[12] P. Ovidius Naso, *Fasti* (Oxford World's Classics), A. Wiseman and P. Wiseman, Eds. Oxford: Oxford University Press, 2004, See Book I for Janus.

[13] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey", 2009, [Online]. Available: https://dl.acm.org/doi/10.1145/1541880.1541882 (visited on 01/24/2026).

[14] UCI Machine Learning Repository, "Kdd cup 1999 data", Subset: HTTP src_bytes vs dst_bytes, 1999, [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (visited on 01/24/2026).

[15] N. Moustafa and J. Slay, "Unsw-nb15: A comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)", in *2015 Military Communications and Information Systems Conference (MilCIS)*, IEEE, 2015, pp. 1–6.

[16] X. Liu *et al.*, "From black box to glass box: A practical review of explainable artificial intelligence (xai)", 2025, [Online]. Available: https://www.mdpi.com/2673-2688/6/11/285 (visited on 03/06/2026).

[17] C. Hopps, "Analysis of an Equal-Cost Multi-Path Algorithm", 2000, [Online]. Available: https://www.rfc-editor.org/info/rfc2992 (visited on 01/27/2026).

[18] NIST and NICE, "The cyber range: A guide", 2023, [Online]. Available: https://www.nist.gov/system/files/documents/2023/09/29/The%20Cyber%20Range_A%20Guide.pdf (visited on 01/23/2026).

[19] National Initiative for Cybersecurity Education (NICE), "Workforce framework for cybersecurity (nice framework)", Nov. 2020, [Online]. Available: https://csrc.nist.gov/pubs/sp/800/181/r1/final (visited on 01/23/2026).

An Analysis of Malware Threats Facing the IoT:
A taxonomy of IoT malware

Mr Ross Heenan
Cybersecurity and Computing
Abertay University
Dundee, United Kingdom
email: r.heenan@abertay.ac.uk

Prof Ian Ferguson
Cybersecurity and Computing
Abertay University
Dundee, United Kingdom
email: i.ferguson@abertay.ac.uk

Dr Laith Al-Jobouri
Games technology and maths
Abertay University
Dundee, United Kingdom
email: l.al-jobouri@abertay.ac.uk

Abstract – **From its origins in "Weiser's vision" in the 90's there has been widespread adoption of Internet of Things (IoT) technologies across consumer, commercial, industrial, logistics, utilities, and healthcare environments. This has brought with it an expansion and rise in vulnerabilities and malware threats due to an increase in potential vectors of attacks and the general discovery, privacy and security issues facing IoT solutions. Many IoT systems across these sectors contain a mixture of Information Technology (IT) and Operational Technology (OT) network segments and many existing legacy networks of systems have incorporated or adopted IoT components or services into their networks. These are generally built with compatibility and operability in mind with security usually less considered or as an afterthought. A large variety of IoT malware exists including botnet and Denial of Service (DoS) variants, brickers, cryptominers, ransomware, stalkerware and Industrial Controller Systems (ICS) malware variants and there can also be significant threat leveraged from IoT malware to certain critical systems or services being controlled or monitored. Threats can also be leveraged from a compromised IoT system and used for deployment of threats including DoS reflection or amplification attacks. The research presented here provides a review and threat analysis of the varieties of IoT malware that currently exist with a case analysis and comparison of two high profile ICS capable targeting malware known as BlackEnergy and Industroyer. These variants were responsible for the attack and compromise of energy utilities providers networks in Ukraine between 2014 and 2022 and show examples of malware threats using different routines to gain compromise of critical ICS systems.**

Keywords – IoT Malware; IoT Threat analysis; IoT security; BlackEnergy; Industroyer

I. INTRODUCTION

Malware has remained a persistently increasing threat to computer systems and networks from the first computer worms and viruses in the 1980's, to the evolution of variants utilising more advanced techniques including fileless deployment, process ghosting or hollowing or targeted Common Vulnerability Exposure (CVE) use in more recent years. The past four decades have seen the birth of the internet, the .com boom of the 90's, the emergence of Cloud computing and Blockchain technologies, revisiting Artificial Intelligence (AI) technologies and the introduction of IoT. This has brought an evolution in types of threat and seen the emergence of targeted types of malware including ransomware, stalkerware, brickers, cryptominers and ICS malware among others. IoT has seen rapid growth in recent years with mass adoption of IoT solutions into many aspects of life including automotive, consumer, commercial, industrial and healthcare solutions with estimates of over 14 billion devices to be in use by 2023 [2]. A significant fraction of these are estimated being deployed as consumer products, utilities management and monitoring solutions and also automotive, asset and logistics tracking and management and monitoring solutions. Cisco also estimates a future global market value of around $14 trillion [3]. Thus, there is a clear requirement for priority of provision in appropriate protection for these assets.

There are many challenges in developing and maintaining an effective security posture for IoT systems or networks including lack of capability or availability of computational resource, ensuring compatibility for cross platform systems and technologies or other constraints concerning case specific operational or environmental requirements. The adoption of numerous communications technologies and protocols, such as purpose built lightweight protocols like Low Power Wide-Area (LPWA) or Long Range Wide-Area (LoWRAN) and also the wider use of communication protocols in general including Cellular communications like 3, 4 and 5th Generation (5G), Narrow-band IoT (NB-IoT), Extended Coverage GSM (EC-GSM), Satellite, Long-Range (Lora), Message Queuing Telemetry Transport (MQTT), Constrained Application Protocol (CoAP), Multicast DNS (mDNS), Bluetooth, ZigBee and various others means there is a wider footprint to consider in terms of consideration of threat and provision of secure systems and network communications.

Since the emergence of IoT technologies there have been significant and growing numbers of vulnerabilities and cases of exploitation being identified or reported concerning IoT systems or solutions [4]. The common types of threat being initially targeted and leveraged in these cases include ransomware, Distributed DoS (DDoS), brickers, botnets and also cross-platform malware or threats, many of these were able to be leveraged due to lack of awareness in implementing appropriate security mechanisms and often products of insecure development practice in the products themselves [4]. Insecure development practices include the use of insecure, predictable or hardcoded default settings or credentials and the use of insecure technologies or ecosystem interfaces, such as Application Programming Interface (API's).

Whether an IoT system is targeted in a consumer, commercial, industrial, healthcare or other type of setting there can be a variety of severity of threat facing it ranging to severe and even life-threatening from potential compromise. There has been widespread incorporation of IoT solutions in sensitive or critical settings across industry sectors including utilities, industrial and healthcare. In these settings, one vulnerable IoT endpoint could allow for propagation of threat which dependent on the system concerned could manifest into a significant increase in damage or disruption. There have

already been severe examples of the exploitation of critical IoT systems through the use of malware. One of the first recognised examples of this would be the Stuxnet incident in 2010 where malware was used to compromise critical ICS systems in an Iranian Nuclear facility [64] in Natanz, with many researchers and officials suggesting the incident being attributed to a collaborative operation between United States and Israeli forces.

A more recent example of the use of malware in compromising IoT systems is the Mirai botnet malware which was reported in one instance to have compromised around 300,000 devices to leverage a botnet to deploy various DDoS flooding attacks that were reported to reach volumes of over 1 Terabits per second (Tbps) [19]. Other examples include the Bricker malware variants Brickerbot and Silex that corrupt or overwrite areas of compromised devices to leverage DDoS or Permanent DoS (PDoS) or ICS capable targeting malwares including the Lemonduck, Industroyer or BlackEnergy variants. These will be discussed further in the following sections of the paper.

Due to the varied nature of the IoT landscape and also the widespread use of certain types of devices or technologies, another factor of compounded threat is the increasing emergence of CVE threats against widely used software, hardware or communication technologies like Wireless, 4G, 5G or Bluetooth. A good example of this is the recently discovered vulnerability in the Thales m2m module (CVE-2020-15858) [5], which is widely used in Wireless IoT technologies. Vulnerable widely used software has also been shown to be an example of this in the various OpenSSH vulnerabilities [6] that allow compromise of endpoints for use in generation and forwarding of malicious traffic or the recent Key Reinstallation Attack (KRACK) vulnerability [7] that allows reinstallation of cryptographic keys to hijack wireless connections. There have also been vulnerability disclosures in Bluetooth technologies, such as the Bluetooth Impersonation Attacks (BIAS) [8] or Key Negotiation of Bluetooth (KNOB) [9] CVE disclosures and also vulnerabilities in cellular technologies including potential interception and eavesdropping through exploitation of the Recovering encrypted Voice over LTE (ReVOLTE) vulnerability [10] or other methods.

Another significant issue is the emergent threat to IoT systems of cross platform malware, these variants have the ability to compromise different architectures and types of systems. Combination of such variants with bricker or ransomware payloads could represent a notably increased threat to an IoT system. Malware targeted at IoT presents a heightened threat towards potential intrusion through a single vulnerable endpoint's presence, a wider variety of potential CVE threats and a likely increased number of devices to target. This all provides a much larger threat surface that malware authors are capable of utilising in targeting and development of a malicious threat payload in order to translate to as substantial compromise as possible which can often amount to the compromise of millions of IoT devices. There has already been a collection of malware threats discovered successfully leveraging exploitation of IoT systems, such as bricker variants including, Brickerbot [11] and Silex [12], botnet malware including Mirai [13] or Hajime [14] and also ICS or Supervisory Controller and Data Acquisition (SCADA) threat capable malware including the LemonDuck [15], Industroyer [16] and BlackEnergy [17] variants.

Malware threats are capable of deployment against a target using a wide range of exploitation methods with popular techniques including the use of Remote Code Execution (RCE), process injection techniques or fileless compromise methods, or the use of social engineering, phishing campaigns or Hypertext Markup Language (HTML) smuggling to leverage deployment of malicious payloads [20].

It is also important to note that some variants will use what is known as a zero day, this is an exploitation method that has not been previously observed or recorded.

There are various methods that can be used for the analysis of malware, these are categorised into static, dynamic and hybrid techniques. Static analysis of a malware source is where a malware payload is analysed using passive techniques like hashing and file header, strings, dependencies and source analysis. Dynamic analysis of a malware source will utilise behavioural analysis techniques to monitor system and network activity during the malwares execution. There are also hybrid analysis methods that utilise other advanced techniques like disassembly where the source code can be reverse engineered and memory analysis or debugging which will allow live interaction with a running program or in this case malware. These methods allow extraction of evidential artefacts, such as hashes of unique signatures of files and Indicators of compromise (IOC's) which are unique system or network evidential artefacts that show evidence of compromise. These can be used to create what is known as Yara detection rules which can be used by Intrusion Detection Systems (IDS) and Anti-virus software as detection rules for classification. Using dynamic methods like memory or behavioural analysis and reverse engineering the tactics, techniques and procedures (TTP's) used by a malware threat can also be distinguished which again can be used for response, classification and detection.

IoT malware has the advantage of being able to target a larger threat surface in exploitation of IoT systems, this is due to the larger numbers of limited capability devices which constitute a typical IoT architecture and also a wider variety of hardware, software or networking technologies being present to target in comparison to a traditional computer network.

This can result in a higher potential in commonality of presence of vulnerable software, hardware or network technology components due to the increased amounts of devices or components in some IoT systems. This can create the potential for the threat to be exponentially increased particularly the more evolved that IoT malware variants become. The FLocker [42] Android ransomware, Trickbot [43] botnet malware and the LemonDuck [15] [44] cryptomining malware are recent examples of these types of evolving malware with modular or advanced capabilities.

There have been sustained discoveries of vulnerabilities across consumer, commercial, industrial, healthcare and other IoT products in recent years with various high-profile vendor products being concerned including Amazon [26], Apple [27], Fitbit [28], Google [29] [30] and Ring [30]. There have also been vulnerabilities discovered in various automotive manufacturers products, such as BMW [32], Chrysler [33], Ford [34], Honda [35], Jeep [33] and Tesla [36] vehicles that allowed intrusion through keyless hacks or replay attacks, compromise of the vehicles Electronic Control Unit (ECU), hijacking control of the vehicles lights, wipers, locks,

dashboard readings and even the vehicle's steering, gearing, and braking controls.

The evolution of the IoT has seen with it the emergence of a variety of malware threats including botnets, cryptominers, brickers, and ransomwares with recent case examples including the well-known and now evolved Mirai botnet malware [21], the Darlloz and Muldrop cryptominer malwares [22], the Brickerbot [23] and Silex [22] bricker malware variants and also recent ransomware compromises against high profile consumer targets including the WastedLocker Ransomware [24] used in the attack against Garmin in 2020 [25].

Many initial examples of compromises were often products of insecure development or deployment, however there is also an increasing use of CVE's [62], zero-day threats or more advanced threat vectors being utilised for example cross-platform malware or complex malware with modular or advanced capabilities [15] [45] [46] [47].

An additional concerning threat to the IoT is the emergence of collections of vulnerabilities identified in widely used Transmission Control Protocol/Internet Protocol (TCP/IP) stack libraries by IoT devices. These include the Ripple20 [37], NAME:WREK [38], NUCLEUS13 [39], ANMESIA33 [40], and URGENT/11 [41] disclosures. These collections of threats contain differing amounts of vulnerabilities in a number of vendors products including HP, Schneider Electric, Intel, Rockwell Automation, Caterpillar, Baxte and many others across medical, transport, industrial, commercial, utilities, and consumer IoT products. These concern CVE's in protocol or TCP stack vulnerabilities that allow leveraging of DoS, corruption of memory and code execution capabilities against a target and are estimated to amount to billions of devices being at risk.

The collection of previously discussed issues presents an expanded threat surface facing these environments which can imply an increased severity or wider propagation of threat to a target network through the presence of a single vulnerable endpoint making malware a more pervasive threat to IoT systems or networks in comparison to traditional computer networks.

This paper aims to present a review of the variance of the malware threats facing the IoT and provide understanding of the evolvement of the threat landscape and the range and severity of potential compromise possible.

This will be achieved by providing a taxonomy of the range of existing IoT malware threats in a review of emergent trends and research with analysis of example variants and a case study of analysis of two recent significant malware threats to industrial IoT in the BlackEnergy and Industroyer malware variants. This research aims to provide the reader with an awareness of the malware threat landscape facing the IoT and an awareness of the capabilities that these variants of malware can possess. It also aims to improve levels of confidence in capability of analysis and ability of identification of countermeasures or advisory measures if performing analysis of similar types of threats.

The remainder of this paper will be structured as follows, the following section will look at an examination of related academic literature. Subsequent sections will include a case study analysis section looking at the analysis of two IoT malware variants. This will be followed by a discussion section providing summative and critical discussion of the research explored and the findings after which a short conclusion section is provided.

II. LITERATURE REVIEW

The following section will review related research in the fields of IoT malware and attacks against IoT solutions as well as recent proposed approaches or solutions for detection and analysis of IoT malware.

A particular issue with malware threats and analysis is attribution. Research by Jinchin Choi et al [60] looks at the area of IoT malware with a concentration on vulnerable endpoints and dropzones used in the deployment of malware threats to IoT endpoints. The study uses 2.423 IoT malware samples from 4 families of malware including Gafgyt, Tsunami and LightAidra [61]. By reverse engineering the samples using the Radare2 open-source framework and other tools including Censys, Shodan and UltraTools the extraction of strings provided identification of dropzones and target IP addresses. These were used to identify patterns or commonalities in geolocation or ports and services used in the samples and also to identify commonalities in targeted CVE threats.

A dropzone is an area of network address which is categorised and grouped by geographical location. Dropzone addresses were identified and categorised through entries noted as being linked to specific data transfer technologies. Target network addresses are listed also. These addresses are then grouped by the family of malware. This helps to identify common Indicators of compromise IoC's and highlights the potential implications that a compromise can have in regard to the scale or severity depending on the system compromised and highlights how large scale distributed and decentralised attacks can be enabled through propagation, amplification or reflection. It is important to identify IOC’s as these can map to suspicious or malicious host, network or file-based behaviours and dictate the level of potential severity an exploitation of these is capable of. Some of the dropzone address are noted as not having current data, this could mean that some of these are networks could have either been taken down or modified due to being previously compromised but also as the study notes this can mean that a dropzones operation can be dynamic or "short-lived - long enough to carry out an attack and short not to be detected". This highlights that the speed of analysis is important to ensure the most complete and accurate understanding of activity and behaviour can be captured.

A large number of target network addresses are noted as being shared between dropzones potentially due to attackers using the same methods of acquiring targets or this could also be possible due to the same malware being hosted or passed from different dropzones. This research provides a good understanding of the types of vulnerability being exploited in IoT endpoints and shows the top 5 dropzones per number of target IP addresses include the UK, US, Canada and France. The results also showed a large distribution of US based target entries pointing to China, Vietnam and Brazil. Research from Imperva Incapsula states that these 3 countries were heavily affected by compromises from the Mirai botnet malware. This is useful as it allows patterns in targeting and deployment used by the 4 malware families to be identified. The analysis is useful but is considered limited due to the restricted and slightly dated collection of variants used.

The broad subject of IoT Attacks and Malware is explored in research by Anand Mudgerikar and Elisa Bertino in [2], the work explores the types of existing malware and the methods of exploitation they use in attack and classifies types of attack

on IoT systems and networks into the categories of Passive/Information Stealing Attacks, Service Degradation Attacks and botnet-based attacks. The study notes the emergence of variants of IoT ransomware or jack-ware and that these variants of malware differ from traditional variants that will simply perform a DoS until the demanded ransom payment is provided. Many of the emerging examples will generally perform full-disk encryption or overwrite critical areas of memory to completely disable the system or service. This is bricker malware, the recent Brickerbot and Silex variants are two of the first examples of this type of variant. The research also provides further analysis of the methods used in the exploitation of IoT targets by malware categorised as Degradation based attacks, Network level attacks and Application-level attacks.

The work presents an informative look at some of the types of existing threat and variants of malware facing IoT systems and networks. Jamming, Node Tampering, Tag cloning, such as Radio Frequency Identification (RFID) and Injection are noted as types of Degradation-based attacks. Insider based attacks, Sinkhole, routing, flooding, injection and authentication-based attacks are listed for Network level attacks and for Application-level based attacks list flooding and types of malware including cryptomining, ransomware and brickers as being common trends. One recommendation is the use of AI based IDS solutions for detection of components or patterns from kill chains and identification of reconnaissance, intrusion, exploitation, lateral movement, obfuscation, or ex-filtration parts of an attack, although this type of capability is not always available.

In "An Analysis of the use of CVE's by IoT Malware" by Raphaël Khoury et al [62], research explores an analysis of 27 variants of IoT malware that have emerged between 2008 to 2019 with the concentration on their use of known or recorded threats known as CVE's. Thirteen of the variants analysed used methods such as dictionary attacks, abuse of insecure development features or default configurations and 1 variant used an unrecorded vulnerability exposure although it may just be that the CVE is not contained in the database or data source used.

The main types of threat recorded as being leveraged against IoT systems are identified as being DoS, sabotage, espionage and cryptomining. A list of botnet malware is also provided with the concerned CVE's that they target. From the results gathered the authors note that it is common for malware developers to consult CVE databases to identify exploitation methods for use in development of IoT malware payloads. This has become more prevalent since 2016 and there appears to be a trend towards the use of less complex CVE's to leverage compromise with just 3 of the samples in analysis having a high complexity score for execution for the Common Vulnerability Scoring System (CVSS). This means more variants are utilising techniques that will allow low to zero touch interaction in order to be successfully deployed against a target. There are 2 versions of the CVSS system that are used, these are V2 introduced in 2007 and V3 introduced in 2015. Out of the samples analysed, 92.1% scored against the V2 system and 93.8% of samples scored against the V3 system show the use of CVE's that do not require user interaction to leverage. A common technique used by hackers is to look for low hanging fruit and to be able to leverage exploitation with the most ease and least noise or interaction with the most expansive coverage. The top Common Weakness Identification (CWE) listed as being compromised by malware include improper input validation, improper control of generation of code, improper neutralization of special elements used in an Operating System (OS) command, improper neutralization of special elements used in a command, improper restriction of operations within the bounds of a memory buffer and improper authentication. These account for 67% of cases showing that there is a significant issue with regard to insecure development and operational practices in the deployment of IoT solutions.

Another analysis of IoT malware threat is provided by Ibrahim Gulatas et al in [63], this provides the analysis of samples of 64 IoT malware families that have been identified between 2008 and 2022 looking at the variance of features including the architectures targeted, deployment methods, vectors of attack and methods of persistence. 80% of the malware samples offered botnet capabilities with the other 20% using sabotage, cryptomining or data exfiltration techniques, 19% of the samples contain multiple threat payloads or have multiple threat capabilities. 49 and 48 of the 64 samples variants target the Advanced RISC Machine (ARM) and Microprocessor without Interlocked Pipe Stages (MIPS) processor architectures with 41 targeting Intel based architecture. The research provides an informative summary of the 64 IoT malware variant families covered organised by the previously mentioned categories and also the types of consumer, commercial and industrial IoT devices that each malware is known to target. This is a valuable piece of research that allows reviewal and comparison of the collection of IoT malware variants covered and their adversary behaviours. Many of the example variants covered deploy botnet or cryptominer threat payload capabilities in compromise of a target, with other variants objectives being focused on espionage or data exfiltration activities against targets and also a few variants identified with bricker payloads. These identified variants are capable of targeting a wide range of types of IoT devices including Digital Video Recorder's (DVR's) and IP or webcams, Routers, Network Attached Storage (NAS) devices, toys and also SCADA control systems. Notably, the Hydra and ChuckNorris malware variants discovered around 2008 were some of the first identified examples of malware targeting IoT devices, these both used Internet Relay Chat (IRC) based, Command and Control (C&C) servers and dictionary attack methods against insecure services present on targets to leverage botnet capabilities including further propagation of threat payloads or deployment of DDoS attacks. Later examples including Gafgyt in 2014 and Mirai in 2016 were also capable of targeting IP cameras, NAS devices and DVR's to leverage botnet compromise.

One of the first IoT cryptominer malware identified around 2013 named Darlloz targeted set top boxes, toys and webcams devices and used a PHP Hypertext Processor (PHP) Common Gateway Interface (CGI) vulnerability to leverage RCE on the target device to mine cryptocurrency. Bricker malware variants, such as the Brickerbot malware identified around 2017, used various threat vectors to target smart bulbs, toys and webcams and are also noted as an emergent trend in more destructive malware. One other variant of note is the VPNFilter malware identified around 2018 that is capable of targeting SCADA control systems in order to leverage hijacking or Man in The Middle (MiTM) capabilities or bricking of the target device by overwriting critical operational areas of memory to leverage Permanent DoS (PDoS). Other interesting findings from the research include

proposal of the identification of four main malware families of Hydra, Tsunami, Gafgyt, and Mirai to be established as initial parent families in the evolution of development of IoT malware and as milestones with many of their behaviours or features being adopted by emerging variants.

A good point noted in the research is the emergence of cross-architecture malware that contain different payloads where required for targeting different Central Processing Unit (CPU) architectures, this presents a significant issue for detection of a signature of the malware as there will essentially be different signatures for the same variant through mutation of the signatures dependent of the CPU architecture of the target environment. This combined with the increasing variance in attack vectors along with the adoption of more advanced methods of obfuscation and communication present significant challenges for future intrusion detection for IoT systems and networks.

