An Analysis Framework for Steganographic Network Data in Industrial Control Systems

Tom Neubert, Bjarne Peuker, Eric Schueler
Henning Ullrich, Laura Buxhoidt, Claus Vielhauer
Brandenburg University of Applied Sciences
Brandenburg, Germany
e-mail: surname.lastname@th-brandenburg.de

***Abstract*—This paper presents a novel analysis framework for steganographic embedding methods in Industrial Control Systems (ICS), which enables the opportunity for a comprehensive comparison of different embedding methods based on a single uncompromized network traffic capture as cover. It is motivated by the observation that industrial control systems are increasingly under attack by stealthy malware, e.g., for reloading malicious code and for data in- and exfiltration. Although multiple detection mechanisms based on published attacks have been developed in recent years, the diversity of steganographic attacks is still a major challenge, and the elaboration of further analysis mechanisms for detection and attribution is a constant arms race. In an exemplary evaluation of three embedding methods by the proposed framework, it is demonstrated that it is possible to assign 88.6% of the samples to a specific steganographic embedding method based on a machine learning approach, proofing the conceptual functionality of the framework. Also, it is shown that different characteristics of payloads can be identified in part. The proposed concept can thus help to derive further detection & defense mechanisms and to differentiate between embedding methods, as well as embedded message types, which increases the potentials for an attribution of attackers in the future, in addition to the detection of incidents.**

Keywords-Information Hiding; Intrusion Detection and Attribution; Network Steganography; Stealthy Malware; Industrial Control Systems; Analysis Framework.

I. Introduction

During the last several years, information hiding based malware (also stealthy malware) has become more and more popular and is increasingly used by attackers, this can be confirmed by recently presented attack vectors like [1]. Stealthy malware uses completely unobtrusive data to create hidden channels, which are then utilized to embed malicious code or to activate malware for example. Since the Stuxnet-Attack in 2010, it has been clear that also Industrial Control Systems (ICS) are under attack with stealthy malware. In this attack, Ink-files were utilized as cover data and in-memory code injections were used to hide the attack [2]. Additionally, recent attacks like the Ukrainian [3] and the Indian power grid attack [4] demonstrate that attacks with information hiding based malware on ICS become more and more common, especially due to the motivation to stay undetected as long as possible in order to in- and exfiltrate stealthy data.

Currently, several potential information hiding attack vectors for stealthy malware with steganographic embedding techniques and potential defense mechanisms are introduced (e.g., in [5], [6] and [7]). Unfortunately, a framework for comprehensive analysis and comparison of these methods is missing to identify potential similarities, differences, and effects on the cover data and to derive defense mechanisms for specific embedding methods. Additionally, a comprehensive analysis could enable the possibility to distinguish between analyzed embedding methods after a detection, which can lead to the opportunity to identify potential attackers (attribution).

Thus, this work contributes a novel analysis framework, which offers the possibility to compare and analyze multiple steganographic embedding methods based on only a single uncompromized network traffic capture from an exemplary ICS to offer the chance to analyze and evaluate the behaviour of covert information hiding channels (which are used for stealthy steganographic malware). Additionally, the introduced analysis framework for network steganography in ICS is used for an extensive evaluation of three exemplary selected embedding methods (two from state-of-the art and one novel embedding method) to find out, if there is a possibility to differentiate between them and the embedded types of messages (invariant and heterogeneous message) using a machine learning approach based on handcrafted features and a neural network as classification engine.

The paper is structured as follows: In Section II, we present related work and fundamentals. In Section III, we introduce our novel analysis framework. Our evaluation setup to analyze three embedding methods with the novel framework including evaluation goals, data and environment is presented in Section IV. Section V presents the evaluation results and Section VI concludes the paper with a summary and future work.

II. Fundamentals and Related Work

In this section, we summarize fundamentals for network steganography in ICS, present recent steganographic attack vectors for network steganography in ICS, and introduce a method to produce synthetic steganographic network data for a fast and easy generation of network data with recent steganographic embedding methods. Furthermore, an overview of methods to analyze steganographic network data for detection and attribution purposes is given.