A timeline of historical Cyberattack incidents against ICS networks is provided by Kevin Hemsley and Ronald Fisher in [64], this work presents a timeline of significant Cyber incidents against Industrial Control Systems or Industrial IoT (IIoT) between 2010 and 2017, a number of these cases discussed in this following section involved significant or widespread compromise through the use of malware and also provide a good perspective of examples of relevant cases of malware threat against industrial IoT systems.

A. Stuxnet – 2010

One of the first suggested uses of state sponsored Cyber warfare against an industrial facility was reported in 2010, this was an operation that is proposed by many researchers and officials to be attributed to US and Israeli allied forces [68]. This deployed a Cyber-attack operation using the Stuxnet Malware against the Natanz Nuclear facility in Arak, Iran with the aim to disrupt or disable their nuclear weapons production capability, this was also known by the codename Olympic Games.

The attack exploited zero-day vulnerabilities in the Windows 7 Server Message Block (SMB) protocol and Siemens SCADA controller software technologies using a worm malware deployed from a removable Universal Serial Bus (USB) device. The malware established propagation of threat through the deployment of a rootkit and the use of fraudulently obtained digital certificates to allow further propagation of threat to be deployed from a remote C&C server. It also used other methods of evasion including injection of false data to provide legitimate looking readings from the Programmable Logic Controllers (PLC's) to the Human-Computer Interface (HCI) whilst performing covert corruption of the controllers of the facilities uranium hexafluoride centrifuges to instruct them to run a speed's that exceeded their operational capacity [64]. This resulted in reportedly a fifth of the centrifuges being damaged and failure of the facility and in turn the uranium enrichment program and marks one of the first recorded uses of state sponsored cyber threat to cause significant damage to critical infrastructure in a conflict setting [18].

B. Night Dragon – 2010

Also in 2010, a malware named Night Dragon was used to target oil and energy companies through phishing campaigns that deployed compromise of the Windows Active directory and operating system to allow for privilege escalation and propagation of threat through the injection of a Remote Access Trojan (RAT). This enabled further compromise and sensitive data exfiltration through Dutch and US based C&C servers [18]. Whilst this malware did not directly compromise the IoT or industrial control systems it could have allowed hijacking of the Human-Machine Interface (HMI) of the control systems through the remote desktop capabilities using the RAT component and highlights that there are various vectors of attack that can be used in order to gain compromise of such critical systems. A similar example of this type of vector of attack is the Duqu malware that appeared shortly after the Night Dragon variant in 2011 [18] [64].

C. Shamoon – 2012

Another variant similar to the Night Dragon and Duqu malwares was the Shamoon malware that targeted energy industry giants including RasGas and Saudi Aramco in 2012, along with intrusion and compromise to exfiltrate sensitive information the Shamoon variant caused destruction to the Master Boot Record (MBR) partition table and storage space by overwriting them with random data effectively disabling systems [68]. The malware also presented a graphic of a US flag on fire on the compromised system. In August 2012, Saudi Aramco were attacked with around 30,000 systems reported to be compromised. About a week and a half later RasGas were compromised by the malware. Further attacks were reported against Saudi Arabia's civil aviation agency in 2016 which reportedly resulted in the loss of data from thousands of systems [18] [64]. Again, this malware did not provide the direct compromise of ICS or IoT systems although it did provide the potential to be capable of this through an indirect vector of attack.

D. Havex – 2013

Another malware that has targeted ICS systems around 2013 is a RAT malware named Havex. This variant was capable of enumeration of systems, shares and ICS devices through exploitation of the Distributed Component Object Model (DCOM) based version of the Open Platform Communications (OPC) protocol. It was also capable of using a C&C server to propagate further exploitation. A threat actor tracked as EnergeticBear who are linked to Russian state intelligence services are proposed as being connected to the malware campaign. This shows another example of these types of attacks being proposed as being attributed to or deployed by state sponsored or linked actors and the capability to compromise critical infrastructure environments [64].

E. BlackEnergy – 2014

A malware variant named BlackEnergy was identified targeting the HMI's of ICS systems of the Ukrainian utilities provider Prykarpattyaoblenergo in 2015. The malware was proposed to be linked to a threat actor group affiliated with Russian intelligence services tracked as Sandworm and was suggested to have been active since 2011 and targeted multiple ICS vendors including Siemens and General Electric (GE) [64]. The malware carried out enumeration of shares and local or removable storage in order to attempt propagation of threat and DDoS attacks to disrupt services. A further variant of the malware identified as BlackEnergy3 with extended capability was reported to have compromised a further Ukrainian energy facility to cause an outage

resulting in mass blackout [17] [73]. This malware was also reported to have been successful in targeting rail and mining infrastructure in Ukraine [68]. The attack was reported to have lasted a few hours and may not seem long, but it should show that the potential damage that an attack of this type could cause could be catastrophic.

F. Industroyer – 2016

In December 2016 a second attack against another Ukrainian utilities provider occurred using a variant of malware tracked as Industroyer or CRASHOVERRIDE [64] [68]. The malware used a firmware rootkit, hijacking of Virtual Private Network (VPN) connections and modules that target ICS specific protocols including the International Electrotechnical Commission (IEC) 101, 104, 61850 and OPC protocols to leverage compromise and also provided support for targeting Distributed Network Protocol 3 (DNP3) [15]. The malware targeted circuit breakers of Remote Terminal Units (RTUs) in substations with the aim of causing sustained outage to Uninterruptible Power Supply (UPS) controllers by causing circuit breakers in 30 substations to trip reportedly affecting around 225,000 customers resulting in mass outage of services in Kiev. Similar to the BlackEnergy3 attack in 2015, technical help services were targeted with DoS at the same time as the malware deployment to hinder any assistance being acquired and to maximise the effect of the outage which was reported to have lasted a few hours [69].

G. NotPetya – 2017

In 2017, the NotPetya malware was identified as being responsible for one of the most economically destructive Cyber-based attacks in history across the Ukraine with reportedly around 10% of the computer systems in the country being compromised along with economic damage amounting to 0.5% of the Ukraine's Gross Domestic Product (GDP). Similar to the Petya malware the NotPetya variant used the EternaBlue and EternalRomance exploits to leverage access and carried out encryption of target systems however the NotPetya variant differed in that its aim was to corrupt systems by carrying out irreversible encryption of compromised targets MBR's in turn disabling the compromised system from booting [64]. The malware also targeted middle eastern Industrial control safety systems and many others with victims across airline, banking, government, healthcare and utilities sectors identified across 65 countries. Some of the high-profile targets compromised included FedEx, Maersk and Rosneft with damage estimations proposed of around $10 billion [68]. The NotPetya malware was again proposed to be linked to Russian state backed or affiliated threat actors the Sandworm group [48].

H. Triton/Trisis – 2017

Also in 2017, Cyber security analysis firms Symnatec and FireEye along with others reported the emergence of a malware variant known as Triton, Trisis and also HatMan. The threat framework targeted Middle eastern Industrial control safety systems with the aim of compromise of safety instrumented systems of the manufacturer Schneider Electric's Triconex [68] [49].

Triton was capable of modification of the in-memory firmware of the targeted device and injection of further malicious payload allowing access to memory contents, process corruption and RCE through malicious packet injection [70].

Triton is one of the first examples of a payload that targets specific types of industrial safety systems, this is particularly dangerous as it poses a significant threat to ICS systems and the safety systems defending them as well as the critical assets or human lives they are protecting [64]. The fact that more of these examples are being described as frameworks presents that the complexity and capabilities of these types of malware variants is significantly evolving.

Research in [65], by Shalia Sharmeen et al, looks at the security of mobile devices as part of an IoT network in particular the suspicious system and API calls and permissions that can be accessed by Android malware. The work notes various potentially suspicious system calls, permissions and API calls that's presence could be considered suspicious depending on the case. This is useful for developers or security practitioners unfamiliar with this.

Some of the suspicious system calls listed including UMASK, FCHOWN32, FSYNC, SYS224 and SYS248 could be of interest and potentially appear suspicious as it is common for malware to use uncommon or obscure routines or methods to attempt to carry out malicious operations in a stealthy manner. There are however various listed that are of clear commonly legitimate use with examples including MKDIR and CONNECT, RENAME and SOCKET. It also notes various potentially suspect API calls and permissions although many of these are genuinely used but could be of use to be aware of. Overall, the work does present a useful look at potentially suspicious system call presence in Android software.

A comparative analysis of five of the most high-profile malwares capable of targeting ICS systems is provided by Yassine Mekdad et al in [74] with a comparison of the Stuxnet, Havex, BlackEnergy2, CrashOverride, and TRISIS malware. The TRISIS malware is another example of ICS capable targeting malware that uses two CVE's (CVE-2018-7522, CVE-2018-8872) to target exploitation of Triconex safety systems [68]. The paper presents a two-layer approach of analysis of ICS malware with a cyber threat intelligence layer that maps indicators in the ICS kill chain of the example and a hybrid analysis layer of static and dynamic analysis. The solution uses Cuckoo sandbox to perform the automated Static and Dynamic analysis of the malware samples along with a threat analysis gathering stage to identify the "intrusion activities and their complexity" used in the example. This solution could prove to be valuable however the sandbox may present significant issues in versatile analysis of different samples aimed at varied or multiple architectures and may require development of a Sandbox environment specific to this.

Another useful approach for analysis of IoT malware is presented by Gaurav Pramod Kachare et al in [66], the research presents a solution that proposes to address the lack of presence of versatile sandbox environments for performing reverse engineering or analysis of IoT malware. There are some existing sandbox analysis solutions for IoT including v-sandbox, IoTPOT and Executable and Linkable Format (ELF) analyser although these solutions possess shortcomings in capabilities or some kind of constraint, such as being specifically scoped to ELF files, unable to perform a particular part of analysis or unable to address certain Anti-Virtual Machine (Anti-VM), Anti-debugging, evasion or

other advanced or unseen techniques presented by particular malware variants.

The challenge is to create the most legitimate and secure testing and emulation environment that allows a realistic execution environment to trick the malware into running and to allow the most legitimate and effective analysis potential. The research presents a conceptual design for a Sandbox solution for reverse engineering and emulation of IoT malware that allows static, dynamic and network analysis to extract features from the concerned malware sample. This research aims to provide solutions to address some of the previously noted shortcomings in existing solutions including providing the capability to be able to support emulation of multiple CPU architectures and performing advanced analysis of system and network data.

The proposed solution is capable of analysis of a malware sample using various static, network and real time or dynamic analysis techniques to extract features from the example, these features from the malware binary files are converted to 8-bit vector image files then to grayscale images generated from the grayscale vector values provided. Convolutional Neural Networks (CNN) are then used to attempt to provide accurate classification and generate automated reports.

It is arguable that the proposed solution is more effective than other known approaches in analysis, the research has promise in capability and for analysis of large batches of samples and it does appear to be more versatile compared to some existing solutions like v-sandbox, IoTPOT, IoTBOX and ELF analyser. Issues that could present challenges to the proposed solution include evasive or complex malware or zero-day threats and the ever-existing issue of false positive classification. The conceptual solution presented could provide a useful system that can attempt much of the heavy lifting in extraction of interesting or suspicious features from provided suspect malicious samples for further investigation.

A review of some of the recent approaches being explored in the detection of IoT malware is presented in the work by Sangeeta Kakati et al in [67]. The work notes that across IoT environments a significant percentage of devices use Android based operating systems including Electronic Chart Display and Information System (ECDIS) and Automatic Identification System (AIS) maritime systems, automotive solutions and also smartphones, watches, toys, smart television's as well as various other smart appliances.

The survey discusses various detection approaches discussed in recent research including traditional static, dynamic or hybrid analysis using feature extraction or analysis of suspicious Opcode, string patterns or API calls accompanied by machine learning classification techniques like Support Vector Machine (SVM), K-Nearest Neighbours (KNN) and fuzzy and decision tree methods with varying proposed classification accuracies of around 95 to 99%. It also discusses other machine learning assisted methods using Blockchain or CNN techniques. Blockchain technologies can be of use due to the distributed nature of ledgers which could present a useful resource that could assist in ensuring ease of access, portability and integrity of data or authentication in IoT settings. The study cites various methods for using CNN for classification. A method where the malware binary files are converted to 8-bit vector files then to grayscale images similar to the research in [66] is noted. These signature files were then processed with a CNN model with an accuracy of around 95 to 98% classification accuracy. While these methods are useful for batch or high-level analysis there would likely be issues in the analysis of cross platform malware or advanced variants using techniques like staged, polymorphic or metamorphic payloads.

Some of the examples discussed in the related works present useful areas in research and promising opportunities in the potential advancement in classification and detection of IoT malware. However, there remains problematic challenges of cross-platform variants, zero-day threats, false positive classification and the countering of obfuscation, anti-reversal and other deceptive or evasive measures employed by advanced threats.

Feature analysis or identification and classification of IoT malware studies also show that IoT malwares are becoming more complex advancing adversarial behaviours including smart and cross platform threats and poly and metamorphic payloads as the security of IoT systems and networks is beginning to be more considered.

III. CASE ANALYSIS

The following section will present an analysis of the BlackEnergy and Industroyer malwares and aims to provide a technical understanding of the two variants which are two of the most recent and significant malware threats to industrial IoT systems.

A. BlackEnergy

The BlackEnergy malware has evolved from the first variant in 2007 which primarily concentrated on the deployment of DDoS to further spamming and reconnaissance capabilities being provided in the second variant. The third generation has further evolved to support a modular architecture and significant Advanced Persistent Threat (APT) capabilities. Research by K. Stoddart in [68] notes the BlackEnergy2 threat as being "identified in energy-sector systems worldwide" around 2014 and was suggested to be primarily targeted at Ukrainian infrastructure with many intelligence and security researchers suggesting the threat being attributed or connected to the Sandworm group linked to Russian intelligence services.

The BlackEnergy2 variant appeared around 2010 and uses process injection and rootkit techniques to deploy its payload and provides an evolved modular structure with additional enumeration and reconnaissance abilities. This means the malware has expanded capabilities that can be deployed as required by target making the payload more efficient and also more complex to perform analysis for reverse engineers. It was used in an attack against Ukrainian power grid infrastructure in 2014 and was capable of targeting various vendor specific HMI products including Siemens SIMATIC, Advantech/Broadwin WebAccess and GE CIMPLICITY.

BlackEnergy2 is distributed through a malicious email attachment which deploys a Trojan containing 8 imported Dynamic Link Libraries (DLL's) and 164 functions and uses further propagation and privilege escalation to leverage authenticated connection to SCADA HMI's to allow injection of commands to open circuit breakers and cause outages.

The BlackEnergy3 variant employs a method of exploitation that targets abuse of the parsing of INF configuration files through a known CVE (CVE-2014-4114) in the Object Linking and Embedding (OLE) packager held in the Windows packager.dll that allows sharing of media between office and other applications. The vulnerability

allows possible injection of malicious macros to Microsoft Office documents with a potential to leverage RCE against the target. Microsoft has since patched this vulnerability however it is still a relative threat as it is still possible to engineer a target into enabling the macros in a file. BlackEnergy3 was listed as being present in a Ukrainian energy providers infrastructure that was victim to an attack in 2015 causing an outage in a report by the US department of Homeland security. It was used to perform 3 simultaneous attacks against substations of DTEK Kyiv Region Grids (formerly PJSC Kyivoblenergo) which opened breakers, corrupted hardware and performed DoS against response services. The attacks resulted in seven 110 kV and twenty-three 35 kV substations being disconnected for around three hours and caused outages for approximately 225,000 customers across 3 different providers.

An analysis of one of these attacks is provided by the Electricity information sharing and analysis center (ESIC) of SANS in [73] which suggests that a collection of factors contributed to the ease or capability of intrusion or compromise including a lack of 2-factor authentication of VPN connections and a lack of capability in network monitoring facilities, such as IDS or Intrusion Prevention Systems (IPS). The attacks were deployed through spear phishing campaigns to gain access to the IT or business networks of the providers, this was achieved through a payload embedded in a malicious Microsoft Excel or Word office document. On acceptance of a request to enable macros an embedded malicious macro in the document is then executed which will initiate the process of deploying the BlackEnergy payload. The malware then pivoted through the network with harvested credentials, abuse of VPN accounts and use of native tooling or living-off-the-land techniques to access the ICS segments of the network. This then allowed injection of malicious instructions to HMI's in the SCADA network segment to open breakers in substations while also deploying a KillDisk payload to delete the MBR of compromised devices and leveraging corruption of network-connected UPS equipment by injection of malicious bricking firmware to serial to ethernet controller equipment which in turn would disable the equipment raising power outage. An attempt at preventing direct remote response by deploying telephony DoS attacks against call center response services was also carried out. It is suggested that the attackers were knowledgeable of RTU equipment or able to enumerate infrastructure through precursor attacks or persistence to be able to target firmware of the serial to ethernet hardware and also achieve successful interaction with the variance of Distributed Management Systems (DMS) in the targets attacked.

An analysis of a sample of the BlackEnergy3 source is provided by Udi Shamir on behalf of SentinelOne in [73]. A malware author will usually attempt to obfuscate as much of the malicious source as possible however analysis of a sample of the BlackEnergy variant has shown the presence of entries left in the FONTCACHE.DAT file that provides the address location of the sources Program Database (PDB). This allows the program to be analysed with static analysis tools or connected to a debugger to investigate evidence including the addresses of functions, inspection of variables, parameters or pointers or general inspection of source. Further inspection of the malicious samples OLE structure shows an attached macro present, the extracted macro contains the BlackEnergy payload in chunks as arrays, these segments are reassembled then the payload is executed. The beginning of the arrays shows the values 77 and 90 contained, these are 4d 5A in hex which are the magic number identifier for DOS executable. The reassembled source contains two Portable Executables (PE) named rundll32.exe and FONTCACHE.DAT. This is the mechanism the threat will use to deploy.

The payload will run from the macro to deploy an executable which will deploy the rundll32.exe if not present. The rundll32.exe is a native Windows tool for running DLL's. This will then be used to run the desired DLL, in this case the FONTCACHE.DAT file. This file contains a disguised network sniffing utility which is run using rundll from a startup menu lnk shortcut and registry entry with the ShellExecute Windows API to deploy a further malicious payload. This allows the payload to be run on boot.

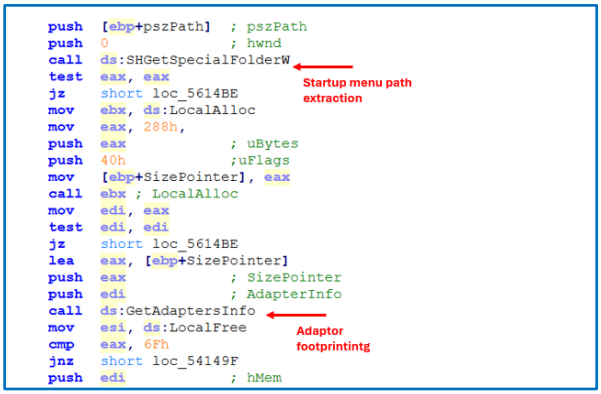


Figure 1: Initial enumeration by source

The sample carries out various actions to attempt evasion and possibly hinder debugging in performing obfuscation through the use of a CryptDecrypt() function and also performing a check using a call to the SetUnhandled ExceptionFilter API to check for the presence of a debugger. The source checks the available network adapters on the target system using an API call to GetAdapterinfo. This call can be seen in Figure 1. This is then provided as a parameter to the previously mentioned startup menu lnk entry to rundll to define the network adapter to use.

An entry is also injected to target system registry at Software\Microsoft\Windows\CurrentVersion\Explorer\Shel l to allow execution on startup. When the rundll call is made on execution of the lnk shortcut entry a process is spawned to kill the original process running the malicious macro and delete the original dropped malicious office file. The attack routine then makes multiple attempts to run the disguised WinPcap service from the FONTCACHE.DAT file using a call to OpenSCManagerA, OpenServiceA, StartServiceA, LoadLibrary or with W appended to the function for wide and also attempts evasion through the use of crypter() and sleep() functions.

This allows the payload to be capable of extensive network sniffing and subversion abilities. Analysis of the source provided in [73] shows the corresponding registry entry of OpenSCManagerW being modified. If successful, this will allow further reconnaissance or control through C&C services. Figure 2 below shows the macros execution, this will be run from the startup registry entry setup in the previous step, once this has executed the macro will deploy the BlackEnergy payload to complete the intrusion kill chain.

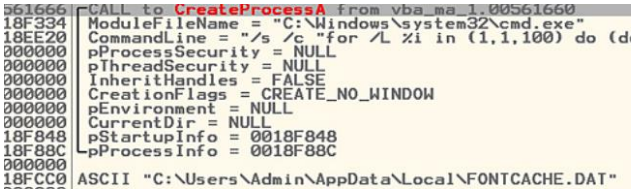


Figure 2: Macro execution

The BlackEnergy payload then will attempt traversal of the network aiming to access the ICS segments to inject malicious commands to the HMI's and open substation breakers while also injecting the KillDisk payload and malicious firmware to brick target devices and their serial to ethernet controllers. This then would cause an outage.

Research in [73] suggests there are similarities between the BlackEnergy payload and other well-known variants like the Sality and Operation Potao Express malware variants [50] identified as being used in attacks around the same time in Estonia and Georgia, this could imply that there is a potential that the same actors may be responsible or that the authors of the Sality and Potao Express variants have modified the payload or borrowed features from it.

B. *Industroyer & Industroyer2*

Industroyer

Another recent example of malware capable of targeting ICS is the Industroyer malware also known as CrashOverride. Research by Dragos in [69] and ESET verify that this variant was responsible for further attacks against Ukrainian power grid infrastructure in December 2016 in Kiev. This has since been attributed to a threat actor group tracked as ELECTRUM who are known to be affiliated with the Sandworm group who are linked to Russian intelligence services. [51].

Dragos researchers note that the Industroyer malware is one of the first malware variants capable of targeting ICS infrastructure along with the Stuxnet, Havex and BlackEnergy variants and also one of the first specifically developed to target electric utilities providers infrastructure.

Industroyer, like BlackEnergy, uses a modular architecture and contains sabotage and wiper modules and modules capable of targeting ICS protocols including IEC-101, IEC-104, IEC-61850 and OPC as well as support for DNP3 meaning the variant has the potential to be highly configurable and adaptable. Its method of compromise abuses DLL's of the concerned protocol for example 101.dll, 104.dll or OPC.dll. Unlike BlackEnergy it is capable of direct interaction with ICS hardware rather than injection of instructions through an HMI.

The attack routine by the malware bears significant similarities to the routine used in the BlackEnergy malware attacks by deployment through a Phishing campaign and opening breakers in substations to cause a significant outage and sabotage of serial to ethernet controllers with malicious firmware, this was also accompanied by a DoS attack against telephony assistance services to hinder response or resolution and sustain the compromise.

The Industroyer source contains 7 imported DLL's and 46 functions and includes launcher and wiper payloads. It also contains a backdoor payload and uses harvested VPN account details to leverage access to the ICS segment of a targeted network through the ICS segments historian database. It then deploys a launcher for propagation of threat through the various ICS protocol targeting modules and creates a malicious service to target corruption of HMI configurations. An overview of the behaviour of the Industroyer malware is provided in [70] by Marcus Geiger et al, their analysis explores the variant's use of native tooling like Powershell to leverage Golden Ticket attacks against the Kerberos service in Windows using Mimikatz or dumping of credentials from the systems memory. These are common exploit routines or techniques used for privilege escalation in Windows operating systems to provide elevated access. This assists with propagation of threat and pivoting to the ICS segment to deploy the Industroyer payload as a service to maximise persistence.

The Industroyer malware is capable of the enumeration of available protocols and configurations used in a target network and the selection of appropriate payloads for use in successful interaction with the target that are then deployed by the launcher payload.

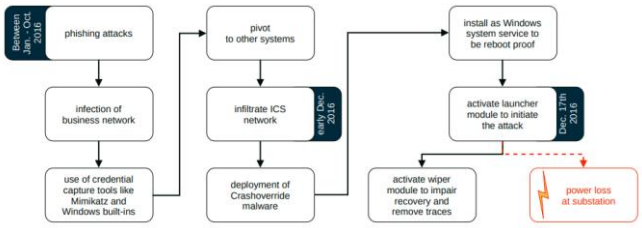


Figure 3: Industroyer compromise routine

As well as this, wiper malware and backdoor payloads are deployed to corrupt entries in the registries of Windows machines with the aim of corrupting the systems from booting successfully and also to attempt to leverage and maintain persistence. An example of the Industroyer malware high level kill chain routine can be seen in Figure 3 above:

Industroyer2

The Industroyer malware has since been evolved by the developers to directly target IEC-104 ICS network protocol rather than targeting multiple.

An overview of attacks involving the variant known as Industroyer2 is provided by K.Stoddart in [68]. The Industroyer2 malware was used in attacks in February, March and April of 2022 carried out against Ukrainain electrical utility infrastructure, this time the attacks were accompanied by the use of a wiper malware named CaddyWiper to attempt to leverage further compromise. These events were inline with the Russian invasion of Ukraine.

The Industroyer2 variant differs from the original Industroyer payload in that it specifically targets the IEC-104 rather than being able to target various ICS protocols with an adaptable modular framework like the first variant [52]. The sabotage modules in the Industroyer2 payload are used to instruct RTU's to open and keep open circuit breakers to leverage a sustained outage. The result of one of the attacks was the temporary disablement of 9 substations supplying around 2 million customers in turn creating significant outage [53].

The source code of Industroyer2 contains a collection of hardcoded configurations of Information Object Address

(IOA) listings which are used for targeting differently configured endpoints in an ICS segment [54]. These IOA listings hold details of the addresses of the Application Service Data Unit (ASDU) or Information Objects and are passed as strings to the module interacting with the IEC-104 protocol [52], this differs from the original strain that uses an Initialisation (.INI) file to store these configurations [55].

Another analysis of two samples of the Industroyer2 variant by Nozomi networks in [71] provides a good discussion of its low-level behaviour. As mentioned previously, the second variant of the Industroyer malware specifically targets the IEC-104 protocol and contains a hardcoded list of unobfuscated string IOA's which are configuration details for different end stations. The hardcoded listings contain entries for 3 separate station configurations providing the ability to target these ranges of station configurations and hold definition for a Station Configuration Header and an IOA Configuration Format.

(1) Station Configuration Header

An example of one of the Station Configuration Headers can be seen below, this contains local IP address and port followed by the ASDU address (in this case 3) then the mode of configuration which can be 0 for use of hardcoded listings or 1 for allowing the specification of two values to be used as a range of hardcoded IOA's to be attempted.

In the following example, we can see a specified address followed by a port and then an ASDU address of 3 and the operation mode set to 0. The header also contains the name of the process to be killed of PService_PPD.exe, the path location where the executable targeted for hijacking is stored of D:\OIK\DevCounter and also other configurable options, such as sleep counter, filename path location and the header number of the following IOA object, this is the end value in the example below of 44.

10.x.y.z 2404 3 0 1 1 PService_PPD.exe 1 "D:\OIK\DevCounter" 0 1 0 0 1 0 0 44

This template of instruction defines the format and configurable options for commands that can be sent to the particular type of station device to be executed. These Station Configuration Headers will be selected as required to target the appropriate station device.

(2) IOA Configuration Format

The IOA Configuration Format holds details of a list of IOA's and there concerned parameters including setting of single or double commands, index number and priority.

The malware begins infiltration by disabling and renaming the PServiceControl.exe and PServicePPD.exe services on the target, this stops the service being restarted. Capture of simulated network behaviour of the malware using Wireshark allowed for analysis of IEC104 protocol and port 2404 specific traffic which is a commonly used port for IEC protocol traffic. Inspection of packets show they contain the values for the IOA being accessed.

The payload then attempts interaction with target substations by initiating a network connectivity test using ***TESTFR act*** packets which are acknowledged by the recipient with ***TESTFR con*** packets. Once this transaction has taken place an attempt to start a data transaction is initiated using ***STARTDT act*** packets which are answered with a ***STARTDT con*** reply if successful, this is followed by an interrogation command C_IC_NA_1. This allows configuration of the type of frames that will be used in communication which can be C_SC_NA_1 for allowance of single command or C_DC_NA_1 double command depending on the endpoints configuration. This transaction can be seen in Figure 4. The malware uses its hardcoded configuration to iterate available IOA's of the target to identify and send the appropriate type of frames. This allows further interaction with the endpoints in the ICS segment of the network and carrying out injection of commands with disruptive intent. The Industroyer2 variant is more complex in nature compared to the first variant and BlackEnergy in that it is capable direct manipulation of ICS devices rather than remote execution and it is also highly configurable or adaptable.

Figure 4: Initial transaction with ICS segment

There are noted to be heavy similarities in source code in the two samples analysed of a sample of Industroyer from 2016 and a sample of Industroyer2 from 2022 [52]. An example of this can be seen in Figure 5. The samples are different in that the execution of the disabling of processes in the main thread of the first Industroyer variant have been transferred into a thread that starts the main thread [71]. However, various other similarities in the code or routines exist in the variants.

Similarities include the use of the same elements and routine to store global data showing an incorporation of this template design. Both samples are also noted as passing this data using the same data structure and similar methods of parsing to various functions and also have a lack of presence of obfuscation.

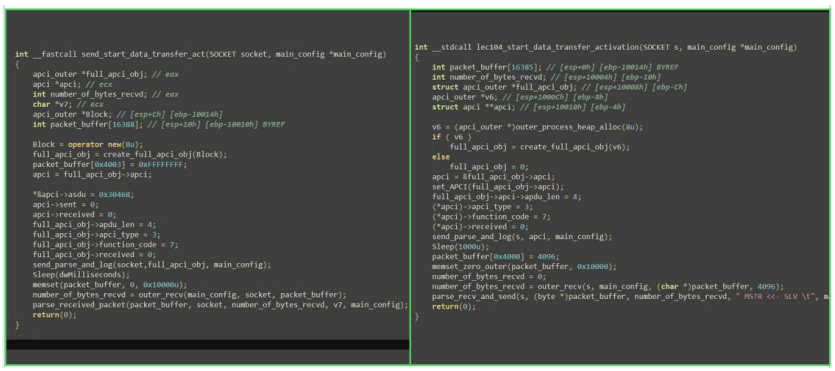


Figure 5: Similarities between Industroyer and Industroyer 2 samples

A report by ESET stated that the attacks targeted against a Ukrainian provider in April 2022 were unsuccessful but aimed to leverage a large-scale outage and deploy the CaddyWiper and other wiper malware to cause further disruption and hinder forensic analysis. It is suggested in some reports that there is a potential that Industroyer or similar types of malware could cause significant and extended outages if there were used in synchronised attacks against various targets. Another factor of note of this analysis is that the authors are highly likely to have been familiar with (or gained familiarity possibly through precursor attacks or insider knowledge of) the Operational Technology environment to achieve the successful design and deployment of the threat operation. This would tie in with reports of the suggested attribution to the Sandworm threat actor group who would likely have substantial knowledge and familiarity with the hardware and software used in the utilities infrastructure of the target [56] [68].

The BlackEnergy and Industroyer malwares mark a new era of threat towards ICS or IIoT systems in the targeting of utilities infrastructure and provide examples of the general advancement of methods and kill chains of attack used to target these types of systems. These cases demonstrate the breadth and variance of potential targets capable and also highlight the potential criticality of severity that a successful attack against certain IoT solutions can imply.

IV. DISCUSSION

Malware threats facing the IoT have shown significant evolvement in the past decade. Many of the early malware threats that emerged leveraged DoS and botnet capabilities, this has evolved into ransomware, bricker, cryptominer and info stealer variants being observed emerging as more recent threat trends.

Two of the first threats of significance discovered, named Hydra and ChuckNorris were first identified around 2008 and 2009 respectively, these were capable of botnet and DoS intents and targeted MIPSel and 32 and 64-bit Intel (x86/x64) architecture modems and routers with the Hydra variant originally targeting D-Link branded routers. The Hydra variant exploits insecure or default credentials in the D-link router authentication mechanism using dictionary cracking attacks or an authentication bypass technique over Secure Shell (SSH) or Hypertext Transfer Protocol (HTTP) with further threat propagated through an IRC based C&C agent for example the downloading of further threat payload, malicious instruction execution or the launching of TCP or User Datagram Protocol (UDP) based DoS flood attacks.
The ChuckNorris variant leverages compromise through the deployment of a Telnet dictionary attack and propagates further threat through a for purpose SSH client downloaded to the target through IRC based botnet control. This can then be used to hijack the device or execute botnet operations including Domain Name System (DNS) spoofing and the deployment of UDP DoS flood attacks.

One of the first cryptominer IoT malware was discovered in 2013 named Darlloz, this targeted tv set top boxes, toys and webcams using ARM, MIPS, MIPSel and Power PC (PPC) architectures and leveraged use of a known CVE tracked as CVE-2012-1823 which is a PHP CGI vulnerability that can allow RCE on the target. This was one of the first identified examples of a malware threat that leveraged the mining of cryptocurrencies and made use of a known CVE in its exploitation of a target IoT system.

Another example of a different type of variant was observed between 2013 and 2016 and involved the use of a trojanised Android application named X-Agent, a good analysis of this is provided in **[72]** by Crowdstrike. This was a trojanised Android application that was disguised as software for the remote configuration and control of howitzer equipment used by Ukrainian forces and hosted on a legitimate or trusted Ukrainian forum. This was used to provide targeting and espionage capabilities against Ukrainian forces and also enumerated and exfiltrated locational and communications data from the compromised devices which was then used to provide information to assist pro-Russian forces in leveraging offensive operations against the targets. The campaign was reported to have resulted in the loss of around 80% of the Howitzer Artillery equipment it was used to target within a period of 1 year. This was attributed by various researchers to the FancyBear threat actor group who are also tracked by the label APT28 and are linked to the Russian GRU intelligence service [18]. While this example seems extreme it highlights the varying potential severity that attacks against an IoT system can result in and also the heterogeneous characteristics and sensitivity of environments that an IoT system can involve. Further examples of botnet capable malware have been identified targeting a number of devices including IP cameras, NAS devices, DVR's and routers. These include the Gafgyt malware in 2014 and the Mirai malware in 2016, both of these used dictionary cracking methods against insecure services present including SSH, Telnet, Microsoft Structured Query Language (SQL) and MYSQL. These were both used to leverage significantly large-scale botnets and DDoS attacks. The Mirai variant was particularly destructive in its intent as once a target is compromised the Mirai malware will instruct the compromised device to refuse any further HTTP, SSH and Telnet connections essentially disabling the device and to hinder any attempt to recover it.

An additional emergent trend identified around 2017 was that of Bricker malware that will sabotage or brick a target device to disable it. One of the first examples of this type of variant was the Brickerbot variant in 2017 that targeted smart bulbs, toys and webcams through known vulnerable vectors in the Busybox OS which is a widely used embedded Linux framework in mobile and IoT or smart devices. It also used Dictionary attacks against HTTP, Simple Object Access Protocol (SOAP), SSH, Telnet and Home Network Administration Protocol (HNAP) protocols and was also capable of targeting a wide range or vendor specific devices with target specific threat payload. Another example of this type of variant is the Silex malware discovered around 2019 which was also capable of targeting a wide range of architectures and devices and used dictionary attack techniques against insecure services present to leverage compromise. Both variants will then overwrite critical areas of memory of the device to attempt to leverage bricking or PDoS against the device or service it is providing.

There is an increase in malware identified that is capable of targeting SCADA control systems since around the time of the Stuxnet incident in 2010. This includes the discovery of the BlackEnergy, Industroyer, LemonDuck, and VPNfilter variants along with others. The VPNFilter variant was discovered around 2018 and targets ARM, MIPS and x86

architecture SCADA control systems and NAS and router devices.

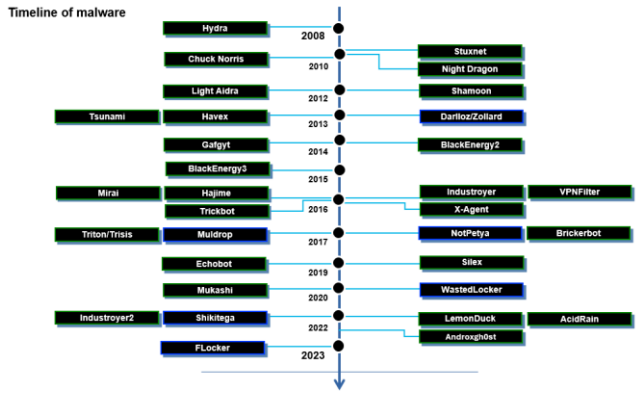


Figure 6: Timeline of significant IoT malware

The variant uses dictionary attack techniques to attempt to leverage hijack or MiTM compromise of the target to further propagate compromise through data exfiltration, DNS poisoning, DDoS or PDoS through the overwriting of critical OS components to render the system inoperable.

The following image in Figure 6 shows a timeline diagram of the 32 of the most significant malware variants targeting IoT solutions over the past fifteen years by the year they were first observed.

These examples show that there is a clear evolution in adversary behaviours and more extensive malware threat campaigns towards IoT systems in recent years. The BlackEnergy and Industroyer variants are examples of malware that are capable of significant and in cases potentially critical threat to the operation and safety of ICS systems. The capabilities of deploying wiper and bricking payloads to corrupt firmware, deletion of critical system data and disruption of operations is a considerable threat to these types of networks, systems, data and operational security. Both cases show that there are often multiple vectors of potential vulnerability or attack in the attempt to compromise IoT or ICS systems which increases the challenges in creating secure while efficient environments.

The BlackEnergy3 and Industroyer2 variants both show evolution in their methods from predecessor counterparts with the BlackEnergy variant originally being used for DoS attacks to using a Trojan and macro as methods of exploitation and further interaction with the target for propagation of compromise. The Industroyer variant transformed from a multiple protocol targeting modular framework to a protocol specific configurable and adaptable threat payload. They are also capable of enumeration of the target, advanced process injection, C&C interaction, espionage and sabotage.

This work along with the studies in various of the referenced research show that there has been an advancement of adversarial routines by malware authors in use of obfuscation, communication techniques and others methods of evasion in malware targeting IoT systems. The adversarial routines of attackers are likely evolving in parallel with improving security practices. There has been an emergence of cross platform or architecture malware along with more target specific or what could be considered a level of smart payloads. Early examples, such as Brickerbot sabotage devices through the compromise of products of insecure development or practice for example insecure default configurations or the Ramnit Trojan that creates a File Transfer Protocol (FTP) network from targets and propagates by malicious links. Others like the Trickbot malware directly exploit CVE's in MikroTik router devices to propagate further botnet threat [43]. More evolved recent examples emerging include the LemonDuck [44] variant that targets industrial & manufacturing systems, this contains payloads to target exploitation through SQL injection or abuse of SMB or other insecure technologies present.

The BlackEnergy and Industroyer malware are examples of some of the most significant malware threats to industrial IoT systems as they are payloads capable of disruption or sabotage of critical facing systems and are a concerning example of what future potential threats could be.

Traditional ICS systems are built with the view of convergence of IT and OT, this with the introduction of IoT creates an architecture that is created to ensure compatibility but may overlook or omit certain aspects of security. As always, an attacker will generally exploit low hanging fruit to gain compromise, there is however a much larger pool of potential threat vectors concerning IoT security.

A shortlist summary of the number of significant variants and types of malwares found to be targeting IoT can be seen in Table 1 on the following page. This shows the malware names, types, year or emergence, architectures and types of devices that they target and the vector of attack that they use in compromise of a target.

The adversarial routines of malicious actors are evolving along with IoT technologies developing and becoming more secure, threat operations are also possibly maturing through experience. There has been a noted increase in the use of CVE's [62] by malware threats to target compromise, the Gafgyt and VPNFilter variants are examples of this and also as mentioned previously the Echobot malware has a range of 71 CVE's that it can use to target vulnerable systems [58] [59]. Malware authors can make use of advanced techniques including fileless threat deployment, covert process injection or hollowing, code caves, zero days, cross platform threats, smart payloads, custom routines, advanced obfuscation or evasion techniques, AI or various others. The advancement of adversary behaviours in subversion and compromise against IoT targets will present further challenges to IoT security, for example when attackers begin to more prevalently utilise advanced techniques including polymorphic, metamorphic, staged or fileless payloads or use of more evasive segmentation or stronger obfuscation and encryption in these payloads there will likely require a parallel advancement in security and analysis practices. This means effective security strategies and practices are fundamental in order to protect IoT technologies from threat of IoT malware.

Some useful directions of research towards the mitigation of IoT malware threat include the design and development of lightweight device side monitoring or altering tools, the use of network-based hardening techniques including periodic authentication or use of forensics artefacts, such as beacons, canary tokens or honey tokens or the development of an analysis methodology and proactive solutions for the effective analysis of mobile and IoT malware.

TABLE I: MALWARE SUMMARY TABLE

Malware	Year	Type	Target	Architecture	Vector
Hydra	2008	Botnet, DDoS	Routers	MIPSel, x86	CVE-2018-7043
ChuckNorris	2009	DDoS, Hijacker	Modems, Routers	MIPSel	Telnet, Dictionary attack
Darlloz	2013	Cryptominer (Dogecoin)	Webcams, Toys, Set top boxes	ARM, MIPS, MIPSel, PPC	Dictionary attack, CVE-2012-1823, instruction injection
Gafgyt	2014	Botnet, DDoS	Routers, IP cameras, DVR's	ARM, MIPS, x86, PPC, Sparc…	RCE, CVE-2014-8361, CVE-2017-17217, CVE-2017-18368, CVE-2018-15887
BlackEnergy3	2015	ICS, Sabotage, DDoS	Siemens, GE HMI equipment	x86/64	Phishing, malicious macro injection, CVE-2014-4114
Trickbot	2016	Botnet, DDoS	Mikrotik routers	ARM	Brute force attack, CVE-2018-14847
Mirai	2016	Botnet, DDoS	Routers, IP cameras, DVR's, NAS, Printers	ARM, MIPS, x64	Dictionary attack
Industroyer	2016	ICS, Sabotage, PDoS	RTU's, UPS	x86, x64	Phishing, direct tampering of ICS equipment using (IEC) 101, 104, 61850 and OPC
Muldrop	2017	Cryptominer	Raspberry Pi	ARM	Default credentials
Brickerbot	2017	Sabotage, DDoS	Webcams, Toys, Smart bulbs	Busybox OS	Dictionary attack
VPNFilter	2018	MiTM, Sabotage	SCADA, NAS, Routers	ARM, MIPS, x86	Dictionary attack, 14 known CVE's
LiquorBot	2019	Cryptominer (Monero)	Routers	ARM, MIPS, x86, x64	Dictionary attack, 12 known CVE's
Silex	2019	Sabotage, DDoS	Any	ARM, MIPS, x86, SH4, Sparc	Dictionary attack
Echobot	2019	Botnet, DDoS, Propagation	Modems, Routers, ECDIS, PLC/RTU's/ NAS, Smart TV, ICS	ARM, MIPS, MIPSel, x86, x64, SH4, Sparc, PPC	71 Known CVE's
WastedLocker	2019	Ransomware	Garmin devices	ARM	Phishing, backdoor injection
Industroyer2	2019	ICS, Sabotage, PDoS	RTU's, UPS	x86, x64	Phishing, direct tampering of ICS equipment using IEC 104, Wiper payload deployment
Mukashi	2020	Botnet, DDoS	NAS (Zyxel)	ARM, MIPS, MIPSel, x86, SH4	Dictionary attack, Pre-authentication command injection, CVE-2020-9054
AcidRain	2022	Botnet, DDoS, Sabotage	KA-SAT Modems, Routers	MIPS	Firmware exploitation, Wiper payload deployment
Androxgh0st	2022	DoS, Hijacking, Info Stealing	Many	ARM, MIPS, x86/64	CVE-2017-9841**,** CVE-2018-15133, CVE-2021-41773
Shikitega	2022	Cryptominer (Monero)	Any	x86, x64	CVE-2021-3493, CVE-2021-4034
Flocker	2023	Ransomware	Smartphones, TV's Tablets, Android IoT devices	ARM, x86, x64	Phishing, Social engineering, trojanised apk

The use of effective network segmentation particularly in appropriate segmentation of IT and OT infrastructure endpoints or components or a zero-trust network strategy is a useful approach but this may not always suit operational requirements. A useful approach in ensuring the most secure practice is the development of zero trust configurations, these would most likely be setting or device specific and could consist of whitelisted definitions of allowed operations and blacklisted definitions of suspected or known malicious actions.

The VPNFilter [79] and AcidRain [80] malwares are more recent examples of this type of variant. Both variants are of particular note. The AcidRain malware was successfully targeted at Satellite router and terminal equipment to cause widespread outages. The VPNFilter malware variant utilises 14 CVE's it can use to target devices. The Echobot malware [57] is one of the more recent good examples of malware incorporating multiple CVE's to target compromise making use of 71 known CVE's.

Another rising trend is in ICS targeting malware including BlackEnergy 3 [17], Industroyer [51], Industroyer 2 [51] and LemonDuck [44] as recent examples. These have shown significant capabilities in disruption with the BlackEnergy 3 and Industroyer 2 variants being used to raise significant outages.

The Androxgh0st malware identified in 2022 was one of the most active variants in 2024 and is a final good example of evolving threat behaviour, this example uses info stealing and hijacking techniques to leverage compromise of Android based devices.

A clear shift in malware threat trend can be observed from the cases reviewed in this paper. There are already examples of malware that contain extensive threat payloads, advanced adversary behaviours and cross-platform targeting capabilities. The advancement of behaviours, increase in

cross-platform threats, ease of deployment through emerging resources like Malware-as-a-Service (MaaS) where malware threat campaigns are deployed from throw away infrastructure, the challenge of timely attribution and also the issue of zero-day threats of which there could arguably be higher potential for presence in an IoT setting mean that malware will remain a significant challenge to the security of IoT systems and networks.

There is requirement for effective resources or solutions that can provide the extensive analysis capabilities required for the analysis of the variance of IoT malware. Not only is there a lack of capable environment to achieve all of the emulation, virtualization or analysis capabilities required, there is also a lack of a standard analysis methodology for application. These are potential areas of valuable future research.

V. CONCLUSIONS AND FUTURE WORK

This paper aims to have provided insight in the review and analysis of IoT malware threats and outlined the variance of threats that exist. The review of related literature looking at these areas and the centralisation of certain relevant research in the investigation and analysis of some of the example variants covered is another value that could provide useful assistance to other researchers.