A. Network Steganography in ICS

"Steganography is the art and science of concealing the existence of information transfer and storage", according to [8]. The subdomain network steganography targets the transfer and

storage of hidden information in network communication traffic. From an attacker's perspective, a warden (e.g., intrusion detection system) observes the network traffic and the embedding of stealthy malware should be inconspicuous in a sense that a warden would not be able to differentiate between genuine communication and communication with hidden information embedding [5]. An embedding of hidden information with steganographic techniques can be realized, for example by manipulating the network packets payload on least significant values or by modulating time intervals between specific packets [9].

Network steganography and stealthy malware in ICS are special, due to the lower amount of available data for potential embedding than in traditional Information Technology (IT) networks. Furthermore, the transmitted network packets are usually smaller in ICS since only meta-data or few values (e.g., from sensors) are transferred per packet. Additionally, ICS specific protocols like OPC UA (Open Platform Communications Unified Architecture) [10] or Modbus-TCP [11] are often encapsulated in TCP/IP (or other transport protocols), which enables the opportunity for utilizing the data fields of the ICS specific protocols in addition to TCP/IP protocol headers. It is also not uncommon for the ICS-specific payload to be transmitted unencrypted, because ICS are considered as closed networks and not subject to attacks.

Potential network steganographic embedding patterns and a related terminology is summarized in [12]. A generic taxonomy and overview with the intention of a unified understanding of terms and their applicability for network steganographic methods can be found in [8].

B. Selected Steganographic Embedding Methods for ICS

Two recent and relevant exemplary attack vectors with steganographic embedding techniques in ICS are presented in this section. These Embedding Methods (EM) are selected because both use a timestamp modulation (i.e., timing channel) to embed hidden information, which is a plausible attack vector since every network packet includes them. The presented EM will be analyzed and compared with our novel framework.

1) Selected Embedding Method 1 (EM_1): The approach presented in [5] uses package timestamps (T_i) for embedding while utilizing a dynamic encoding approach based on the hour, minute, and second values, as well as an embedding key and an initialization vector. Similar to the aforementioned approach, low-value-digits of the timestamp are manipulated. This approach is able to hide one *ASCII*-symbol in four of the five highlighted digits of a timestamp in the coding "HH:MM:SS.fffffff", where H,M,S,f stand for digits of the hour, minute, second and fractional digits of the second of the time value respectively (Example: T_i= 10:00:00.123**45678**9). The actual embedding positions are determined using the embedding key and the first digit right of the floating point digit for the fractional second values. Converting a sequence of *ASCII*-symbols to binary values results in a bitstream that is embedded chronologically into every available package. Due to the different modulated values of the variables involved, the encoding of the output values vary in perception.

2) Selected Embedding Method 2 (EM_2): A quite simple and easy to comprehend embedding method is introduced in [6]. The embedding scheme assumes an attack vector with a corrupted Programmable Logical Controller (PLC) via Supply-Chain-Attack. The PLC sends delays in millisecond range ($\mu s_1, \mu s_2, \mu s_3$) to embed a hidden message via timing delays. This means an exemplary timestamp T_i = 10:00:00.123**456**789 is manipulated on the digit positions $\mu s_1 = 4, \mu s_2 = 5, \mu s_3 = 6$. Based on the introduced steganographic taxonomy in [8] (see Section II-A), this represents a *Random State/Value Modulation*. The embedding scheme converts an *ASCII*-message into a bitstream *BS*. For embedding a bit of *BS*, three consecutive PLC OPC UA timestamps are altered (T_i, T_{i+1}, T_{i+2}). To stay inconspicuous, the following three PLC packet timestamps stay untouched (T_{i+3}, T_{i+4}, T_{i+5}). The approach arbitrarily chooses the digit '4' to embed bit = 0 and digit '9' to embed bit = 1.

C. Synthetic Steganographic Data Generation

Diverse and heterogeneous steganographic data for ICS is needed to train and evaluate potential defense mechanisms. But each steganographic embedding needs a mostly sophisticated and complex setup which is very time consuming to assemble and in addition, it raises various security and safety issues. Because of this, the approach of [6] introduces a concept to generate artificial steganographic network data with a limited embedding pace and a specific steganographic embedding technique based on TCP-timestamps. Based on [6], an advanced Synthetic Steganographic Embedding (SSE)-concept is presented in [13]. It offers the possibility to embed hidden information literally everywhere in uncompromized network packet recordings with a embedding pace near real time. This makes it possible to quickly and easily generate test data for many different embedding methods for analysis. In [6], it is assumed that the most important aspects to be simulated in a network traffic are:

1) the physical network including layout and components,
2) the network traffic including types of flows, directions, protocols used, typical payloads, etc., and
3) the type and characteristics of the (steganographic) hidden channel.

Both approaches simulate only the last aspect (3) of this list, the two others are directly used from an uncompromized recording of a physical setup. The SSE-concept has two Synthetic Embedding Options (SEO), one option focuses on a very fast and efficient embedding without accessing structural elements of a packet (called SEO_A), the other option delivers a more comfortable embedding with an easier access to structural elements of a network packet based on json-objects (called SEO_B). In this work, the SSE-concept from [13] is used to generate the steganographic data based on the selected embedding methods in our novel analysis framework, which is presented in the following Section III.

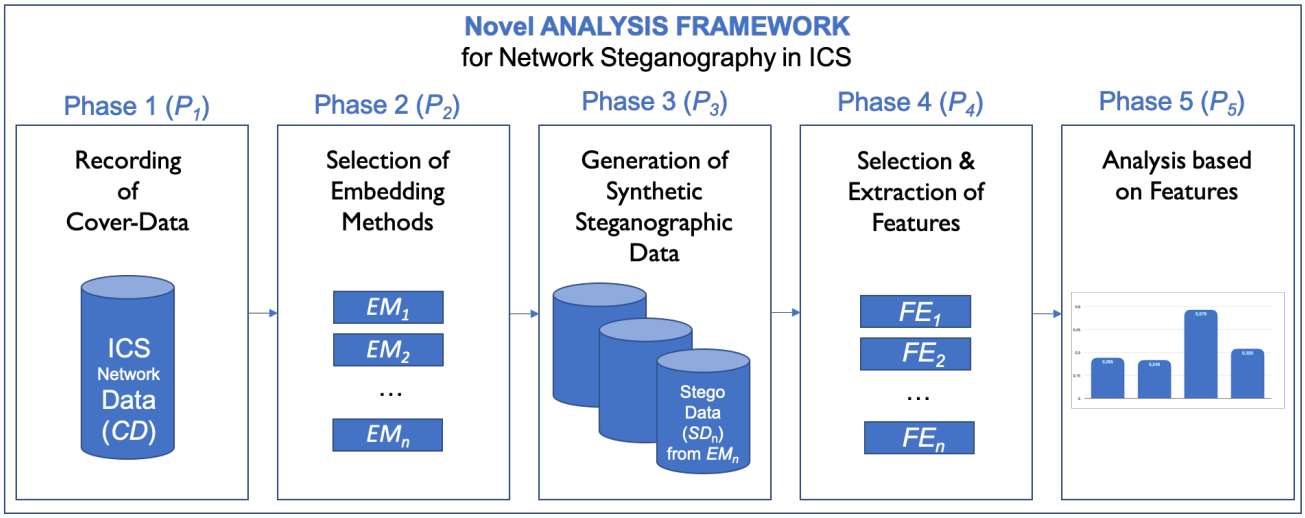

Figure 1. Novel Analysis Framework

D. Analysis of Steganographic Network Data

A basic overview of potential methods to analyze and defend against stealthy malware based on network steganography is presented in [14]. In [15], a novel machine learning based approach is presented to detect network steganography in network recordings based on a handcrafted feature space with an accuracy of 92.9%. The approach analyzes the last six digit positions of a numeric value, because it is best suited for an unobtrusive embedding. The approach performs a frequency analysis of occurrence for the digits 0 to 9 on these six specific positions. This results in 10 features (values) for each digit position between 0.0 and 1.0 representing the percentage of occurrence for each digit 0 to 9. This feature space will be used in this work to analyze the different embedding methods used in this paper for analysis.