This paper should assist in providing an understanding of the range of routines or techniques that malware developers can employ to target and leverage compromise or exploitation of IoT solutions. The analysis and comparison of the BlackEnergy and Industroyer variants of malware could also prove to be useful assistance to other researchers or as a basis of research in analysis of or comparison to future variants that may emerge.

Future direction of additional research in this area could include the analysis and comparison of collections of malware variants targeting automotive, healthcare or types of IoT solutions to identify their capabilities, commonalities and unique elements. Also, further analysis of the Industroyer2 malware variant in regard to the low-level analysis of its modular components could provide further understanding of the malware's exploitation routine

It is likely that there will continue to be an increase in the development and adoption of IoT and smart technologies. As new technologies, systems and networks are introduced, IoT malware threat trends are likely to shift in parallel to target exploitation of these. The development of proactive analysis solutions and a standard analysis methodology for the analysis of IoT malware are two areas that would be of useful future research direction and if developed could provide a valuable resource.

REFERENCES

[1] M. Weiser, "The computer for the 21st century," Available at: https://www.cs.cmu.edu/~jasonh/courses/ubicomp-sp2007/papers/02-weiser-computer-21st-century.pdf [retrieved: August, 2025].

[2] Cisco, "Cisco annual internet report (2018–2023) white paper," Available at: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html [retrieved: June, 2025].

[3] The Government Office for Science, "The internet of things: Making the most of the second digital revolution," Available at: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/409774/14-1230-internet-of-things-review.pdf [retrieved: June, 2025].

[4] [Unit 42, "2020 Unit 42 IoT threat report," Available at: https://unit42.paloaltonetworks.com/iot-threat-report-2020/ [retrieved: June, 2025].

[5] A. Gowdiak, "Security vulnerabilities in Telit Cinterion IoT (formerly Thales) devices," Available at: https://seclists.org/fulldisclosure/2023/Apr/11 [retrieved: June, 2025].

[6] Mitre, "OpenSSH CVE list," Available at (cve.mitre.org) [retrieved: June, 2025].

[7] Tenable, "ArubaOS WPA2 key reinstallation vulnerabilities (KRACK)," Available at: https://www.tenable.com/plugins/nessus/103855 [retrieved: June, 2025].

[8] D. Antonioli, "Bluetooth BIAS attacks," Available at: https://francozappa.github.io/project/bias/ [retrieved: June, 2025].

[9] Carnegie Mellon University, "CERT/CC vulnerability note VU#918987: Bluetooth BR/EDR supported devices are vulnerable to key negotiation attacks," Available at: https://www.kb.cert.org/vuls/id/918987/ [retrieved: June, 2025].

[10] Montsecure, "Call me maybe: Eavesdropping encrypted LTE calls with ReVoLTE," Available at: https://montsecure.com/research/revolte-attack/ [retrieved: July, 2025].

[11] Radware, "BrickerBot results in permanent denial-of-service," Available at: https://www.radware.com/getattachment/Security/Threat-Advisories-and-Attack-Reports/1418/ERT-Alert-BrickerBot-PDoS-2.pdf.aspx/?lang=en-US [retrieved: June, 2025].

[12] Ilascu, "New Silex malware trashes IoT devices using default passwords," Available at: https://www.bleepingcomputer.com/news/security/new-silex-malware-trashes-iot-devices-using-default-passwords/ [retrieved: June, 2025].

[13] B. Krebs, "Source Code for IoT Botnet 'Mirai' Released," Available at: https://krebsonsecurity.com/2017/12/mirai-iot-botnet-co-authors-plead-guilty/ [retrieved: June, 2025].

[14] VX-Underground, "Hajime worm battles Mirai for control of the Internet of Things," Available at: https://www.vx-underground.org/malware_defense.html [retrieved: June, 2025].

[15] C. Huey, A. Windsor and E. Brumagin, "Lemon Duck spreads its wings: Actors target Microsoft Exchange servers," Available at: https://blog.talosintelligence.com/lemon-duck-spreads-wings [retrieved: June, 2025].

[16] Cherepanov, ESET, "Win32/Industroyer: A new threat for industrial control systems," Available at: https://web-assets.esetstatic.com/wls/2017/06/Win32_Industroyer.pdf [retrieved: June, 2025].

[17] Securelist, Kapersky, "BlackEnergy APT attacks in Ukraine employ spearphishing with Word documents," Available at: https://securelist.com/blackenergy-apt-attacks-in-ukraine-employ-spearphishing-with-word-documents/73440/ [retrieved: June, 2025].

[18] M.B. Gazula, MIT CAMS, "Cyber warfare conflict analysis and case studies," Available at: https://cams.mit.edu/wp-content/uploads/2017-10.pdf [retrieved: June, 2025].

[19] B. Krebs, "Mirai IoT botnet co-authors plead guilty," Available at: https://krebsonsecurity.com/2017/12/mirai-iot-botnet-co-authors-plead-guilty/ [retrieved: June, 2025].

[20] Rapid7, "Types of cyber attacks: Hacking attacks and techniques," Available at: https://www.rapid7.com/fundamentals/types-of-attacks/ [retrieved: June, 2025].

[21] A.Remillano II, J. Molina, "Mirai botnet attacks IoT devices via CVE-2020-5902," Available at:

https://www.trendmicro.com/en_us/research/20/g/mirai-botnet-attack-iot-devices-via-cve-2020-5902.html [retrieved: June, 2025].

[22] Gulatas, H. Hakan Kilinc, A. Halim Zaim and M. Ali Aydin, "Malware threat on edge/fog computing environments from Internet of Things devices perspective," Available at: https://ieeexplore.ieee.org/document/10083045 *IEEE Access*, vol. 11, pp. 33584–33606, doi:10.1109/access.2023.3262614 [retrieved: June, 2025].

[23] C. Cimpanu, "BrickerBot author retires claiming to have bricked over 10 million IoT devices," Available at: https://www.bleepingcomputer.com/news/security/Brickerbot-author-retires-claiming-to-have-bricked-over-10-million-iot-devices/ [retrieved: June, 2025/].

[24] NCC Group, "WastedLocker: A new ransomware variant developed by the Evil Corp group," Available at: https://research.nccgroup.com/2020/06/23/wastedlocker-a-new-Ransomware-variant-developed-by-the-evil-corp-group/ [retrieved: June, 2025].

[25] Mitnick Security, "An overview of the 2020 Garmin ransomware attack," Available at: https://www.mitnicksecurity.com/blog/2020-garmin-Ransomware-attack [retrieved: June, 2025].

[26] Mitre, "Amazon CVEs," Available at: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=Amazon+fire+alexa [retrieved: July, 2025].

[27] Mitre, "Apple CVEs," Available at: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=watchOS+%26%26+tvOS [retrieved: July, 2025].

[28] Mitre, "Fitbit CVEs," Available at: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=Fitbit [retrieved: July, 2025].

[29] Mitre, "Google Nest CVEs," Available at: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=Google%2Bnest [retrieved: July, 2025].

[30] J. Stckliey, T. Davis, SOS Daily News, "A whole flock of Google Nest indoor camera issues found," Available at: https://www.sosdailynews.com/?articleid=+305494CF8F5AD05E81214387321385CF [retrieved: July, 2025].

[31] Mitre, "Ring CVEs," Available at: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=Ring%2Bdoorbell [retrieved: July, 2025].

[32] Mitre, "BMW CVEs," Available at: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=BMW [retrieved: July, 2025].

[33] Mitre, "Chrysler/Jeep CVEs," Available at: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=Chrysler [retrieved: July, 2025].

[34] Mitre, "Ford CVEs," Available at: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=Ford [retrieved: July, 2025]

[35] Mitre, "Honda CVEs," Available at: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=Honda [retrieved: July, 2025].

[36] Mitre, "Tesla CVEs," Available at: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=Tesla [retrieved: July, 2025].

[37] JSOF, "RIPPLE20," Available at: https://www.forescout.com/research-labs/ripple20-vulnerability/ [retrieved: July, 2025].

[38] Forescout, "Name:Wreck," Available at: https://www.forescout.com/research-labs/namewreck/ [retrieved: July, 2025].

[39] Forescout, "Nucleus:13," Available at: https://www.forescout.com/research-labs/nucleus-13/ [retrieved: July, 2025].

[40] Forescout, "Amnesia:33," Available at: https://www.forescout.com/research-labs/amnesia33/ [retrieved: July, 2025].

[41] Armis, "Urgent/11," Available at: https://www.armis.com/research/urgent-11/ [retrieved: July, 2025]

[42] B. E. Duan, V. Zhang and K. Ye, "Flocker mobile ransomware crosses to smart TV," Trend Micro, Available at: https://www.trendmicro.com/ru_ru/research/16/f/flocker-Ransomware-crosses-smart-tv.html [retrieved: July, 2025].

[43] D. Atch, N. Frumovich and R.Bevington, Microsoft Threat Intelligence, "Uncovering Trickbot's use of IoT devices in command-and-control infrastructure," Available at: https://www.microsoft.com/en-us/security/blog/2022/03/16/uncovering-trickbots-use-of-iot-devices-in-command-and-control-infrastructure/ [retrieved: July, 2025].

[44] Microsoft Threat Intelligence, "When coin miners evolve, part 1: Exposing LemonDuck and LemonCat," Available at: https://www.microsoft.com/en-us/security/blog/2021/07/22/when-coin-miners-evolve-part-1-exposing-lemonduck-and-lemoncat-modern-mining-malware-infrastructure/ [retrieved: July, 2025].

[45] X. Zhang, O. Upton, N. Beebe and K. Choo, "IoT Botnet Forensics: A Comprehensive Digital Forensic Case Study on Mirai Botnet Servers," Trend Micro. Available at: https://www.sciencedirect.com/science/article/pii/S2666281720300214 [retrieved: July, 2025].

[46] V. Singhal, R. Nigam, Z. Zhang, and A. Davila, "New Mirai variant targeting network security devices," Unit 42. Available at: https://unit42.paloaltonetworks.com/mirai-variant-iot-vulnerabilities/ [retrieved: July, 2025].

[47] Microsoft Threat Intelligence, "When coin miners evolve, part 2: Hunting down LemonDuck and LemonCat attacks," Available at: https://www.microsoft.com/en-us/security/blog/2021/07/29/when-coin-miners-evolve-part-2-hunting-down-lemonduck-and-lemoncat-attacks/ [retrieved: July, 2025].

[48] R. Holt, ESET Security Community, "Sandworm: A tale of disruption told anew," Available at: https://www.welivesecurity.com/2022/03/21/sandworm-tale-disruption-told-anew/ [retrieved: July, 2025].

[49] CISA, "Russian State-Sponsored and Criminal Cyber Threats to Critical Infrastructure," Available at: https://www.cisa.gov/news-events/cybersecurity-advisories/aa22-110a [retrieved: July, 2025].

[50] R. Lipovsky, A. Cherepanov, ESET, "Operation Potao Express," Available at: https://web-assets.esetstatic.com/wls/2015/07/Operation-Potao-Express_final_v2.pdf [retrieved: July, 2025].

[51] Hall, CCDCOE, "Industroyer – Crash Override," Available at: https://cyberlaw.ccdcoe.org/wiki/Industroyer_–_Crash_Override_(2016) (cyberlaw.ccdcoe.org in Bing) [retrieved: July, 2025].

[52] D. Zafra et al, Mandiant, "INDUSTROYER.V2: Old malware learns new tricks," Available at: https://cloud.google.com/blog/topics/threat-intelligence/industroyer-v2-old-malware-new-tricks/ [retrieved: July, 2025].

[53] Greenberg, "Russia's Sandworm hackers attempted a third blackout in Ukraine," Wired. Available at: https://www.wired.com/story/sandworm-russia-ukraine-blackout-gru/ [retrieved: July, 2025].

[54] Splunk, "Threat update: INDUSTROYER2," Available at: https://www.splunk.com/en_us/blog/security/threat-update-industroyer2.html [retrieved: July, 2025].

[55] ESET Ireland, "Industroyer2: Industroyer reloaded," Available at: https://blog.eset.ie/2022/04/12/industroyer2-industroyer-reloaded/ [retrieved: July, 2025].

[56] Microsoft Digital Security Unit, "Special report: Ukraine – An overview of Russia's cyberattack activity," Available at: https://cdn-dynmedia-1.microsoft.com/is/content/microsoftcorp/microsoft/final/en-us/microsoft-brand/documents/1212211-ms-ukrainespecialreport-fy23-link-update.pdf [retrieved: July, 2025].

[57] E. Kreminchuker, M. Zavodchik, "Echobot malware now up to 71 exploits, targeting SCADA," F5 Labs. Available at: https://www.f5.com/labs/articles/threat-intelligence/echobot-

malware-now-up-to-71-exploits--targeting-scada [retrieved: July, 2025].

[58] R. Nigam, Unit 42, "Mirai variant Echobot resurfaces with 13 previously unexploited vulnerabilities," Unit 42. Available at: https://unit42.paloaltonetworks.com/mirai-variant-echobot-resurfaces-with-13-previously-unexploited-vulnerabilities/ [retrieved: August, 2024].

[59] J. Choi et al, "IoT malware ecosystem in the wild," *IoT Malware Ecosystem in the Wild*. Available at: https://dl.acm.org/doi/abs/10.1145/3318216.3363379 doi:10.1145/3318216.3363379 [retrieved: August, 2024].

[60] Mudgerikar and E. Bertino, "IoT attacks and malware," in *Cyber Security Meets Machine Learning*, Springer, pp. 1–27. Available at: https://link.springer.com/book/10.1007/978-981-33-6726-5 [retrieved: August, 2024].

[61] R. Khoury, B. Vignau, S. Hallé, A. Hamou-Lhadj, and A. Raz, "An analysis of the use of CVEs by IoT malware," in *Foundations and Practice of Security*, Springer, pp. 47–64. Available at: https://link.springer.com/book/10.1007/978-3-030-70881-8 [retrieved: August, 2024].

[62] FortiGuard Labs, "Linux/Darlloz.A," Available at: https://www.fortiguard.com/encyclopedia/virus/5902488 [retrieved: August, 2024].

[63] K. Hemsley and R. Fisher, "A history of cyber incidents and threats involving industrial control systems," in *Critical Infrastructure Protection XII*, Springer, pp. 222–249. Available at: https://link.springer.com/chapter/10.1007/978-3-030-04537-1_12 [retrieved: August, 2024].

[64] S. Sharmeen, S. Huda, J. H. Abawajy, W. N. Ismail, and M. M. Hassan, "Malware threats and detection for industrial mobile-IoT networks," *IEEE Access*, vol. 6, pp. 15941–15957, doi:10.1109/ACCESS.2018.2815660. Available at: https://ieeexplore.ieee.org/document/8315029 [retrieved: August, 2024].

[65] G. P. Kachare, G. Choudhary, S. K. Shandilya, and V. Siha, "Sandbox environment for real-time malware analysis of IoT devices," in *Computing Science, Communication and Security*, Springer, pp. 169–183. Available at: https://link.springer.com/chapter/10.1007/978-3-031-10551-7_13 [retrieved: August, 2024].

[66] S. Kakati, D. Chouhan, A. Nag, and S. Panja, "Survey on recent malware detection techniques for IoT," in *Pattern Recognition and Data Analysis with Applications*, Springer, pp. 647–660. Available at: https://link.springer.com/chapter/10.1007/978-981-19-1520-8_53 [retrieved: August, 2024].

[67] K. Stoddart, "On cyberwar: Theorizing cyberwarfare through attacks on critical infrastructure," in *Cyberwarfare Threats to Critical Infrastructure*, Palgrave Macmillan, pp. 53–145. Available at: https://link.springer.com/chapter/10.1007/978-3-030-97299-8_2 [retrieved: August, 2024].

[68] Dragos, "CRASHOVERRIDE: Analysis of the threat to electric grid operations," Available at: https://nsarchive.gwu.edu/sites/default/files/documents/3869008/Dragos-CRASHOVERRIDE-Analyzing-the-Threat-to.pdf [retrieved: August, 2024].

[69] M. Geiger, J. Bauer, M. Masuch, and J. Franke, "An analysis of BlackEnergy, CrashOverride, and Trisis," in *IEEE ETFA*, pp. 1537–1543, doi:10.1109/ETFA46521.2020.9212128. Available at: https://ieeexplore.ieee.org/document/9212128 [retrieved: August, 2024].

[70] Nozomi Networks, "Industroyer2: Nozomi Networks Labs analyzes the IEC-104 payload," Available at: https://www.nozominetworks.com/blog/industroyer2-nozomi-networks-labs-analyzes-the-iec-104-payload [retrieved: August, 2024].

[71] CrowdStrike Global Intelligence Team, "Use of FancyBear Android malware in tracking Ukrainian field artillery units," Available at: https://www.crowdstrike.com/wp-content/brochures/FancyBearTracksUkrainianArtillery.pdf [retrieved: August, 2024].

[72] U. Shamir, SentinelOne, "Analyzing a new variant of BlackEnergy 3 likely insider-based execution," Available at: https://www.scribd.com/document/421931582/BlackEnergy3-WP-012716-1c [retrieved: August, 2024].

[73] Y. Mekdad, G. Bernieri, M. Conti and AE. Fergougui, "The rise of ICS malware: A comparative analysis," Springer. Available at: https://link-springer-com.libproxy.abertay.ac.uk/chapter/10.1007/978-3-030-95484-0_29 [retrieved: August, 2024].

[74] K. Hsu, Z. Z. Zhang and R. Nigam, "New Mirai variant targets Zyxel network-attached storage devices," Unit 42. Available at: https://unit42.paloaltonetworks.com/new-mirai-variant-mukashi/ [retrieved: August, 2024].

[75] Famera, B. Hilger, S. Bhunia and P. Heil, "Analyzing the Mirai IoT botnet and its recent variants: Satori, Mukashi, Moobot, and Sonic," arXiv.org. Available at: https://arxiv.org/abs/2508.01909v1 [retrieved: August, 2024].

[76] Checkpoint "November 2024's most wanted malware: Androxgh0st leads the pack, targeting IoT devices and critical infrastructure," Check Point Blog. Available at: https://blog.checkpoint.com/research/november-2024s-most-wanted-malware-androxgh0st-leads-the-pack-targeting-iot-devices-and-critical-infrastructure/ [retrieved: August, 2024].

[77] CISA, "Known indicators of compromise associated with Androxgh0st malware," Available at: https://www.cisa.gov/news-events/cybersecurity-advisories/aa24-016a?=&web_view=true [retrieved: August, 2024].

[78] Picus Labs, "Androxgh0st malware: Unmasking the silent threat to cloud and web security," Available at: https://www.picussecurity.com/resource/blog/androxgh0st-malware-cloud-web-security-threat [retrieved: August, 2024].

[79] CISA, "VPNFilter Destructive malware," Available at: https://www.cisa.gov/news-events/alerts/2018/05/23/vpnfilter-destructive-malware [retrieved: September, 2025]

[80] Mitre, "Acid Rain," Available at: https://attack.mitre.org/software/S1125/ [retrieved: September, 2025].

A Smart-Contract–Based Validation Framework for Secure and Auditable Federated Learning in Dementia

Elif Calik
ADAPT Research Centre,
School of Computer Science,
University of Galway,
Galway, Ireland
email: elif.calik@universityofgalway.ie

Ayse Keles
ADAPT Research Centre,
School of Computer Science,
University of Galway,
Galway, Ireland
email: ayse.keles@universityofgalway.ie

Malika Bendechache
ADAPT Research Centre,
School of Computer Science,
University of Galway,
Galway, Ireland
email: malika.bendechache@universityofgalway.ie

***Abstract*— Typically arising from neurodegenerative diseases (most notably Alzheimer's disease, Lewy body dementia, frontotemporal dementia, and Parkinson's disease), dementia diagnosis and prognosis increasingly leverage Machine Learning (ML) across heterogeneous data modalities—including neuroimaging, structured Electronic Health Records (EHRs), and emerging digital biomarkers. However, strict privacy regulations and institutional barriers impede data pooling, while cross-site heterogeneity undermines model robustness. We present a permissioned-blockchain-enabled Federated Learning (FL) framework that addresses these challenges through a validation-first design grounded in a consensus Minimum Dataset (MDS) for dementia. Participating hospitals map local EHR/imaging fields to the shared MDS and perform local rule checks before training. A lightweight smart contract records hash-anchored validation receipts on-chain, creating an immutable audit trail without exposing raw data. Provenance links are maintained off-chain using content-addressed objects (e.g., InterPlanetary File System - IPFS) for EHR/imaging pointers and model artifacts; their identifiers are anchored on-chain to ensure verifiable version control. Local models are trained exclusively on validated records; contributions are aggregated (e.g., weighted Federated Averaging (FedAvg)) to produce a hash-verified global model distributed uniformly to all sites. The framework is technology-agnostic, minimizes operational overhead by storing only digests on chain, and directly targets key clinical adoption requirements: secure collaboration, reproducibility, and output compatibility under non-Independent Identically Distributed (non-IID) conditions. By standardizing input pre-training and enforcing transparent, tamper-evident exchanges, the approach aims to improve aggregation stability, cross-site generalizability, data quality, and information-governance compliance in dementia ML workflows.**

Keywords*— *Federated learning; blockchain; smart contract; data quality; neurodegenerative disease; dementia minimum-dataset.

I. INTRODUCTION

Dementia, which is marked by an irreversible decline in cognitive functions, such as memory, language, and decision-making, and is typically driven by progressive neurodegeneration, remains difficult to diagnose and forecast at scale. Early and precise diagnosis is crucial for timely and effective interventions, especially during the initial stages of the disease [1]. This need has fueled a growing interest in using Machine Learning (ML) models to assist clinicians with disease classification, progression prediction, and personalized care [2]. In ML applications, diverse data modalities are increasingly leveraged, including neuroimaging (e.g., Magnetic Resonance Imaging (MRI) to measure brain atrophy, a hallmark of neurodegenerative diseases) [3], with models, such as Support Vector Machines (SVMs) achieving high Alzheimer's Disease (AD) classification accuracy [4], structured clinical data from EHRs capturing demographics, labs, and diagnoses for progression prediction (e.g., Mild Cognitive Impairment (MCI) to AD) [5] , and emerging digital biomarkers, such as spontaneous speech [6] and wearable device data for real-time monitoring and early detection of cognitive decline [7].

However, a major obstacle is the scarcity and lack of diversity in datasets, which hinders the development of generalizable ML models. Pooling large volumes of patient data into a single repository for training is often impossible due to strict privacy regulations and institutional barriers, creating what are known as "data islands" or silos [8]. FL has emerged as a promising solution to these challenges [9][10]. Its core principle is to facilitate collaborative model training across decentralized institutions without the need for centralizing or explicitly exchanging raw patient data [11]. Instead, the ML algorithm is brought to the data, a process that inherently addresses patient privacy concerns and fosters trust [12]. In FL, a shared global model is trained on multiple local datasets, with the data samples remaining on their respective nodes [13]. Only model updates, such as the weights and biases of a deep neural network, are exchanged between local nodes and a central server or among the nodes themselves. This decentralized approach aligns with "Zero Trust", as it shares only model parameters without exposing raw data.

Two primary architectural paradigms exist in FL: client-server and peer-to-peer [14]. In the most common is the client-server or centralized model. A central server orchestrates the training process by distributing a global model to clients, collecting their locally trained model parameters, and aggregating them to create an improved global model. In contrast, the peer-to-peer architecture, nodes coordinate directly with their neighbours to obtain the global model by exchanging updates [14].

A major technical challenge in FL is data heterogeneity, which violates the independent and identically distributed (IID) assumption of traditional ML, especially prevalent in healthcare, where different hospitals may serve distinct patient populations or use varying equipment and diagnostic criteria [15]. Statistical heterogeneity in FL arises from the non-IID nature of healthcare data, where distributions vary significantly across institutions or devices due to differences in patient populations, clinical practices, and data acquisition protocols. Such variability leads to skewed data distributions that challenge model convergence, reduce generalizability, and limit the effectiveness of standard aggregation algorithms like FedAvg [16].

A central requirement for clinical adoption of FL is the ability to ensure both secure collaboration and consistent model performance across heterogeneous healthcare institutions. Blockchain-enhanced FL has already demonstrated significant promise by addressing privacy, secure parameter sharing, transparency, and accountability

through encrypted model updates, secure aggregation, peer-to-peer transfer protocols, and immutable audit trails [17]. However, a critical gap in blockchain-enabled FL for healthcare is the lack of robustness to heterogeneity, which is especially pronounced in dementia care, where data sources range from cognitive tests and neuroimaging to speech, gait, and wearables sensor data [7]. In this consortium setting, a permissioned blockchain can be used not as a compute substrate, but as a shared, tamper-evident coordination and provenance layer across independent hospitals. A classic centralized architecture (e.g., a single coordinator with a conventional database and logs) can orchestrate FL, but it implicitly concentrates trust: one operator becomes responsible for enrollment, event ordering, and the integrity of validation and model-version records, and disputes about "who submitted what and when" ultimately depend on that operator's logs. In contrast, anchoring validation receipts and model version identifiers on a permissioned ledger creates a jointly verifiable audit trail that no single site can unilaterally rewrite, while still keeping patient-level content off-chain. We adopt a permissioned design to align with hospital governance (authenticated membership, controlled read/write access) and to keep on-chain activity lightweight—only fixed-size digests and content identifiers (CIDs) are recorded—while model artifacts and pointers remain off-chain.

To address these gaps, our framework introduces a smart contract–based mechanism that standardizes data representation and enforces interoperability prior to local training. By harmonizing input features through a lightweight, verifiable smart contract framework, we reduce distributional discrepancies between participating institutions, thereby facilitating more robust aggregation and improving cross-site generalizability. This combined use of blockchain for secure collaboration and smart contracts for heterogeneity management establishes (i) robustness of aggregation to noise, bias, and malicious behavior, (ii) consistency and reproducibility of global outputs across heterogeneous sites, and (iii) a traceable, accountable process that meets information-governance expectations.

The rest of this paper is structured as follows. Section II surveys the relevant state of the art, Section III details the proposed research methodology, and Section IV presents the conclusions and discusses avenues for future work.

II. Related Work

Integrating blockchain with FL in AI-assisted brain imaging offers a promising direction for secure, collaborative, and regulation-compliant analytics in neurology [18]-[20]. Blockchain complements FL by adding a tamper-proof ledger, transparent coordination through smart contracts, and decentralized control over model sharing. Together, these features address long-standing issues, such as a lack of trust, weak data integrity, limited auditability, and vulnerability to adversarial manipulation that can undermine multi-institutional learning in healthcare.

The strength of Bhatia et al. [18] lies in its explicit operationalization of "data quality" as a measurable, pre-aggregation criterion. Their decentralized data-evaluation mechanism deploys miners to execute submitted local models against a concealed validation set and admit only those updates that exceed a predefined accuracy threshold. This "quality-gating" approach protects aggregation from data poisoning and label-noise attacks, stabilizes performance under heterogeneity, and preserves fidelity to the clinical signal. Its limitation, however, is the reliance on curator-held validation data and threshold design, which necessitate careful governance to avoid bias toward particular case mixes or scanner characteristics.

While the design of Imboccioli et al. [19] does not intrinsically judge the "quality" of an update by its predictive merit, it enhances the verifiability, provenance, and process fidelity that underpin trust in distributed clinical AI. Rather than evaluating accuracy explicitly at the gate, their framework orchestrates training phases via an immutable state machine, anchors cryptographic digests of model parameters on-chain, and coordinates storage of ciphered parameters (e.g., via IPFS) for reliable retrieval. This phased, auditable mechanism compels every participant—including the aggregator—to follow the same rules, thereby inhibiting out-of-protocol actions, covert parameter tampering, or selective sharing. A notable implication for output compatibility is that every collaborating hospital retrieves an identical, hash-verified global model, which curtails divergence due to inconsistent versions or update timing.

Third, Rajit et al. [20] approach safeguards against in-transit manipulation and supports traceability of each contribution, thereby reinforcing both data quality (through integrity guarantees) and output compatibility (through consistent application of an auditable, deterministic aggregator). They emphasize secure, auditable integration of client contributions with a focus on handling non-IID data distributions often observed across medical centers. Their architecture transforms each client's weight updates into SHA-256 hashes appended to the blockchain, enabling immutable provenance and tamper-evident transport. Aggregation proceeds via a weighted FedAvg scheme, producing a global model that reflects the relative information content contributed by each site. Importantly, the authors examine non-IID scenarios and introduce an accelerated aggregation variant to expedite convergence while maintaining accuracy—an attractive property for time-sensitive clinical deployments.

Taken together, these studies illustrate instructive trade-offs. Accuracy-based gating [18] directly controls predictive quality but hinges on governance of validation data and thresholds. Protocol-centric integrity [19] ensures observability, reproducibility, and non-repudiation but leaves performance vetting to external mechanisms. Hash-anchored, weighted aggregation [20] prioritizes provenance and distributional robustness, yet presumes acceptable upstream quality in local training and curation.

Building on these insights, this study advances a blockchain-enhanced FL framework for dementia imaging that treats data quality and output compatibility as first-class design objectives. Unlike prior work, our approach introduces smart contract–driven coordination to integrate quality control with verifiability, thereby aligning predictive performance, process integrity, and clinical trust.

To sum up, we propose a dementia-oriented blockchain-enabled FL workflow that couples validation-first interoperability (via an MDS) with tamper-evident provenance and versioning, aiming to reduce representation-driven heterogeneity prior to training while keeping patient data local.

III. Material and Methods

In multidisciplinary healthcare research, a persistent challenge is the limited opportunity for synchronous, in-person collaboration with clinicians, who typically operate on a 24/7 schedule. In domains dealing with complex and

sensitive data, establishing minimum datasets (MDS) for diagnosis is therefore a critical, consensus-driven step that enables coordinated progress. However, disease-specific minimum datasets are not always available. In this context, to enhance input quality for FL and to promote output consistency, we developed a dementia MDS. A smart-contract–based proposed framework was subsequently provided.

When constructing the MDS, we systematically reviewed the International Classification of Diseases 11th (ICD-11) [21] sub-diagnostic groups associated with dementia: (A) Alzheimer disease—(i) early-onset, (ii) late-onset, (iii) mixed with cerebrovascular disease, (iv) mixed with other non-vascular aetiologies, (v) unspecified onset; (B) Vascular dementia; (C) Dementia with Lewy bodies; (D) Frontotemporal dementia; (E) Substance/medication-induced dementia—(i) alcohol, (ii) sedative/hypnotic/anxiolytic, (iii) volatile inhalants, (iv) other specified; (F) Dementia due to other diseases—(i) Parkinson disease, (ii) Huntington disease, (iii) exposure to heavy metals/other toxins, (iv) human immunodeficiency virus (HIV), (v) multiple sclerosis, (vi) prion disease, (vii) other specified cause, (viii) unspecified cause. Note: Code (G)—behavioural/psychological symptoms in dementia—is a supplementary code and not a primary aetiology.

The MDS includes key sociodemographic variables [22]: age (treated as a primary covariate for dementia risk and progression), sex (included because phenotypes and biomarker levels can differ by sex), and total years of formal education (as a proxy for cognitive reserve). To capture cognitive status, we incorporated Mini-Mental State Examination (MMSE) (global cognition), Montreal Cognitive Assessment (MoCA) (sensitive to mild cognitive impairment), Clinical Dementia Rating—Global Score (CDR_global) (ordinal staging of cognitive/functional impairment), Clinical Dementia Rating—Sum of Boxes (CDR_sb) (a composite sensitive to subtle clinical change), and Instrumental Activities of Daily Living (IADL_total) (functional autonomy in complex daily tasks). Neuropsychiatric symptoms are assessed using the Geriatric Depression Scale—Short Form (GDS_total) for depressive symptoms and the Neuropsychiatric Inventory Questionnaire (NPIQ_total) for the overall burden of behavioural/psychological symptoms. Comorbidities included are hypertension (HT), reflecting vascular risk associated with cognitive decline; diabetes mellitus (DM), reflecting metabolic risk; atrial fibrillation (AF), reflecting cardioembolic/low-perfusion risk; and history of stroke/transient ischaemic attack, capturing prior cerebrovascular events. Vital signs and laboratory measures comprise systolic blood pressure (SBP) (vascular risk profiling), diastolic blood pressure (DBP) (related to cerebral perfusion), body mass index (BMI) (nutritional/metabolic status relevant to prognosis), thyroid-stimulating hormone (TSH) (screens for reversible cognitive effects), vitamin B12 (B12) (deficiency can cause treatable cognitive impairment), haemoglobin A1c (HbA1c) (medium-term glycaemic control), and low-density lipoprotein cholesterol (LDL) (marker of atherosclerotic burden). Imaging variables: MRI Protocol / Modality Name (mri_protocol_name), Slice orientation and acquisition plane (acq_plane), Scanner Manufacturer (scanner_vendor), Series completeness (srs_comp), Field of View (FOV), and Matrix size (matrix_size). Blood-based biomarkers [23][24] include plasma phosphorylated tau-181 (plasma_ptau181) (associated with Alzheimer pathology), plasma neurofilament light chain (plasma_nfl) (marker of neuro-axonal injury and prognosis), and the amyloid-beta 42/40 ratio (abeta42_40) (a peripheral amyloid signature with diagnostic/prognostic value). The apolipoprotein E ε4 (apoe_e4) carrier status indicates whether at least one ε4 allele is present [22]-[25]. Finally, digital measures comprise gait speed (GS) (a functional digital marker linked to cognitive decline), mean daily step count (MDSC) (a proxy for physical activity and overall health), and speech pause rate (SPR) (a digital marker of language/fluency changes). Table 1 presents the structure of the constructed MDS, summarizing each variable by category, description, variable name, requirement level, and data type. Consistent with the requirement stratification of the MDS, variables are organized into a core, flexible, and extensible framework: variables designated as “Yes” constitute the minimum indispensable elements for consistent characterization; “Conditional” items provide methodological flexibility where equivalent instruments are acceptable; “Recommended” variables enhance clinical and analytic utility when available; and “Optional” measures allow scalable enrichment without compromising baseline interoperability across heterogeneous resource environments.

We built a practical, end-to-end workflow that lets hospitals collaborate on dementia models without moving patient data. The idea is simple: every hospital prepares its data in the same way, proves on a blockchain that it meets agreed-upon rules, trains locally, and then shares only model files (never raw data) for aggregation. In our proposed implementation, the coordination and provenance layer is realized using a permissioned Hyperledger Fabric network, where participating hospitals join as authenticated members under a shared governance model. The workflow logic is implemented as a Fabric chaincode written in TypeScript, exposing functions to (i) record MDS-validation receipts as fixed-size hashes and metadata, (ii) register model-version identifiers per training round, and (iii) enforce role-based access control and monotonic versioning to prevent unauthorized submissions or replay. Model artifacts themselves will remain off-chain (e.g., stored via content-addressed storage such as IPFS), while the ledger stores only cryptographic digests and CIDs to keep on-chain activity lightweight and auditable.

As shown in Figure 1, the proposed smart contract–based framework follows a validation-first pipeline: (i) each hospital validates its data against the agreed minimum dataset (MDS) and anchors validation hashes on the Hyperledger Fabric ledger; (ii) EHR and imaging artifacts remain off-chain in IPFS and are referenced via CIDs linked to those hashes; (iii) local training is performed strictly on records with on-chain validation receipts; (iv) local model updates are stored in IPFS and their versions are anchored on-chain; and (v) the aggregator builds the global model, stores it in IPFS, and anchors the released version on-chain so that every site retrieves the same, hash-verified artifact.

TABLE I. MINIMUM DATASET FOR DEMENTIA DIAGNOSIS

Category	Variable description	Variable name	Requirement level	Type
Clinical Diagnosis (ICD-11)	Primary ICD-11 code	icd11_dementia_code	Recommended	string
Socio Demographics	Age in years at baseline	age_years	Yes	integer
	Biological sex	sex	Yes	categorical
	Total years of formal education	education_years	Yes	integer
Cognitive Status	Mini-Mental State Examination	mmse_total	Conditional (MMSE or MoCA)	integer
	Montreal Cognitive Assessment	moca_total	Conditional (MMSE or MoCA)	integer
Clinical Staging and Function	Clinical Dementia Rating - Global Score	cdr_global	Conditional (CDR or CDR-SB)	ordinal
	Clinical Dementia Rating - Sum of Boxes	cdr_sb	Conditional (CDR or CDR-SB)	float
	Instrumental Activities of Daily Living	iadl_total	Yes	integer
Neuropsychiatric Symptoms	Geriatric Depression Scale - Short Form	gds_total	Conditional (GDS or NPI-Q)	integer
	Neuropsychiatric Inventory Questionnaire	npiq_total	Conditional (GDS or NPI-Q)	integer
Comorbidities	Hypertension	ht	Yes	binary
	Diabetes Mellitus	dm	Yes	binary
	Atrial Fibrillation	af	Yes	binary
	History of Stroke / Transient Ischaemic Attack	stroke_history	Yes	binary
Vitals and Labs	Systolic Blood Pressure	sbp_mmhg	Recommended	integer
	Diastolic Blood Pressure	dbp_mmhg	Recommended	integer
	Body Mass Index	bmi_kg_m2	Recommended	float
	Thyroid-Stimulating Hormone	tsh_mIU_L	Recommended	float
	Vitamin B12	b12_pg_mL	Recommended	float
	Hemoglobin A1c	hba1c_percent	Recommended	float
	Low-Density Lipoprotein Cholesterol	ldl_mmol_L	Recommended	float
Imaging	MRI Protocol / Modality Name	mri_protocol_name	Yes	string
	Slice orientation and acquisition plane	acq_plane	Yes	string
	Scanner Manufacturer	scanner_vendor	Yes	string
	Series completeness	srs_comp	Yes	boolean
	Field of View	fov	Yes	float
	Matrix_size	matrix_size	Optional	list
Blood Biomarkers	Plasma Phosphorylated Tau-181	plasma_ptau181_pg_mL	Optional	float
	Plasma Neurofilament Light Chain	plasma_nfl_pg_mL	Optional	float
	Amyloid-beta 42/40 Ratio	abeta42_40_ratio	Optional	float
Genetics	Apolipoprotein E e4 Carrier Status	apoe_e4_carrier	Optional	binary
Digital Measures	Gait Speed	gait_speed_m_s	Optional	float
	Mean Daily Step Count	daily_step_count_mean	Optional	integer
	Speech Pause Rate	speech_pause_rate	Optional	float

We next describe each step in detail in the following section (Figure 1).

Step 1 — Agree on the minimum dataset (MDS): Before anything else, participating hospitals align on a compact list of variables that are most informative for dementia (e.g., age, sex, education, cognitive and functional scales, key comorbidities, selected labs, MRI descriptors, and optional blood/digital biomarkers). We encode this list as a machine-readable JSON schema that defines types (integer, float, categorical), valid ranges, and simple "either/or" rules (for example, MMSE or MoCA; CDR-Global or CDR-Sum of Boxes). This schema becomes the sole source of truth for what "valid input" means across all sites.

Step 2 — Validate inputs with smart contracts (on-chain): Each hospital checks its EHR entries and related imaging metadata against the MDS. Rather than trusting a central service, we use a permissioned blockchain with a lightweight smart contract that exposes a validate() function. To keep the contract lightweight, schema conformance checks are executed locally using the agreed JSON schema and validator version, and the chaincode records a signed, time-ordered receipt that binds (a) the schema/version identifier, (b) a digest of the validation outcome, and (c) the submitting member identity to a training round. This creates an immutable receipt that the input met the agreed-upon rules, without revealing sensitive content.

Step 3 — Keep pointers off-chain, link them on-chain: To preserve privacy, the actual EHR locators (where the record lives in the hospital system) and imaging identifiers are stored off-chain in the IPFS. IPFS gives each object a CID. We then store the hash of that CID on the blockchain alongside the validation event. Later, anyone can prove that the same object was used, simply by recomputing its hash—yet no clinical details appear on the chain.

Step 4 — Train locally on validated caches: Only records that have a valid on-chain receipt are admitted to the hospital's training cache. Each institution trains its own model on-premises; no patient-level data leaves the site. This step supports both client–server FL and peer-to-peer FL.

Step 5 — Share models via IPFS, anchor versions on chain: After local training, the site packages its model weights (and, if desired, optimizer state) and uploads them to IPFS. The resulting CID is hashed and committed to the blockchain (e.g., via commitModel()), creating a permanent, tamper-evident version tag. The aggregator (or peers) read these on-chain events to discover the latest models and fetch them from IPFS.

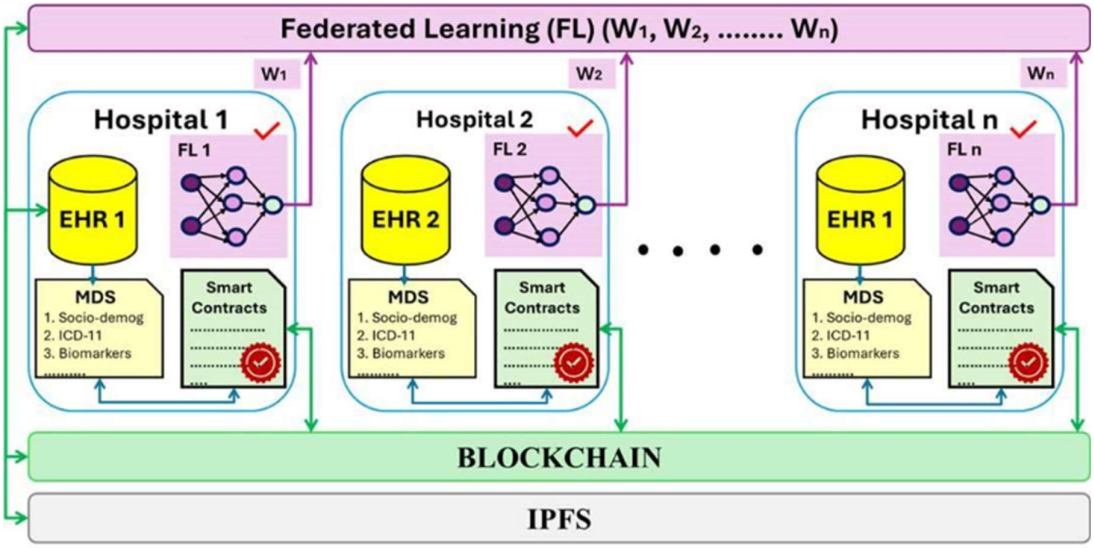


Figure 1. Proposed Smart Contract-based framework

Step 6 — Aggregate and redistribute a single, verified global model: The default aggregator is weighted FedAvg, but robust alternatives can be used without changing the attestation layer. The aggregated global model is also stored in IPFS, and its version hash is written on the chain. This guarantees every site downloads the same, hash-verified global model, avoiding silent version drift.

On the other hand, threat model, attack surface, and evaluation plan can be articulated as follows: We consider a consortium setting in which participating hospitals are authenticated members of a permissioned network, yet a subset of clients may behave maliciously. Assets include patient-level data (kept on premises), the correctness of MDS validation outcomes, model updates, and the integrity of global model releases and version lineage. Adversaries may attempt (i) poisoning or backdoor updates, (ii) replay of stale submissions, (iii) unauthorized contract invocation, and (iv) denial-of-service against validation or model-commit endpoints; additionally, the network and off-chain storage are not assumed confidential or highly available. Under this threat model, the ledger layer primarily targets integrity, accountability, and reproducibility by anchoring tamper-evident validation receipts and model-version identifiers, while privacy against inference attacks and robustness against poisoning require complementary mechanisms (e.g., secure aggregation, differential privacy, and robust aggregation). From an implementation perspective (i.e., Hyperledger Fabric, TypeScript chaincode), we explicitly bound the smart-contract attack surface by enforcing role-based access control, monotonic round/version checks to prevent replay or rollback, fixed-size inputs to limit on-chain payloads, and rate limits to reduce DoS risk. Because IPFS provides integrity via content addressing but not confidentiality or guaranteed availability, off-chain artifacts are protected through encryption and consortium pinning/replication policies, supported by operational controls for membership management, key rotation, and revocation. Finally, we will evaluate the framework through (a) learning outcomes (AUC/F1, sensitivity/specificity, calibration, and convergence), (b) systems overhead (end-to-end round time, Fabric endorsement/commit latency and throughput, communication volume per round, ledger growth, and IPFS retrieval/pinning costs), and (c) adversarial robustness by injecting realistic attacks (label-flipping, backdoor triggers, sign-flip/scale, and replay) at varying fractions of malicious clients; we will report attack success rate, performance degradation, and "when defenses are enabled" detection accuracy and robustness of the final global model.

The key contributions of the proposed Smart Contract-based frameworks are as follows.

i. Security and privacy: Raw data never moves; only fixed-size digests and CIDs are recorded on the permissioned ledger.
ii. Auditability and reproducibility: Every key step (input validation, model commits, global releases) leaves a signed, time-ordered trail.
iii. Consistency across sites: Because the same schema gates inputs and the same versioned model is redistributed, results are easier to reproduce and compare.
iv. Heterogeneity control: Standardizing inputs before training reduces avoidable site-to-site differences, helping aggregation work better.

IV. Conclusion and Future Work

We introduced a pragmatic, end-to-end workflow for multi-institutional dementia modeling that combines FL with a permissioned blockchain and an MDS-driven validation layer. The design secures collaboration without centralizing patient-level data, provides deterministic provenance via on-chain hashes and off-chain content addresses, and reduces avoidable distributional drift by standardizing inputs prior to local training. Anchoring both input validation and model versioning on the chain yields a reproducible lineage for all contributions and ensures every site receives the same, hash-verified global model. Collectively, these properties address persistent blockers to clinical deployment—trust, auditability, and cross-site consistency—while remaining lightweight and compatible with existing hospital governance.

In our future work, we will conduct prospective, multi-site pilots to quantify accuracy, calibration, convergence, and operational burden under realistic non-IID conditions. We will also validate the proposed architectural framework through simulation studies and/or prototype implementations, enabling a clearer assessment of feasibility and any measurable performance gains over baseline FL. To make

these comparisons interpretable, we will isolate the effect of the MDS validation layer by evaluating (i) standard FL without schema gating, (ii) MDS-gated FL using a classic architecture (conventional logging rather than on-chain receipts), and (iii) the full permissioned-ledger design with on-chain validation receipts and model version anchoring. We will harden privacy and robustness with secure aggregation, differential privacy, and robust aggregators, guided by a formal threat model and adversarial testing. We will broaden modality support (in the dementia use-case; speech, gait, wearables), provide interoperability tooling (EHR/Fast Healthcare Interoperability Resources (FHIR) mappings, validators), and mature lifecycle operations (drift monitoring, version pinning/rollback) to enable sustained clinical deployment.