III. Novel Analysis Framework

Our novel analysis framework to compare and evaluate different embedding methods to offer the possibility to make a distinction between them for a potential determination or classification of attackers or embedded message types is shown in Figure 1. The concept includes five phases:

- Phase 1 (P_1): the recording of cover-data (see Section III-A),
- Phase 2 (P_2): the selection and formalization of embedding methods (see Section III-B),
- Phase 3 (P_3): the generation of synthetic steganographic data (see Section III-C),
- Phase 4 (P_4): the selection and extraction of features (see Section III-D), and
- Phase 5 (P_5): the analysis based on the features (see Section III-E).

The phases of the framework are described in detail in the following subsections. An exemplary initial evaluation of three embedding methods performed in our evaluation (evaluation setup in Section IV and results in Section V).

A. Recording of Cover Data (P_1)

The framework begins with Phase 1 where network Cover Data (CD) has to be recorded from an uncompromized laboratory ICS setup. CD can be recorded with different capturing tools, we recommend *Wireshark* [16]. The output file is provided in *pcap* or *pcapng* file format for further processing and should contain only relevant traffic for a specific purpose. These file formats are well suited logging protocols for the structural recording of network data. CD is used to build the statistic baseline of the ICS network data to illustrate the impact of the embedding during the analysis and it is also the basis for the steganographic embedding with the selected embedding methods (see Phase 2) to generate the steganographic network data in Phase 3.

B. Selection and Formalization of Embedding Methods (P_2)

Once a network cover data file is recorded, embedding methods for the analysis in Phase 5 have to be selected and should be formalized with a pseudo code representation for a uniform, comparable and comprehensible illustration. In this work, we select two embedding approaches from state-of-the-art and elaborate a novel embedding method. As explained in Section II-B2 all of the algorithms work with an Array A ($A = \{T_1, ..., T_i\}$) that contains all Timestamps T_i of network packet available for manipulation in our pseudo code representation. The specific formalizations for the state-of-the-art approaches EM_1, EM_2, and the novel embedding method EM_3 will be described in the following subsections.

1) Formalization of Embedding Method EM_1: EM_1 takes a dynamic encoding approach while manipulating low value digits of the OPC UA timestamp. An initialization vector I and an encoding key K are used in addition to variables taken from each timestamp to encode the hidden message. D, E,

F, G (meaning: see Figure 2) are all derived directly from the timestamp, as well as H ($H = \{H_0, ..., H_3\}$), which is the 4-digit field in which the encoded message c is embedded. After the encoding process depicted in Figure 2 is finished, the output of S decides the embedding position in H.

Algorithm 1 Steganographic Embedding Method EM_1

$AM \leftarrow A$
$i \leftarrow 0$
$K \leftarrow 4\ Digit\ Key$
$I \leftarrow 4\ Digit\ Initialization\ Vector$
while $i < Length(A)$ **do**
  $D \leftarrow Hour\ value\ of\ T_i$
  $E \leftarrow Minute\ value\ of\ T_i$
  $F \leftarrow Second\ value\ of\ T_i$
  $G \leftarrow Value\ of\ digit\ 1\ after\ floating\ point\ of\ T_i$
  $H \leftarrow Value\ of\ digit\ 2{-}6\ after\ floating\ point\ of\ T_i$
  $S \leftarrow G \oplus DigitSum(K)\ mod\ 2$
  $O \leftarrow D \times E \times F\ mod\ 10000$
  $K' \leftarrow \sum_{n=0}^{3}((K_n \oplus (G + I_n))\ mod\ 10) \times 10^n$
  $K'' \leftarrow O \oplus K'\ mod\ 10000$
  $c \leftarrow m \oplus K''\ mod\ 8192$
  if $S == 0$ **then**
    $H_0, H_1, ..., H_3 \leftarrow c$
  else if $S == 1$ **then**
    $H_1, H_2, ..., H_4 \leftarrow c$
  end if
  $AM[i] \leftarrow T_i$
end while

Figure 2. Formalized Algorithm for EM_1.

2) Formalization of Embedding Method EM_2: Iterating through A, EM_2 embeds a bit of the input bitstream into 3 consecutive timestamps, encoding 0 and 1 by the digital values of 4 and 9, respectively. In the process, three different digits are used for the embedding represented in $\mu_1 - \mu_3$. Manipulated timestamps are then saved in the AM array. This is repeated for each bit in the bitstream until the end of A is reached or all bits are embedded (the algorithm is represented in Figure 3).