COMPETING INTERESTS:

The authors declare no conflicts of interest.

ACKNOWLEDGMENT

This research was funded by Digi+ Marie Skłodowska-Curie Action (MSCA) CoFund under grant number 101081609, and supported by Taighde Éireann - Research Ireland under Grant number 13/RC/2106/P_2 (ADAPT Centre), and the European Union's Horizon 2020 Research and Innovation Programme under the Marie Skłodowska-Curie Action (MSCA) grant agreement No. 101152004-101109961. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

REFERENCES

[1] B. Xin, D. Zhang, H. Fu, and W. Jiang, "Association between multimorbidity and the risk of dementia: a systematic review and meta-analysis," Arch. Gerontol. Geriatr., vol. 131, Art. no. 105760, Apr. 2025, doi: 10.1016/j.archger.2025.105760.

[2] S. Dattola, A. Ielo, G. Varone, A. Cacciola, A. Quartarone, and L. Bonanno, "Frontotemporal dementia: a systematic review of artificial intelligence approaches in differential diagnosis," Front. Aging Neurosci., vol. 17, Art. no. 1547727, 2025, doi: 10.3389/fnagi.2025.1547727.

[3] D.-H. Shih, Y.-H. Wu, T.-W. Wu, Y.-K. Wang, and M.-H. Shih, "Classifying dementia severity using MRI radiomics analysis of the hippocampus and machine learning," IEEE Access, vol. 12, pp. 160030–160051, 2024, doi: 10.1109/ACCESS.2024.3483833.

[4] S. Klöppel et al., "Automatic classification of MR scans in Alzheimer's disease," Brain, vol. 131, no. 3, pp. 681–689, 2008, doi: 10.1093/brain/awm319.

[5] J. Pan, Z. Fan, G. E. Smith, Y. Guo, J. Bian, and J. Xu, "Federated learning with multi-cohort real-world data for predicting the progression from mild cognitive impairment to Alzheimer's disease," Alzheimers Dement., vol. 21, no. 4, Art. no. e70128, Apr. 2025, doi: 10.1002/alz.70128.

[6] X. Ouyang, "Design and deployment of multi-modal federated learning systems for Alzheimer's disease monitoring," in Proc. 21st Annu. Int. Conf. Mobile Syst., Appl. Serv. (MobiSys), Helsinki, Finland, 2023, pp. 612–614, doi: 10.1145/3581791.3597505.

[7] G. Cornelius, W. Hodgson, R. Maguire, and K. Egan, "Wearable technology, smart home systems, and mobile apps for the self-management of patient outcomes in dementia care: systematic review," J. Med. Internet Res., vol. 27, Art. no. e65385, 2025, doi: 10.2196/65385.

[8] A. Mitrovska, P. Safari, K. Ritter, B. Shariati, and J. K. Fischer, "Secure federated learning for Alzheimer's disease detection," Front. Aging Neurosci., vol. 16, Art. no. 1324032, Mar. 2024, doi: 10.3389/fnagi.2024.1324032.

[9] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," Knowl.-Based Syst., vol. 216, Art. no. 106775, Mar. 2021, doi: 10.1016/j.knosys.2021.106775.

[10] S. Zhan, L. Huang, G. Luo, S. Zheng, Z. Gao, and H.-C. Chao, "A review on federated learning architectures for privacy-preserving AI: lightweight and secure cloud–edge–end collaboration," Electronics (Basel), vol. 14, no. 13, Art. no. 2512, 2025, doi: 10.3390/electronics14132512.

[11] M. I. Sharif, M. Mehmood, M. P. Uddin, K. Siddique, Z. Akhtar, and S. Waheed, "Federated learning for analysis of medical images: a survey," J. Comput. Sci., vol. 20, no. 12, pp. 1610–1621, Oct. 2024, doi: 10.3844/jcssp.2024.1610.1621.

[12] R. Seyghaly, J. Garcia, and X. Masip-Bruin, "A comprehensive architecture for federated learning-based smart advertising," Sensors (Basel), vol. 24, no. 12, Art. no. 3765, 2024, doi: 10.3390/s24123765.

[13] H. Guan, P.-T. Yap, A. Bozoki, and M. Liu, "Federated learning for medical image analysis: a survey," Pattern Recognit., vol. 151, Art. no. 110424, Jul. 2024, doi: 10.1016/j.patcog.2024.110424.

[14] L. Shanmugam, R. Tillu, and M. Tomar, "Federated learning architecture: design, implementation, and challenges in distributed AI systems," J. Knowl. Learn. Sci. Technol., vol. 2, no. 2, pp. 371–384, 2023, doi: 10.60087/jklst.vol2.n2.p384.

[15] M. Nasajpour et al., "Federated learning in smart healthcare: a survey of applications, challenges, and future directions," Electronics (Basel), vol. 14, no. 9, Art. no. 1750, 2025, doi: 10.3390/electronics14091750.

[16] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in Proc. 20th Int. Conf. Artif. Intell. And Statist. (AISTATS), 2017.

[17] N. Nezhadsistani, N. S. Moayedian, and B. Stiller, "Blockchain-enabled federated learning in healthcare: survey and state-of-the-art," IEEE Access, vol. 13, pp. 119922–119945, 2025, doi: 10.1109/ACCESS.2025.3587345.

[18] L. Bhatia and S. Samet, "A decentralized data evaluation framework in federated learning," Blockchain: Res. Appl., vol. 4, no. 4, Art. no. 100152, 2023, doi: 10.1016/j.bcra.2023.100152.

[19] F. Imboccioli, G. Cialone, and S. Ferretti, "Decentralization of learning and trust in healthcare: blockchain-driven federated learning for Alzheimer's MRI image classification," in 2024 IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops), 2024, pp. 739–744, doi: 10.1109/PerComWorkshops59983.2024.10502820.

[20] S. Rajit, Z. F. Ananna, M. M. Ehsan, N. N. Punom, and S. Siddique, "Multi-class brain tumor classification of MRI image using federated learning with blockchain," in 2024 IEEE Region 10 Symp. (TENSYMP), 2024, pp. 1–8, doi: 10.1109/TENSYMP61132.2024.10752160.

[21] World Health Organization, "International classification of diseases (ICD): ICD-11 for mortality and morbidity statistics," [Online]. Available: WHO ICD-11. Browser: https://icd.who.int/browse11. Accessed: Mar. 8, 2026.

[22] W. Qi et al., "Mapping knowledge landscapes and emerging trends in AI for dementia biomarkers: bibliometric and visualization analysis," J. Med. Internet Res., vol. 26, Art. no. e57830, Aug. 2024, doi: 10.2196/57830.

[23] T. Sekimori, K. Fukunaga, D. I. Finkelstein, and I. Kawahata, "Advances in blood biomarkers and diagnosis approaches for neurodegenerative dementias and related diseases," J. Integr. Neurosci., vol. 23, no. 10, Art. no. 188, Oct. 2024, doi: 10.31083/j.jin2310188.

[24] F. Santos, V. Cabreira, S. Rocha, and J. Massano, "Blood biomarkers for the diagnosis of neurodegenerative dementia: a systematic review," J. Geriatr. Psychiatry Neurol., vol. 36, no. 4, pp. 267–281, Jul. 2023, doi: 10.1177/08919887221141651.

[25] E. Solje, A. Benussi, E. Buratti, A. M. Remes, A. Haapasalo, and B. Borroni, "State-of-the-art methods and emerging fluid biomarkers in the diagnostics of dementia—a short review and diagnostic algorithm," Diagnostics, vol. 11, no. 5, Art. no. 788, 2021, doi: 10.3390/diagnostics11050788.

SoK: Toward Protecting Internet-Accessible Legacy Systems

William Yurcik[†‡]
Centers for Medicare & Medicaid Services
Baltimore, MD USA
Email: william.yurcik@cms.hhs.gov

Gregory Koenig
Independent Researcher
Atlanta, GA USA
Email: koenig@acm.org

Gregory Pluta
University of Illinois at Urbana-Champaign
Champaign, IL USA
Email: gpluta@illinois.edu

Gianni Pezzarossi
Stuart Turner
University of Illinois at Urbana-Champaign
Champaign, IL USA
Email: {gpezza2, slturne1}@illinois.edu

Fabio Roberto de Miranda
Luciano Pereira Soares
Insper
São Paulo, SP Brazil
Email: {fabiomiranda, lucianops}@insper.edu.br

***Abstract*— The security problem with legacy systems is large, persistent, and structurally embedded in critical sectors. It is not just a patching problem but rather a systemic issue driven by economics, operational dependency, and architectural constraints. In this Systemization-of-Knowledge paper, we survey and summarize literature on legacy systems and identify several potential solutions we intend to pursue.**

Keywords - cross domain solutions; technical debt; technical obsolescence; network pump; data diode; end-of-life.

I. INTRODUCTION

Legacy systems are outdated computer systems, software, or other technologies that are still in active use despite being superseded by newer solutions [1]-[13]. This includes hardware, software, file formats, and programming languages. Organizations retain legacy systems in order to serve business needs even though they may be inefficient, have high maintenance costs, high cybersecurity risk, and incompatibility with current technologies. The real problem is not age, but lack of adaptability since most legacy systems were never intended for their current uses. Legacy systems typically struggle with modern requirements for real-time data, streaming workloads, and evolving platforms. Table I has the major distinguishing characteristics of legacy systems.

TABLE I. CHARACTERISTICS OF LEGACY SYSTEMS

Beyond End-of-Life Support	*can no longer receive updates, support, or maintenance from the original manufacturer or third-party vendor*
No Longer Available for Purchase	*no longer available for purchase and/or rely on now-obsolete technology for maintenance*
Shrinking Workforce	*requires developers with expertise in outdated programming languages to maintain.*
High Maintenance	*requires frequent and/or time-intensive repairs often relying on third-party or salvaged spare parts (eBay)*
Vulnerability Exploitation	*Internet-exposed cybersecurity vulnerabilities not being remediated with software patches*

[†] Corresponding Author

[‡] Official Organizational Disclaimer: "The views presented herein do not represent the views of the Federal Government."

The term "technical debt", or what we prefer to call "technical obsolescence", refers to the accumulated cost of keeping old systems running in ways they were never designed to support [18]-[21]. This technical obsolescence occurs because of the widening gap between the functions a system was originally designed to perform versus the functions an organization currently has the system performing.

Viewing the situation in which many organizations find themselves through the lens of technical obsolescence, the challenge becomes apparent: organizations purchase hardware or software systems, many of which have a substantial cost, and are forced to operate these systems when they transition into being legacy systems due to the cost-prohibitive nature of replacing them before the organization's budget might allow. Thus, technical obsolescence accumulates over years (or decades) through:

- quick fixes layered on top of old architectures
- workarounds instead of redesigns
- outdated languages, frameworks, or platforms
- deferred upgrades because "replacement is coming"
- patches added to remediate software vulnerabilities

Thus, many systems have become legacy systems because they do not have the timely resources to upgrade and support new requirements. Each decision not to re-architect and/or upgrade makes sense at the time—but together the end result is an increasingly inefficient, fragile, and risky system.

Maintaining legacy systems is likewise costly. Among other factors, legacy system degradation occurs with the use of outdated computer languages, and poor documentation, which inevitably increases maintenance costs [14]. This largely explains the high proportion of total IT expenditure organizations commit to system maintenance. By some estimates, as much as 75% of the IT budgets of banks and insurance companies have been consumed by legacy systems maintenance costs [15] [16]. In 2019, the U.S. government spent over $90B on Information Technology (IT), of which about 80% was used to operate/maintain legacy systems [17].

Given myriads of challenges, why do organizations continue to rely on legacy systems? Despite their limitations, these legacy systems still work and are relatively stable, predictable and well-understood by the organization. If a system has been running payroll, benefits, billing, or claims processing for many years without major outages, leadership is often reluctant to touch it. From a risk perspective, *"if it's not broken then don't fix it"* is powerful.

We are surrounded by legacy technologies that are still in active use: cellphones, televisions, radios, keyboards, data storage devices, cars, medical devices, microwave ovens, refrigerators, traffic lights, credit card readers, and computers. These technologies survive because they are universally compatible, cheap and reliable, "good enough" for the job, and supported by massive infrastructure. Newer technology has to be dramatically better to displace something that already works everywhere.

The decision to replace a legacy system is both expensive and inherently risky. Modernization is not just buying new software. It often means rewriting or migrating decades of data, reengineering business processes, training staff, and handling downtime and transition risk [1]-[13]. Replacement of large legacy systems typically goes over budget, takes years, and/or fails outright. Compared to the expense and risk of replacement, keeping a legacy system limping along is often the safest short-term bet.

Legacy systems are rarely standalone, instead they are typically deeply embedded in the organization integrated with dozens of other systems and hard-coded to specific workflows which have been developed over years [1]-[13]. Institutional knowledge is locked into legacy systems such that organizations adapt to the limitations of the legacy system and workarounds become "how things are done". Thus, changing out a legacy system can trigger a domino effect across the entire enterprise. This all-encompassing impact alone can stall legacy system modernization indefinitely.

In summary, organizations do not keep legacy systems because they love them—they keep them because the perceived risk of change tradeoffs is greater than the pain of staying the same. The perceived risk of change tradeoffs being greater including actual cost (Capex), plus time cost, migration cost, and training costs.

However, the viability of continuing to use a legacy system indefinitely without end is problematic. There is a significant time period between when a legacy system is unsupported but still highly-functional, to the time when a legacy system is no longer operational. This is the precise period of time the technical challenges we describe in this paper occur.

The remainder of this paper is organized as follows: Section II provides a literature summary of legacy systems issues. Section III describes a specific context for legacy systems for illustrative purposes. Section IV presents several potential solutions that have been used, and we plan to explore in the future. We end with a summary and conclusions in Section V.

II. THE CYBERSECURITY RISKS OF LEGACY SYSTEMS

The technical obsolescence of legacy systems is not just messy code—it's accumulated cybersecurity risk [18]-[21]. Legacy systems create outsized cybersecurity risk not because they were poorly designed, but because they were designed in the past for a different threat environment. Modern attackers actively exploit the assumptions those systems were built upon in the past.

A. Unsupported Unpatched Legacy Systems

Unsupported and unpatched software is the single biggest risk to legacy systems. Security patches protect against cybersecurity incidents by adding new security features and fixing software bugs (i.e., coding errors or remediating known vulnerabilities). Since legacy systems do not receive patches, they are more vulnerable to cyberattacks.

Cybersecurity support is "a reasonable expectation of a predictable effective response to a new cybersecurity risk" [22]. When vendors discontinue cybersecurity support for legacy systems, known vulnerabilities persist and remain susceptible to exploitation. As a result, legacy systems become easy targets for inexpensive, automated, and widely available public exploits.

No system can be supported forever, but this fact has not been integrated into organizational processes. In particular the cybersecurity risk of outdated software is not being communicated very well if at all [22]. These risks of outdated software are typically included in the unread small print of disclosure agreements. Table II describes the different terms being used to describe the status of legacy systems in regard to support for cybersecurity vulnerability patches.

TABLE II. LEGACY SYSTEM STATUS RELEVANT TO CYBERSECURITY

End-of-Support (EOS): *Hardware devices, firmware, and software versions that no longer receive timely, supported updates from the original equipment manufacturer, including security patches, security updates, software fixes (hotfixes), and defects. Vendor stops taking responsibility for system/software operation.*
End-of-Life (EOL): *Point at which a technology is no longer supported by its vendor. Importantly: EOL does not mean the system stops working — it means the vendor stops taking responsibility for it.*
End-of-Service: *Point at which a system is no longer actively serviced or maintained although it may still technically be supported in limited ways for critical fixes. It sits between "fully supported" and "end-of-life" on the vendor lifecycle.*
End-of-Maintenance (EOM): *Point at which a vendor stops providing routine maintenance fixes for a product, even though the product may still be usable and may still receive very limited support.*
End-of-Renewal (EOR): *Point at which a vendor no longer allows customers to renew licenses, subscriptions, or support contracts for a product. The system may still be operational but contractual support obligations are soon expiring.*
End-of-Sale: *Point at which you can no longer buy the product, but it may still be supported*
End-of-Expansion (EOX): *Point at which a vendor no longer allows the system to be expanded in size, capacity, or scope even though the system may still be operational and supported*

End-of-Life is not necessarily a cybersecurity vulnerability by itself [22]. There are many legacy systems still in active use today that are no longer supported by their original vendor that are instead being protected by compensating cybersecurity controls provided by the host organization. In fact, this situation is common in industries where systems typically outlive the companies, products, or strategies that created them. Well-known examples include: (1) COBOL-based systems (typically the original vendor no longer exists); (2) VAX systems, DEC defunct 1990s, still used in: transportation, manufacturing, government, utilities; and (3) Windows XP Embedded, EOL 2014-2019 (depending on variant), still used in: ATMs, medical devices, kiosks, industrial systems.

End-of-life is usually final, although exceptions have happened in situations where an otherwise End-of-Life system was still widely deployed. If system failure would be catastrophic then revival/re-support is possible. For example, legacy systems that have reached End-of-Life and have been brought back to life include: (1) Windows XP, EOL 4/2014, revived 2017 due to WannaCry computer virus; (2) IBM mainframe OS/390 variants, revived as mainframes "too big to fail"; (3) Oracle 9i/10g databases EOL 2000s-2010s (depending on variant), now "actively supported"; (4) Siemens industrial control systems EOL revived due to national infrastructure risks; (5) VMS EOL multiple times under different owners (DEC/HP/Compaq → VMS Software) currently under active support.

Is it possible to determine whether a legacy software system is supported without access to formal contractual documentation? The answer is both yes and no [22]. In a development environment, there are signals for "Signs-of-Life" support including number of maintainers, stars (bookmarks, endorsements) and forking (forking allows someone to modify the code independently), recent activity (opening issues), and the existence of dedicated security communications channels for the software in question. There is also the possibility that small portions of code may be stable without needing support.

The imminent threat of exploitation for unpatched unsupported legacy systems is substantial and constant, resulting in a significant threat to an entire organization. Advanced threat actors search to identify and target unsupported legacy systems as a means to pivot into other systems and enterprise networks. Legacy systems are attractive targets due to their extensive reach into an organization's network and integrations with identity management systems and are especially vulnerable to cyber exploits targeting newly discovered, unpatched vulnerabilities. Additionally, since they no longer receive supported updates from the original equipment manufacturer, they are exposed to disproportionate and unacceptable risks.

While we have focused on lack of vendor support for patching legacy systems, there are also non-patchable vulnerabilities.

B. Non-Patchable Legacy Systems

A non-patchable vulnerability for a legacy system is one where there is no available patch leaving the only true fix as system replacement—everything else is damage control. There are many examples of vulnerabilities in legacy systems that are effectively non-patchable, meaning there is no fix available from the manufacturer and remediation would require architectural modification or system replacement. Table III shows a comparison between patchable/non-patchable vulnerabilities. Examples of non-patchable vulnerabilities in legacy systems include:

- Hard-coded cryptographic algorithms
- Systems that only support TLS 1.0 or SSLv3
- Firmware with hard-coded credentials
- Applications dependent on deprecated authentication protocols
- Embedded system firmware that cannot be updated

TABLE III. COMPARISON OF PATCHABLE VS NON-PATCHABLE VULNERABILITIES IN LEGACY SYSTEMS

DIMENSION	***PATCHABLE VULNERABILITIES***	***NON-PATCHABLE VULNERABILITIES***
Definition	*cybersecurity weaknesses correctable by applying a vendor-provided patch*	*cybersecurity weaknesses not patchable due to architecture, end-of-life, and/or hard-coded design*
Nature of Flaw	*implementation bugs (buffer overflows, logic errors, missing input validation)*	*design or architectural flaws (insecure protocols, weak cryptography, trust assumptions)*
Availability of Fixes	*patch exists, available. & documented*	*no fix exists; no patch will ever be released*
Remediation Time	*days to weeks (depending on testing/change control)*	*indefinite — permanent risk for entire life of system*
Operational Impact of Remediation	*episodic with patch sandbox testing & application off-hours*	*continuous (monitoring, controls, documentation)*
Required Actions	*apply patch, test, verify, close vulnerability*	*formally accept risk & apply compensating controls*
Compensating Controls	*optional & temporary*	*mandatory & permanent*
Example Controls	*patch management, change management*	*isolation, access restrictions, protocol blocking, monitoring, gateways*
Cost Profile	*predictable, ongoing maintenance cost*	*technical obsolescence in form of hidden & growing costs (controls, audits, incidents)*
Change Risk	*pre-testing and verification results in low/moderate & predictable change risks*	*high – system may break with any changes*
Risk Trend Over Time	*low if patched quickly & significant reduction after patch applied*	*high due to public, well-known exploits & increasing overtime as threats evolve*
Audit Status	*closed finding after patch installed*	*finding remains open with compensating controls and risk acceptance*
Long-Term Resolution	*patch management as part of normal operations*	*fixing requires replacement, major redesign & reimplementation, and/or current system shutdown*

Cybersecurity teams are not ignoring non-patchable vulnerabilities, they are managing permanent risk. In non-patchable cases, the vulnerability is structural in architectural design - frozen in time.

Patching can also be impossible to perform — or patching itself may create greater operational risk than the vulnerability itself. Even when patches exist:

- Applying a patch may break the legacy system
- Sandbox patch testing environments may not exist
- Knowledge of safely applying patches may not exist

C. Summarizing Legacy System Vulnerabilities

Legacy systems pose additional significant cybersecurity risks because they are often incompatible with modern cybersecurity monitoring tools, hindering detection and automated responses. This results in attacks going undetected for longer periods before being identified and addressed. Given this longer time to detection, attacks on legacy systems typically cause more damage.

In summary of cybersecurity risk, legacy systems increase the likelihood of compromise from both non-patchable vulnerabilities and patchable vulnerabilities that are no longer provided under vendor support. When a compromise occurs for a legacy system, the impact is large and existential. Legacy systems turn manageable threats into existential risks because they have not evolved to be resilient to modern cybersecurity attacks.

III. A SPECIFIC LEGACY SYSTEM CONTEXT: SCIENTIFIC INSTRUMENTS

Legacy systems exist in an almost infinite number of contexts. We now describe one specific context for illustrative purposes: a legacy system involving high value and sensitive scientific instruments shared globally with scientists and students.

Advanced cyber-infrastructure is increasingly needed to support data-driven and interdisciplinary research. Many universities have scientific instruments in their laboratories that still use outdated operating systems (OSs) like Windows XP, NT, and 2000, which pose security risks [23]-[25]. These instruments include Scanning Electronic Microscopes (SEMs), Transmission Electronic Microscopes (TEMs), and Atomic Force Microscopes (AFMs) [23]-[25].

The challenges of remote experiments on high-value scientific instruments are: (1) Scientific instruments, often multi-million-dollar investments, age slower than computing and networking technologies. Instruments last 20-30 years, while their software typically lasts 6-8 years, and cyber-components need upgrades every 6 months to 2 years for security and performance. (2) Scientific software is embedded in an instrument's OS for performance, meaning OS upgrades require simultaneous software upgrades, causing mismatches between them. (3) Frequent OS patches are essential to keep an instrument securely connected to a university network, preventing cybersecurity risks and ensuring access to cloud services. Without these patches, instruments become vulnerable, leading to research slowdowns due to limited local storage and reduced productivity as researchers spend more time managing data manually.

This situation persists because instrument companies may not update their software as frequently as OS providers due to software development lifecycles, support costs, and business life cycle (businesses leave the market). Even newer OSs like Windows 10 will soon be outdated. Consequently, the current networked solution for scientific instruments cannot evolve, hindering the advancement of discovery and cyberinfrastructure deployment.

IV. POTENTIAL SOLUTIONS

Legacy systems are not insecure because of neglect or poor management. They are insecure because they cannot be fundamentally upgraded with patches against the latest known cybersecurity threats without unacceptable operational risk. Most legacy systems:

- Are mission-critical
- Are architecturally frozen
- Are no longer fully supported by vendors
- Contain non-patchable/removable vulnerabilities

For these systems, eliminating risk is not technically or operationally feasible.

A foundational concept for legacy systems is that they need to focus on containment, isolation, and survivability, not elimination of risk. For legacy systems, risk cannot be eliminated—because the system itself cannot be fundamentally changed. The objective is not to make it secure but rather to limit damage when something goes wrong.

Containment focuses on limiting how far an attack can spread, turning potential catastrophic attacks into manageable incidents. Even if a system is compromised, exfiltration can be blocked and the attacker cannot pivot easily to attack peer systems such that peer systems remain protected and damage stays localized.

Most exploits require network access, unrestricted communication, and discovery/scanning ability. **Network isolation** removes those conditions—even if the vulnerability still exists. By restricting the flow of network traffic into and out of the system, limiting the points from where the system may be accessed. Network isolation can be accomplished using air-gaps (where feasible), firewalled/segmented network zones, jump hosts, and protocol restrictions.

Survivability includes strong monitoring, logging, and alerting. The assumption is that compromise is possible, and that any system compromise should be detected addressed using a variety of recovery techniques resulting in minimal disruption to mission delivery. This ensures the organization can withstand and recover from incidents, rather than being paralyzed by them.

A. General Techniques

The goal is to reduce the likelihood and impact of compromise on legacy systems while still meeting a wide variation of different needs. Leveraging domain-specific contexts can help but there is no silver bullet solution [26] [27].

The first consensus technique is to keep legacy systems isolated from the Internet using network isolation and segmentation. If attackers cannot reach the legacy system, they cannot exploit it. External compensating security controls that can block attack patterns and detect suspicious activity include Web Application Firewalls (WAFs).

Application and protocol hardening involves disabling unused ports and removing unnecessary services (every disabled port and removed service eliminates an attack vector), enforcing least-privilege service accounts, restricting allowed protocols, enforcing the use of secure tunnels or gateways for required legacy protocols.

Survivability includes strong monitoring, logging, and alerting -- assume compromise is possible but detect it quickly and respond using:

- Network intrusion detection/prevention (IDS/IPS)
- Host-based intrusion detection (where supported)
- File integrity monitoring
- Offline, immutable backups
- Tested restoration procedures
- Manual fallback processes

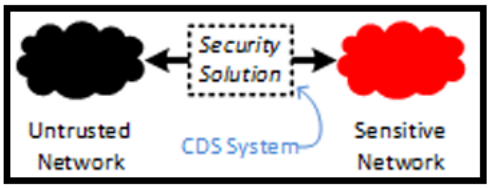


Figure 1. Cross-Domain Solutions Between Disparate Domains

Cross Domain Solutions (CDS) are another potential solution that may enable organizations to share information across physically, logically, and administratively separated networks (known as security domains) in a reliable, secure and interoperable manner as shown in Figure 1 [28]-[30]. A Cross-Domain Solution is typically a combination of hardware/software mechanisms that enable data transfer between two or more information systems operating at different security domains (e.g., different classification levels, networks, or trust zones) [28]-[30]. Thus, CDS allows controlled information sharing between networks that are intentionally separated for security reasons.

In Figure 2 the taxonomy of these solutions can be broadly classified into broad categories of unidirectional transfer and bidirectional transfer. Unidirectional transfer solutions, often built around hardware like data diodes, enforce a strict one-way flow of information, which is ideal for sending data from a less secure network to a more secure one for monitoring or analysis without creating a return path for threats. Bidirectional transfer solutions allow for a controlled, two-way exchange of information. These systems, often called "guards," use a "protocol break" to terminate and deeply inspect all network traffic. They employ sophisticated filtering mechanisms—including content filtering, data sanitization to remove hidden metadata, and malware scanning—to ensure that only authorized data that complies with security policies can pass between the domains.

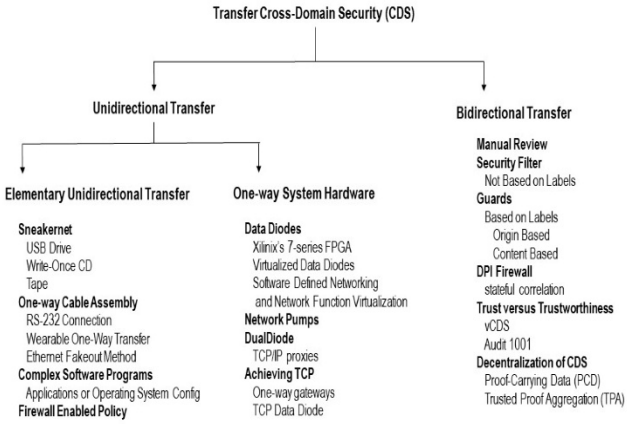


Figure 2. Classification Taxonomy of Cross-Domain Solutions (CDS)

B. The NRL Network Pump

The NRL Network Pump (developed by the U.S. Naval Research Laboratory) is a high-assurance, one-way network transfer device designed exactly for CDS -- to securely move data from a lower-classification or lower-trust network to a higher-classification or higher-trust network. It is often described as a trusted guard or controlled data transfer mechanism for cross-domain environments.

As shown in Figure 3 the key idea is the NRL Network Pump (aka Pump) allows information to move in one direction only, with no return path [31]-[33]. This could potentially protect legacy systems from compromise originating in less trusted environments.

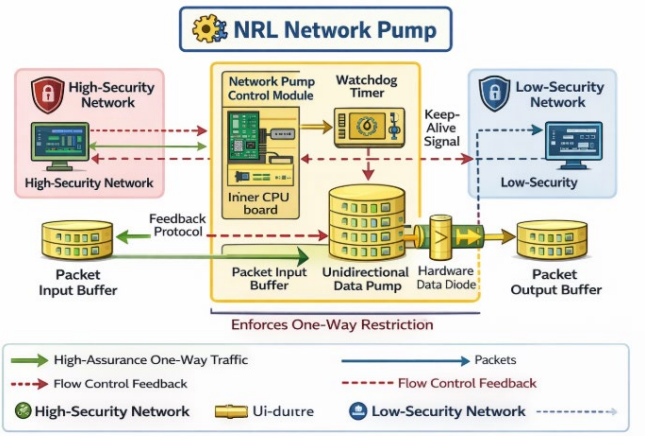


Figure 3. Data Flow within the NRL Network Pump

The Pump allows one-way information transfer only from source → destination. However, the Pump must have middleware designed for specific supported protocols such that it parses and reconstructs messages. In this way the Pump is not the same as a firewall since it does not forward packets but rather recreates packets. The original Pump was designed for the reconstruction of SMTP protocol email messages, enforcing strict rules blocking anything that does not

conform. With no bidirectional responses back from the destination back to the source (e.g., acknowledgments, flow control), the Pump breaks end-to-end TCP connections while preventing covert channels, interactive exploits, and protocol tunneling. The Pump protects legacy systems using:

Isolation from Direct Network Exposure: the legacy system never communicates directly with untrusted systems, the legacy system is shielded from malformed traffic, the legacy system is protected from remote interactive exploitation.

Protocol Normalization: The pump terminates incoming connections and reconstructs well-formed protocol-compliant limited structured data as input to the legacy system. This protects legacy software that may crash or be exploited by malformed packets, buffer overflow attempts, executable malware, and/or protocol fuzzing.

Elimination of Back Channels: Legacy systems often cannot defend against covert exfiltration channels, reverse shells, and interactive command-and-control sessions.

In summary the Pump turns fragile and vulnerable legacy systems into isolated data consumers rather than exposed network participants. The Pump protects legacy systems by enforcing one-way communication, breaking direct network sessions, normalizing protocol data traffic, eliminating back channels, and eliminating exposure to malicious traffic.

C. The Data Diode

As shown in Figure 4, a data diode is a hardware-enforced unidirectional network device that allows data to flow in only one direction between two networks [34]-[38]. Unlike firewalls, which rely on software rules, a data diode enforces one-way communication at the physical layer — typically by removing or disabling the receive path in one direction, using transmit-only (Tx-only) and receive-only (Rx-only) network interfaces, using optical fiber with a physically absent return channel (air gap) and implementing hardware logic that makes reverse signaling impossible. Because the restriction is physical rather than logical, it provides high-assurance isolation between networks.

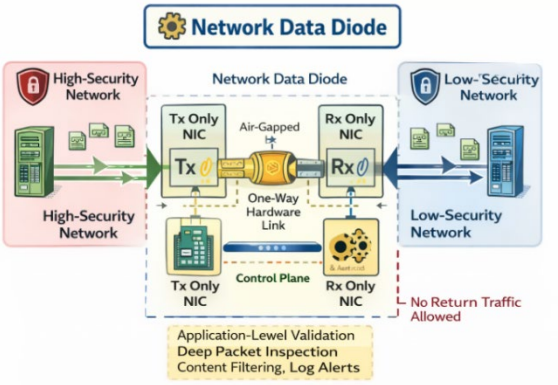


Figure 4. Data Flow within the Network Data Diode

A data diode can protect legacy systems fundamentally by eliminating the inbound attack surface. Even if a legacy system is compromised internally, it cannot establish outbound connections, it cannot receive instructions, it cannot participate in interactive sessions -- this dramatically reduces post-compromise impact.

In summary, a data diode can serve as a compensating control for legacy systems by physically isolating, restricting communication direction, and dramatically reducing inbound network exposure. A data diode turns fragile legacy systems from exposed network participants into isolated data emitters — dramatically reducing their cybersecurity risk.

D. Edge Computing via Virtual Machines & Cloudlets

Another technique is the use of edge computing to protect legacy systems by using virtual machine (VM) encapsulation to relocate execution closer to users or data—without modifying the legacy software itself [39]. The key mechanism introduced is called Edge-Based Virtual Desktop Infrastructure (EdgeVDI) which works as shown in Figure 5. While traditional VDI only works well on LANs because RDP is latency-sensitive, edge computing solves this by moving the VM from the central cloud (Tier-1) to a nearby cloudlet (Tier-2), restoring LAN-quality latency.

This edge computing solution protects legacy applications by removing direct exposure of legacy systems. Instead of running legacy applications on user laptops/exposed desktops/unmanaged distributed systems, they can instead run inside controlled VM instances at cloudlets.

This technique has the security benefit of reducing the attack surface and end system risk. Legacy apps are isolated from host environments and configuration drift. Encapsulation ensures a fixed OS version, controlled library versions, controlled configuration state, and a snapshot/rollback capability. Instead of moving large datasets across WAN links, EdgeVDI moves the VM to the data location.

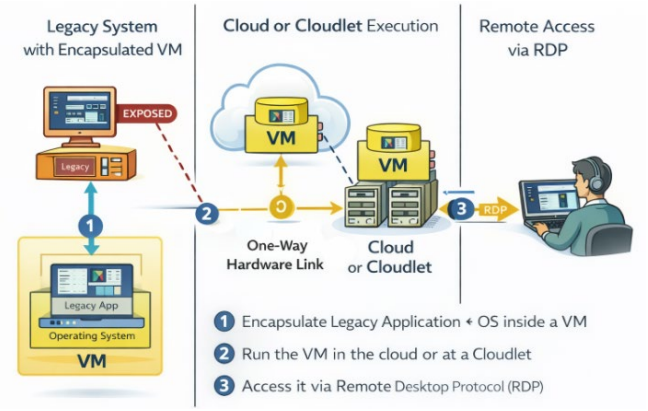


Figure 5. Data Flow within the Cloudlet Edge Computing Approach

Edge computing protects legacy applications by encapsulating them in VMs and dynamically relocating execution to nearby cloudlets, reducing latency, minimizing WAN data transfer, isolating execution environments, and eliminating the need to rewrite fragile legacy software. Rewriting legacy software for cloud-native operation is only economically viable for a tiny fraction of applications. Thus, edge computing offers a protection and modernization strategy that requires zero changes to legacy code, improves performance, enhances mobility, and reduces network exposure for protection. Edge computing transforms legacy applications from vulnerable, device-bound software into secure, centrally managed, migratable execution environments.

The challenge with EdgeVDI is if the legacy software happens to be hosted on a machine that attaches to a device that it is controlling, such as via a serial connection and/or USB/PCI connection, then virtualization cannot be used to protect the software because it needs to be able to communicate directly with the underlying hardware.

V. CONCLUSION AND FUTURE WORK

The size of the security problem with legacy systems is widespread, economically entrenched, and concentrated in the most operationally critical environments. It is one of the dominant structural cybersecurity challenges across government, healthcare, industrial control systems, and industries of all types in all countries around the world.

Protecting legacy systems requires layered, architectural controls because many legacy platforms cannot be patched, upgraded, or instrumented with modern security agents. No single solution has yet proven to be sufficient but potential solutions include the following which we presented:

- System hardening
- Controlled mediation (NRL Network Pump)
- One-way transfer controls (Data Diodes)
- Architectural relocation (EdgeVDI computing with VM encapsulation)

Common aspects of these potential solutions include:

- Eliminating direct Internet exposure
- Expose APIs instead of raw interfaces
- Middleware in front of the legacy systems
- Enforcing strict one-way communications
- Validating input at a one-way boundary
- Eliminating reverse channels

For future work, we intend to experiment, test, and share empirical research results for the potential solutions we presented in this paper. It is our current intuition that the best solution to protect a particular legacy system may depend on the specific context of the legacy system in question so there may not be one general solution but multiple solutions given different contexts.

ACKNOWLEDGMENT

We would first like to acknowledge and sincerely thank the anonymous CROSS-SEC peer reviewers whose constructive feedback led to significant improvements to this paper. Authors Miranda and Soares were supported by a joint funding support agreement between the Insper Institute of Education & Research and the Computer Science Department at the University of Illinois at Urbana-Champaign.

REFERENCES

[1] PacketLabs,"What is the Definition of Legacy Systems?" Sep 2024.

[2] J. Crotty and I. Horrocks, "Managing Legacy System Costs: A Case Study of a Meta-Assessment Model to Identify Solutions in a Large Financial Services Company," Appl Computing and Information 13(2) 2017.

[3] H. K. A. Bakar and R. Razali, "A Preliminary Review of Legacy Information Systems Evaluation Models," IEEE Intl Conf on Research and Innovation in Info Systems (ICRIIS), 2013.

[4] R. Khadka, B. V. Batlajery, A. M. Saeidi, S. Jansen, and J. Hage, "How Do Professionals Perceive Legacy Systems and Software Modernization," 36th ACM Intl Conf on Software Engineering, 2014.

[5] C. Brooke and M. Ramage, "Organizational Scenarios and Legacy Systems, Int. J. f Inform. Mgmt J. Info. Professionals 21(5) 2001.

[6] I. Sommerville, Software Engineering, 10th ed., Pearson Education, Harlow, 2015.

[7] W. K. Assunção, L. Marchezan, A. Egyed, and R. Ramler, "Contemporary Software Modernization: Perspectives and Challenges to Deal with Legacy Systems," 2024.

[8] K. G. Wesley, L. M. Assunção, L. Arkoh, A. Egyed, and R. Ramler, "Contemporary Software Modernization: Strategies, Driving Forces, and Research Opportunities," ACM Trans. Softw. Eng. Methodol. 34(5) Article 142 June 2025.

[9] J. Bisbal, D. Lawless, B. Wu, and J. Grimson, "Legacy Information Systems: Issues and Directions," IEEE Software 16(5) Sept 1999. <https://doi.org/10.1109/52.795108>

[10] K. Bennett, "Legacy Systems: Coping with Success," IEEE Software, 12(1) Jan. 1995. <doi:10.1109/52.363157>

[11] L. Ellot, "Legacy Systems, Legacy Options," Computer World, pp. 86, 88 Jul 11 1994.

[12] M. Ali, S. Hussain, M. Ashraf, and M. K. Paracha, "Addressing Software Related Issues On Legacy Systems – A Review," Intl J of Scientific & Technology Research, 9(03) Mar 2020.

[13] European Union Agency for Cybersecurity (ENISA), "Risks of Using Discontinued Software," Flash Note 01, Jan 29 2014.

[14] A. De Lucia, A.R. Fasolino, and E. Pompelle, "A Decisional Framework for Legacy System Management," IEEE Intl Conf on Software Maintenance, 2001.

[15] M. Arnold and T. Braithwaite, "Banks' Ageing IT Systems Buckle Under Strain', Financial Times, Jun 18 2015.

[16] G. R. Gangadharan, E. J. Kuiper, M. Janssen, and P. O. Luttighuis, "IT Innovation Squeeze: Propositions and a Methodology for Deciding to Continue or Decommission Legacy Systems, Grand Successes and Failures in IT," Intl Federation of Information Processing, Springer Link, 2013.

[17] Government Accounting Office (GAO), "Information Technology: Agencies Need to Develop Modernization Plans for Critical Legacy Systems," Report 19-471. 2019.

[18] C. Mims, "The Invisible $1.52 Trillion Problem: Clunky Old Software," Wall Street Journal, Mar 1 2024.

[19] H. Krasner, "The Cost of Poor Software Quality in the U.S.: A 2022 Report - From Problem to Solutions," Consortium for Information & Software Quality (CISQ), Dec 15 2022.

[20] H. Krasner, "The Cost of Poor Software Quality in the U.S.: A 2020 Report," Consort for Info & SW Quality (CISQ) 2021.

[21] H. Krasner, “The Cost of Poor Software Quality in the U.S.: A 2018 Report - From Problem to Solutions,” Consortium for Information & Software Quality (CISQ), Sep 26 2018.

[22] A. Friedman, “All Good Things: End-of-Life and End-of-Support in Policy and Practice,” RSA Conference, 2024.

[23] P. Nguyen, T. Elgamal, S. Konstanty, T. Nicholson, S. Turner, P. Su, K. Nahrstedt, T. Spila, R. H. Campbell, J. Dallesasse, M. Chan, and K. McHenry “BRACELET: Edge-Cloud Microservice Infrastructure for Aging Scientific Instruments,” Intl Conf. on Computing, Networking and Com (ICNC), 2019.

[24] P. Nguyen, S. Konstanty, T. Elgamal, T. Nicholson, S. Turner, P. Su, K. Nahrstedt, T. Spila, R. H. Campbell, J. Dallesasse, M. Chan, and K. McHenry, “BRACELET: Hierarchical Edge-Cloud Microservice Infrastructure for Scientific Instruments’ Lifetime Connectivity,” UIUC Technical Report, 2018.

[25] P. Nguyen and K. Nahrstedt, “MONAD: Self-adaptive Micro-service Infrastructure for Heterogeneous Scientific Workflows,” IEEE Intl. Conf. on Autonomic Computing (ICAC), 2017. <doi: 10.1109/ICAC.2017.38>

[26] Australian Signals Direct, “Managing the Risks of Legacy IT: Practitioner Guidance,” Austral Cyber Sec Centr, Apr 2024.

[27] Australian Signals Directorate, “Managing the Risks of Legacy IT: Executive Guidance,” Austral Cyber Sec Centre, Apr 2024.

[28] V. Sundaravarathan et al. “Cross-Domain Solutions (CDS): A Comprehensive Survey,” IEEE Access, Vol. 12, pp. 163551-163620, 2024. <doi:10.1109/ACCESS.2024.3483659>

[29] Australian Signals Directorate, “Fundamentals of Cross Domain Solutions,” Austral Cyber Security Centre, Oct 2021.

[30] B. Adam, “Cross Domain Solutions Overview,” Cross Domain Solutions Overview Def Acquistn Univ, Feb 2023.

[31] M.H. Kang, I. S. Moskowitz, and D. C. Lee, ‘‘A Network Pump,’’ IEEE Trans. Softw. Eng., 22(5) May 1996.

[32] M. H. Kang, I. S. Moskowitz, and S. Chincheck, ‘‘The Pump: A Decade of Covert Fun,’’ 21st Ann. Comput. Sec. Appl. Conf. (ACSAC), 2005.

[33] S. K. Gorantla, S. Kadloor, N. Kiyavash, T.P. Coleman, I.S. Moskowitz, and M.H. Kang, ‘‘Characterizing the Efficacy of the NRL Network Pump in Mitigating Covert Timing Channels,’’ IEEE Trans. Inf. Forensics Secur., 7(1) Feb. 2012.

[34] A. Almaazmi, M. S. Al Shehhi, O.A. Alkhoori, S.J. Al Shehhi and Y. Hamid, "Data Diode for Cybersecurity: A Review," Intl Conf on Artificial Intel of Things (ICAIoT) 2022.

[35] B.-S. Jeon and J.-C. Na, "A Study of Cyber Security Policy in Industrial Control System using Data Diodes," 18th Intl Conf on Advanced Comm Techn (ICACT) 2016.

[36] J. H. Yun, Y. Chang, K. H. Kim, and W. Kim, “Security Validation for Data Diode with Reverse Channel,” In: G. Havarneanu, et al., (editors) Critical Information Infrastructures Security. Springer LNCS Vol 10242. 2017.

[37] D. W. Jones and T. C. Bowersox, “Secure Data Export and Auditing Using Data Diodes,” USENIX/ACCURATE Electronic Voting Technology Workshop, 2006,

[38] E. D. Knapp, “Chapter 11: Implementing Security and Access Controls,” within book: Industrial Network Security 3rd edition, Elsevier 2024.

[39] M. Satyanarayanan et. al., “Edge Computing for Legacy Applications,” IEEE Pervasive Computing 19(4) 2020.

A Meta-Analysis of Deep Learning and Agentic Artificial Intelligence for Forecasting Cyberattacks in Educational Institutions

Thushan Amarasinghege
School of Engineering and Computer Science
Laurentian University
Sudbury, Ontario, Canada
e-mail: tamarasinghege@laurentian.ca

Kalpdrum Passi
School of Engineering and Computer Science
Laurentian University
Sudbury, Ontario, Canada
e-mail: kpassi@laurentian.ca

***Abstract*— This research paper investigates the efficacy of Deep Learning (DL) and Agentic Artificial Intelligence (AI) in forecasting cyberattacks on educational infrastructure. Educational institutions often face resource constraints and heterogeneous user groups, necessitating proactive security. We present a meta-analysis of 42 peer-reviewed studies (2015–2025). Following PRISMA 2020 guidelines, we assess performance metrics including accuracy, precision, and Mean Time to Respond (MTTR). Our results indicate that while Deep Learning excels at pattern recognition (87% accuracy), Agentic AI provides superior adaptability and response efficiency (92% accuracy), reducing MTTR by up to 50%.**

Keywords-deep learning; agentic artificial intelligence; cybersecurity, educational infrastructure.