Algorithm 2 Steganographic Embedding Method EM_2

$AM \leftarrow A$
for Bit in $Bitstream$ **do**
  for $i \leftarrow 1$ to 3 **do**
    if Bit_i is 0 **then**
      $T_i[\mu_{i\ mod\ 3}] \leftarrow 4$
    else if Bit_i is 1 **then**
      $T_i[\mu_{i\ mod\ 3}] \leftarrow 9$
    end if
    $AM[i] \leftarrow T_i$
  end for
end for

Figure 3. Formalized Algorithm for EM_2.

3) Formalization of Embedding Method EM_3: With the addition of a key used for dynamic encoding and positioning, an advancement of EM_2 is introduced and represented in EM_3. Thus, EM_3 is a novel and sophisticated approach based on EM_2 which has been elaborated for a more unobtrusive embedding. Introducing the dynamic ciphers C_0 and C_1 for the bit values 0 and 1 respectively dynamic encoding is achieved by using K as seed for a random number generator. In cases of collision, the values are generated again. Adding C_0 and C_1 while reducing the sum to a maximum of 3 using the modulo function the position of each bit in the timestamp is calculated (see algorithm representation in Figure 4).

Algorithm 3 Steganographic Embedding Method EM_3

$AM \leftarrow A$
$i \leftarrow 0$
$K \leftarrow$ "$SyntheticStegoKey$"
for Bit in $Bitstream$ **do**
  for $i \leftarrow 1$ to 3 **do**
    $C_0 \leftarrow 0$
    $C_1 \leftarrow 0$
    while $C_1 == C_2$ **do**
      $C_0 \leftarrow Random(K)\ mod\ 9$
      $C_1 \leftarrow Random(K)\ mod\ 9$
    end while
    $j \leftarrow C_0 + C_1\ mod\ 3$
    if Bit_i is 0 **then**
      $T_i[\mu_j] \leftarrow C_0$
    else if Bit_i is 1 **then**
      $T_i[\mu_j] \leftarrow C_1$
    end if
    $AM[i] \leftarrow T_i$
  end for
end for

Figure 4. Formalized Algorithm for EM_3.

C. Generation of Synthetic Steganographic Data (P_3)

For the creation and generation of the steganographic network data based on the embedding methods from Phase 2 (EM_1, EM_2, and EM_3), the SSE-concept [13] (introduced in II-C) is used. As mentioned, the SSE-concept offers the possibility to generate steganographic network data synthetically and this results in some obvious advantages for our framework: no matter which embedding method is analyzed, it is not required to elaborate a corrupted, complex ICS setup, which generates the steganographic network data with hidden information. Thus it is well suited because it delivers the opportunity for an easy and fast generation of steganographic network data without the need of a physical setup. The SSE-concept includes four segments:

- Segment I: Record and Pre-Process Network Data,
- Segment II: Synthetic Embedding Option A (SEO$_A$),
- Segment III: Synthetic Embedding Option B (SEO$_B$), and
- Segment IV: Retrieval.

Segment I also deals with the recording of network data, thus Segment Element (SE) I.1 can be skipped for our novel analysis framework since the data capturing is completed after P_1. For the synthetic generation of steganographic network data it

offers two synthetic embedding options (Segment II: SEO_A and Segment III: SEO_B, see Section II-C). In the evaluation, this work uses SEO_B since it offers a more comfortable embedding with access to structural elements of a network packet based on json-objects.

D. Selection and Extraction of Features (P_4)

To extract features from pcap or pcapng files, the relevant structural element of the relevant network packets should be converted into csv or txt data to process it afterwards. Therefore, *Tshark* (*Wireshark* console application) [16] with the *-T fields -e field* option can be used to select data fields of network packets that are relevant for feature extraction and analysis. We recommend to use handcrafted statistical feature spaces with as much discriminatory power as possible to analyze steganographic network data. This should lead to comprehensible and plausible analysis results.

In this work, we use the introduced handcrafted feature space from [15], which performs a frequency analysis on least significant digits in timestamps to analyze the three selected exemplary embedding methods (EM_1, EM_2, and EM_3), the approach is briefly described in Section II-D. The selected features shall be extracted for multiple samples from multiple embedding approaches for analysis in P_5.