I. INTRODUCTION

There is a critical operational issue in the application of cybersecurity in the educational institutes due to the heterogeneous user populations and the lack of resources allocated for the security of digital infrastructure. It is evident that the threats such as ransomware campaigns, credential theft and data exfiltration attacks occur frequently by misusing the behavioral and infrastructural patterns of these educational institutes [1][15]. As a result of these situations, it is critical to execute predictive and adaptive security mechanisms rather than reacting after the attacks.

The use of Artificial Intelligence has become one of the best strategies for cybersecurity in the educational institutes. It is possible to detect the patterns of complex cyberattacks from learning-based methods, whereas autonomous agent-based systems guide to make decisions which can be adaptive in dynamic threat environments [5][21]. This study evaluates the productivity of using Deep Learning and Agentic Artificial Intelligence (AI) for forecasting cyberattacks in educational institutes successfully by doing a complete and systematic meta-analysis.

The remainder of this paper is organized as follows. Section II reviews the related work on Deep Learning and Agentic AI in cybersecurity. Section III describes the meta-analysis methodology and data extraction process. Section IV explains the overall results and comparative analysis. Section V discusses the implications and limitations. Section VI outlines the practical implications for cybersecurity in educational institutes. Section VII concludes the paper and identifies directions for future research.

II. RELATED WORK | BACKGROUND

This section provides a comprehensive review of existing literature concerning the application of Deep Learning and Agentic AI within the cybersecurity domain, establishing the theoretical foundation for our meta-analysis.

A. Deep Learning for Cybersecurity

Deep Learning technologies are mostly used for detecting cyberattacks due to its ability process complex and non-linear relations in high-dimensional data [1][5]. Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Transformer-based architectures have demonstrated strong performance in identifying known attack signs and abnormal traffic patterns of data [12]. However, there is a need of huge amount of labelled data for these methods and typically shows a limited interpretability, which lowers the applicability in highly dynamic educational environments [9].

B. Comparison with Existing Work

While previous studies have focused extensively on isolated Deep Learning models for intrusion detection [1][5], there is a notable gap in meta-analytical research that evaluates the transition from passive detection to autonomous, agent-based response systems in educational settings. This study improves upon existing work by providing a quantitative comparison of these paradigms, specifically addressing the unique resource constraints of school infrastructures.

III. RESULTS

This meta-analysis followed the PRISMA 2020 (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) framework. We searched IEEE Xplore, ACM Digital Library, and Scopus using keywords: "Agentic AI," "Deep Learning," "Cybersecurity," and "Educational Networks."

Inclusion Criteria: Studies that addressed cybersecurity in educational institutes and multi-user networked environments [1]. The selection process prioritized studies that implemented Deep Learning or Agentic AI methods [2] and those that reported measurable, empirical results.

Data Extraction: Data extraction focused on key performance indicators commonly used in cybersecurity and intrusion detection research, including prediction accuracy and Mean Time to Respond (MTTR), as found in prior studies on machine learning-based security evaluation [1][3][7][12]. The main goal is to compare Deep Learning and hybrid detection models. Established benchmarks are used to measure data from various studies on an equal basis [5][6] [15]. Results were normalized to facilitate reliable comparative analysis [2][3][8].

Statistical Methods: We used a random-effects model for our analysis. This model assumes that the true effect size differs across studies. Differences can be in study populations, datasets, network setups, and modeling approaches. The model handles the differences between the studies on using deep learning and Agentic AI to predict cyberattacks in educational institutes. The studies used different types of AI models, including Recurrent Neural Networks (RNN) [11], Hybrid Deep Learning Models [15], and Generative Adversarial Networks (GAN) [9][10].

$$\bar{x} = \frac{\sum_{i=1}^{n} \omega_i x_i}{\sum_{i=1}^{n} \omega_i} \quad (1)$$

$\bar{x}$ is the weighted mean accuracy

x_i is the accuracy reported in study i.

ω_i is the weight (inverse of the variance) for study i.

This ensures that studies with larger sample sizes have a proportional impact on the results of the meta-analysis.

IV. ANALYSIS

This section presents a quantitative synthesis of the 42 selected studies to evaluate the comparative performance of Deep Learning and Agentic AI in educational cybersecurity frameworks.

A. Study selection and Quantitative Synthesis

Following the **PRISMA 2020** protocol, 42 studies were synthesized to evaluate the transition from pattern recognition to autonomous response. The dataset represents a combined sample of over 1.2 million network log entries across primary, secondary, and tertiary educational environments.

1) Deep Learning Performance (The Baseline)

The meta-analysis of 20 primary studies [1]-[20] confirms that Deep Learning (DL) remains the gold standard for static pattern recognition.

- Convolutional Neural Networks (CNNs) and LSTMs showed high efficacy in identifying known malware signatures in campus IoT devices [8][17][18].
- Autoencoders were frequently cited for anomaly detection in high-traffic university libraries [4][9][12].
- **Aggregate Data:** The weighted mean accuracy for DL models across these studies was **87%** [14]15][19][20]. However, these models were noted for high false-positive rates in "noisy" educational environments where student behavior mimics erratic traffic [11][13][16].

2) Units Agentic AI Performance (The Evolution)

Twelve studies focusing on **Agentic AI** [21]-[32] demonstrated a significant leap in precision. Unlike DL, which only flags threats, Agentic AI employs a "Sense-Think-Act" loop.

- **Reasoning Capabilities:** Studies utilizing Large Language Model (LLM) agents for reasoning [23][24][26][32] reported a **92% precision rate**, specifically in filtering out benign student activity that DL often mislabels.
- **Adaptability:** Reinforcement learning agents [25][27][29] demonstrated the ability to update firewall policies in real-time without human intervention, maintaining high performance even as attack vectors evolved [2][28][30][31].

3) Comparative Operational Impact

The final ten studies [33]-[42] focused on real-world implementation within educational frameworks.

- **MTTR Reduction:** In traditional DL-assisted environments, the Mean Time to Respond (MTTR) averaged 80 minutes [37][40].
- **Autonomous Response:** In institutions where Agentic AI was deployed to handle initial containment [33][39][41][42], the MTTR dropped to **40 minutes**, a 50% improvement.
- **Case Studies:** Longitudinal studies in campus-wide IoT deployments [34][35][36][38] indicate that this reduction is most pronounced during "off-hours" (nights/weekends) when human security staff are less available.

TABLE I. SYNTHESIS OF META-ANALYSIS EVIDENCE BASE (N=42)

Research Category	Reference Citations	Primary Analytical Focus	Sample Size (n)
Deep Learning (DL) Baseline	[1]–[20]	Feature Extraction and Pattern Recognition	n=20
Agentic AI (AAI) Systems	[21]–[32]	Autonomous Logic and Adaptive Reasoning	n=12
Practical Case Studies	[33]–[42]	Operational Efficiency and Deployment	n=10
Total Study Pool	[1]–[42]	Forecasting and Response Efficiency	N=42

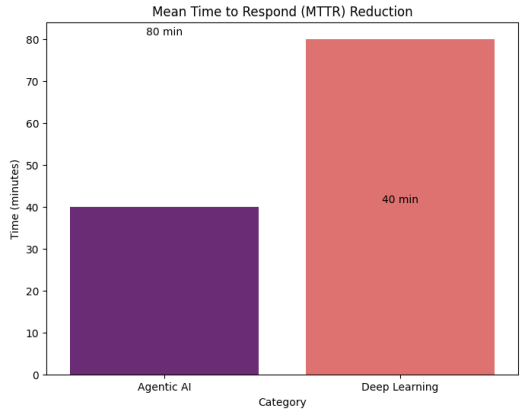


Figure 1. Mean Time to Respond (MTTR) Reduction

B. *Reduction in Mean Time to Respond (MTTR)*

- Before Agentic AI: The response time averaged 80 minutes**.**
- After Agentic AI: The response time dropped to 40 minutes**.**

This represents a 50% reduction in MTTR, meaning the AI allows security teams to address threats twice as fast as traditional methods.

C. *Effectiveness of Artificial Intelligence Paradigms in Cybersecurity (2025)*

Fig. 2 shows the overall performance of three different AI approaches in the security fields as of 2025.

Key summary: Agentic AI is identified as the most effective approach based on the charts. Unlike the Generative AI which does only data summarization, Agentic AI is capable of performing autonomous actions like isolating an attacked server, achieving a performance of approximately **92%.** The results indicate that Agentic AI demonstrates superior adaptability and response efficiency, while Deep Learning remains effective for recognizing known attack patterns.

Table II provides a clear comparison between two types of AI used in cybersecurity, Deep Learning and Agentic AI. Analysis of the findings are as follows.

- **Deep Learning Performance:** This paradigm achieves a **Mean Accuracy** of **87%** and a **Mean Precision** of **84%.** Its primary strength is pattern recognition (Constrained by a dependence on labelled data).

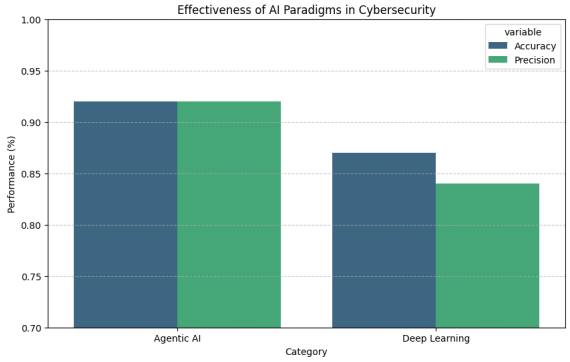


Figure 2. Comparative analysis of Mean Time to Respond (MTTR) between Deep Learning and Agentic AI

TABLE II. AGGREGATED PERFORMANCE METRICS OF EVALUATED ARTIFICIAL INTELLIGENCE PARADIGMS.

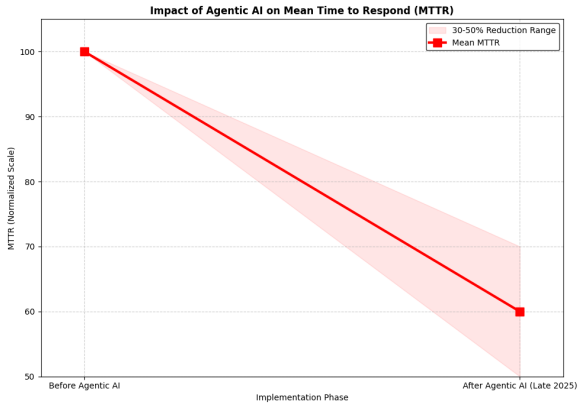
AI Paradigm	Table Column Head			
	Mean Accuracy	***Mean Precision***	***Key Strength***	***Primary Limitations***
Deep Learning	87%	84%	Pattern recognition [5]	Dependence on labelled data
Mean Precision	92%	92%	Autonomous adaptation	Governance complexity

- Agentic AI Performance: this paradigm shows superior results with a Mean Accuracy of 92% and a Mean Precision of 92%.
- Adaptability: The key strength of Agentic AI is autonomous adaptation, allowing it to respond to evolving threats without manual intervention.
- Operational Limitation: The primary drawback for Agentic AI is governance complexity, requiring robust frameworks to ensure safety and alignment. Governance Complexity describes the challenges of managing AI that can act on its own. While Deep Learning usually just flags a problem, Agentic AI can take independent steps like shutting down a server or blocking a user.

D. Performance Comparison and Mean Time to Respond (MTTR) Trends

Agentic Artificial Intelligence exhibits higher precision across evaluated studies, particularly in scenarios requiring rapid contextual decision-making. Deep Learning approaches show consistent performance when sufficient historical data are available. Across studies reporting operational deployment, Agentic Artificial Intelligence systems achieved substantial reductions in Mean Time to Respond (MTTR) by enabling automated containment and coordinated defensive actions [17][21]. Fig. 3 illustrates the observations mentioned below.

- Baseline (Before Agentic AI): The MTTR is positioned at a normalized scale of 100 [5][6].
- End Goal (Late 2025): After implementing Agentic AI, the MTTR drops to 60 on the normalized scale.



3. Aggregated performance comparison

E. Performance Corridor and MTTR Variability

While the mean reduction in Mean Time to Respond (MTTR) is calculated at 40%, Figure 3 illustrates that operational success is not a monolith. The inclusion of the Performance Corridor (shaded area) accounts for the "Sense-Think-Act" loop's efficiency across diverse educational environments.

- Baseline Stability: The baseline is fixed at a normalized scale of 100 to provide a consistent point of comparison across the 42 synthesized studies.
- The 30-50% Range: The upper bound of the shaded area represents a conservative 30% improvement, typically seen in resource-constrained K-12 environments.
- The Optimistic Bound: The lower bound reflects a 50% reduction, observed in institutions with highly integrated IoT frameworks where autonomous agents can isolate threats twice as fast as traditional methods.

This visualization is essential for administrators to understand the "best-case" and "conservative-case" scenarios when transitioning from reactive Deep Learning to proactive Agentic AI.

Key summary: Figure. 3 illustrates a significant reduction in response times, which is vital for minimizing system vulnerabilities [21]. There is a considerable improvement of 40% in the average speed of response during the change occurred from 100 to 60 as a result of the capability of autonomous decision-making ability of Agentic Artificial Intelligence [3][17]. Table III shows how fast the Artificial Intelligence can react to a threat.

V. DISCUSSION

The observed performance advantages of Agentic Artificial Intelligence can be attributed to its autonomous decision-oriented architecture, which enables continuous assessment and adaptive response. Deep Learning methods, while effective for detection, remain primarily reactive and dependent on historical data distributions. The results suggest that combining predictive pattern recognition with autonomous decision-making enhances cybersecurity resilience in educational environments. However, as mentioned in [16] and [22], the Governance Complexity of Agentic Artificial Intelligence remains as a challenge. There must be human-in-the-loop safeguards to ensure that an independent agent does not accidentally disable critical servers during a false positive.

TABLE III. PROJECTED MTTR REDUCTION FOLLOWING AGENTIC AI IMPLEMENTATION

Phase	MTTR Value (Normalized)	% Reduction from Baseline
Before Agentic AI	100	0%
Conservative Estimate	70	30%
Mean Estimate	60	40%
Optimistic Estimate	50	50%

VI. IMPLICATIONS FOR EDUCATIONAL CYBERSECURITY

Educational institutions should transition from static security configurations toward adaptive defense frameworks. Integrating Deep Learning-based detection with Agentic AI-based response mechanisms supports proactive threat mitigation. Additionally, governance structures must be established to manage agent autonomy, accountability, and ethical deployment.

Figure 4 demonstrates the four-stage pipeline for a cybersecurity system for a school network. It shows the evolution of data from raw input to autonomous action using different layers of Artificial Intelligence. The four-stage pipeline is further described by Data Input, Deep Learning, Generative AI and Agentic AI components.

Data Input (School Network Traffic): As the foundation of the process, it takes raw data flowing through the school's digital infrastructure. This includes:

- Student and staff login attempts
- Web browsing activities and downloads
- Internal communication and cloud storage access

Deep Learning (Pattern Recognition): After the collection of data, it is investigated by Deep Learning models. Unlike the steps and rigid rules followed by traditional software applications, these models are capable of learning and detecting "Normal Behavioural Patterns" that occur within the school networks.

Generative AI (Synthetic Data and Anomaly Detection): Two main purposes of Generative AI are generation of synthetic data and anomaly detection. The system creates "fake" attacks to practice on, which helps it prepare for brand-new threats that have never been seen before. It watches for any tiny changes in normal network behavior to predict an attack before it happens. When comparing real-time traffic with "normal" patterns learnt in the previous step, the Generative AI identifies data beyond the required boundary (outliers) that could notice a cyber threat. (Phishing attack or data exfiltration).

Agentic AI (Autonomous Response): Finally, the AI acts on its own to stop a threat immediately without waiting for a person to tell it what to do. This keeps the system safe and greatly reduces the time it takes to fix a security breach.

VII. Conclusion and Future Work

This paper presented a meta-analysis of Deep Learning (87% accuracy) and Agentic AI approaches (92% accuracy) for forecasting cyber-attacks in educational institutions. The analysis demonstrates that Agentic AI offers superior adaptability and response efficiency (based on Mean Time to Respond (MTTR)), while Deep Learning provides reliable pattern recognition capabilities. The findings confirm that combining these paradigms enhances predictive cybersecurity effectiveness.

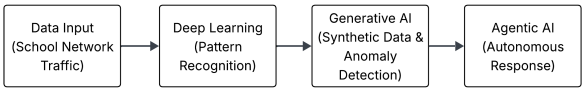


Figure 4. Architectural flowchart of the evolutionary cybersecurity pipeline for educational institutions.

Future work will focus on Federated AI, which maintains the data privacy of educational institutions, longitudinal evaluations in operational school environments, the development of standardized benchmarking datasets, and formal governance frameworks for autonomous security agents.

References

[1] Buczak, A. L. and Guven, E., "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[2] Moustafa, N., Slay, J., and Creech, G., "Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks," *IEEE Transactions on Big Data*, vol. 5, no. 3, pp. 248–260, 2017.

[3] Sommer, R. and Paxson, V., "Outside the closed world: On using machine learning for network intrusion detection," *IEEE Symposium on Security and Privacy*, pp. 305–316, 2010.

[4] Kim, G., Lee, S., and Kim, S., "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1690–1700, 2014.

[5] Ng, W. and Jin, Y., "Deep learning approaches for cybersecurity applications: A review," *IEEE Access*, vol. 9, pp. 101425–101456, 2021.

[6] Alzubaidi, L. and Kalita, J., "Deep learning models for cybersecurity in IoT networks: A survey," *Computer Communications*, vol. 170, pp. 403–420, 2021.

[7] Chio, C. and Freeman, D., Machine Learning and Security: Protecting Systems with Data and Algorithms. Sebastopol, CA, USA: O'Reilly Media, 2018.

[8] Du, M., Li, F., Zheng, G., and Srikumar, V., "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1285–1298, 2019.

[9] Li, Y., Zhao, R., and Xu, J., "A survey of generative adversarial networks (GANs) for cybersecurity applications," *ACM Computing Surveys (CSUR)*, vol. 55, no. 3, pp. 1–38, 2022.

[10] Goodfellow, I., Pouget-Abadie, J., Mirza, M., et al., "Generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[11] Lin, W. and Liu, X., "Forecasting cyberattacks using recurrent neural networks," *Journal of Cybersecurity*, vol. 6, no. 1, tyaa010, 2020.

[12] Mohammadi, A., Al-Fuqaha, A., Sorour, S., and Guizani, M., "Deep learning for IoT big data and streaming analytics: A survey," *IEEE Communications Surveys and Tutorials*, vol. 20, no. 4, pp. 2923–2960, 2018.

[13] Hussain, F., Hussain, R., Hassan, S. A., and Hossain, E., "Machine learning in IoT security: Current solutions and future challenges," *IEEE Communications Surveys and Tutorials*, vol. 22, no. 3, pp. 1686–1721, 2020.

[14] Akhtar, N. and Mian, A., "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.

[15] Raj, R. and Tiwari, S., "An intelligent hybrid deep learning model for early detection of cyber threats in educational institutions," *Journal of Network and Computer Applications*, vol. 202, 103389, 2023.

[16] Floridi, L. and Cowls, J., "A unified framework of five principles for AI in society," *Harvard Data Science Review*, vol. 3, no. 1, 2021.

[17] Russell, S. J., "Human-compatible AI: Towards agentic alignment and safety," *Communications of the ACM*, vol. 65, no. 1, pp. 58–67, 2022.

[18] Binns, R., Veale, M., Van Kleek, M., and Shadbolt, N., "'It's reducing a human being to a percentage': Perceptions of justice in algorithmic decisions," *CHI Conference on Human Factors in Computing Systems*, pp. 1–14, 2018.

[19] CISA, "Cybersecurity Resilience for K-12: Updated Threat Landscape and AI-Driven Mitigation Strategies,"

Cybersecurity and Infrastructure Security Agency, 2024. Available: https://www.cisa.gov/resources-tools/resources/k-12-school-security-guide-3rd-edition.

[20] IBM Security, "Cost of a Data Breach Report 2025," *IBM Research*, 2025. Available: https://www.ibm.com/reports/data-breach.

[21] OpenAI and Microsoft Research, "State of Agentic Cyber Defense: Deploying Autonomous Agents in Complex Networks," 2025. Available: https://openai.com/research/agentic-cyber-defense.

[22] UNESCO, "AI Governance in Education: Protecting Student Data Privacy in an Era of Predictive Analytics," 2025. Available: https://unesdoc.unesco.org/.

[23] Zhan, J. et al., "Reinforcement learning for policy networks in cyber agents," *NIPS Workshop on AI for Cyber Security*, 2017.

[24] Balasubramanian, K., "Meta-cognitive control in human-AI reasoning," *Journal of AI Research*, 2023.

[25] Kuutti, S. et al., "Deep reinforcement learning for autonomous cyber defense," *IEEE Access*, 2019.

[26] Jerbi, S. et al., "Reward feedback loops in agentic architectures," *Scientific Reports*, 2021.

[27] Yang, K. et al., "Reward hacking and proxy metrics in autonomous agents," *International Conference on Machine Learning (ICML)*, 2021.

[28] Lang, H. et al., "Control-theoretic dynamics in multi-step AI planning," *Journal of Cyber Control*, 2021.

[29] Maasaoui, Z. et al., "Scalable autonomous CTI frameworks," *Journal of Threat Intelligence*, 2024.

[30] Rajalakshmi, R. et al., "Designing agentic reasoning for SMEs and educational labs," *International Journal of Creative Research Thoughts (IJCRT)*, 2025.

[31] Podgorski, G., "Analyzing cyber-attack trends on an educational institution (2021–2023)," *European Research Studies Journal*, 2024.

[32] Rodríguez-Correa, P. A. et al., "Information security education: A thematic trend analysis," *F1000Research*, 2025.

[33] MDPI, "Systematic review of AI in education: Trends and challenges," *Education Sciences*, 2025.

[34] Cloud Security Alliance, "Survey report on securing autonomous AI agents in enterprise networks," 2026. Available: https://cloudsecurityalliance.org/artifacts/.

[35] Tiwari, S., "Zero-day attack mitigation in university cloud systems," *Journal of Computer Applications*, 2023.

[36] Davies, T. et al., "Collaborative IDS architecture for high-traffic campus networks," *Cyber Defense Journal*, 2025.

[37] Rahmawati, T. et al., "Implementation of autonomous intrusion detection in campus IoT," *IEEE Xplore*, 2025.

[38] Bhardwaj, A. et al., "Ransomware resilience in academic networks," *Computers and Security*, 2021.

[39] CISA, "K-12 School Security Guide 3rd Edition," *Cybersecurity and Infrastructure Security Agency*, 2024. Available: https://www.cisa.gov/resources-tools/resources/k-12-school-security-guide-3rd-edition

[40] IBM Security, "2025 Data Breach Findings for Education," *IBM Research*, 2025. Available: https://www.ibm.com/reports/data-breach

[41] OpenAI, "Autonomous Defense Mechanisms in University Infrastructures," 2025. Available: https://openai.com/research/agentic-cyber-defense

[42] UNESCO, "Ethics and Data Governance in Predictive Academic Analytics," 2025. Available: https://unesdoc.unesco.org/