E. Analysis (P_5)

Based on the extracted features from multiple embedding methods in P_4 a statistical analysis can be carried out. Therefore, various statistical computational techniques such as machine or deep learning based approaches can be taken into consideration based on the selected and extracted features. Thus, for the analysis, different data mining and machine learning tools or libraries, such as *WEKA* [17], *Orange* [18], *Tensorflow* [19] or *Keras* [20], are well suited to analyze differences and commonalities of embedding methods. Generally, the analysis can focus different use case specific aspects, for example: detectability, attributability, embedding scheme, and more depending on goals and objectives of a study.

For our analysis, we train a machine learning based classification approach based on the extracted features to achieve our evaluation goals (presented in Section IV-A).

IV. Evaluation Setup

In our evaluation, we use our presented framework from Section III to compare and analyze the three introduced embedding methods (EM_1, EM_2, and EM_3). In this section, we describe our evaluation goals, as well as the laboratory ICS setup and the resulting evaluation data. Furthermore, we describe our analysis environment to achieve our goals. The evaluation results are presented in Section V.

A. Evaluation Goals

The evaluation has the following goals:

- G_1: Analysis of the three exemplary embedding methods (EM_1, EM_2, and EM_3) based on the extracted features (see Section III-E) to determine whether a potential distinction between the methods is possible for a potential detection of attackers.
- G_2: Analysis of different message types (invariant and heterogeneous) embedded with EM_1, EM_2, and EM_3 to determine whether a potential distinction between embedded messages is possible.

Results for G_1 are presented in Section V-A and for G_2 in Section V-B.

B. Evaluation Data

As mentioned before, to generate synthetic steganographic ICS network data, we have to record cover data from an uncompromized ICS setup (see Section III-A). Thus, we setup a lean laboratory ICS (for better comprehensibility). In our laboratory setup a *Siemens S7-1500* Programmable Logical Controller (PLC) communicates with an Human-Machine-Interface (HMI). On the PLC (server), multiple exemplary automation tasks are running (e.g., traffic light control, binary process status reports, temperature measuring). The HMI (client) requests all PLC outputs every 100 milliseconds. The network traffic on this setup is captured through a mirror port on the connected switch. The laboratory ICS communicates via the ICS specific OPC UA protocol. We have recorded the network communication for around 61 minutes (3,706 seconds). 38,189 network packets (half requests (client, HMI), half responses (server, PLC) were captured and the file size is 7,351,828 Bytes. This capturing builds our steganographic cover data to generate synthetic steganographic network traffic (see the next subsection). The recorded cover data set is called REC_{CD}.

As described previously, our steganographic network data for evaluation is generated with the SSE-concept [13] and the introduced and formalized embedding methods EM_1, EM_2, and EM_3 (see Section III-B). All these embedding methods assume an attack vector with a corrupted PLC via Supply-Chain-Attack. The PLC sends delayed packets in micro- and nanosecond range to embed a hidden message via timing delays with EM_1, EM_2, and EM_3. This means an exemplary timestamp T_i = 10:00:00.123**456789** is manipulated on the bold marked digit positions. Based on the introduced steganographic taxonomy in [8] (see Section II-A) this represents an *LSB state/value modulation*, as mentioned before. The embedding scheme converts an *ASCII*-message into a bitstream *BS*.

For our setup, we embed separately in cover data CD with the three introduced embedding methods. Furthermore, we embed two kinds of messages with each embedding method separately in CD, a heterogeneous message ('$securware2024$') and an invariant message consisting solely of a single character ('a'), both message types are repeated and embedded as often as possible in the data to corrupt all available packets of the cover data. Thus, six steganographic data sets are created and presented in Table I. These data sets and the cover data set (REC_{CD}) are used for feature extraction (see Section IV-C).

TABLE I
NETWORK DATA SETS FOR FEATURE EXTRACTION; STEGANOGRAPHIC DATA IS EMBEDDED SYNTHETICALLY IN REC_{CD}.

Name	Type of Recording	Embedding Method	Message Type	Hidden Message	No. of relevant Packets
REC_{CD}	Cover Data Recording	-	-	-	19,094
$REC_{EM1_{IV}}$	Steganographic Data	EM_1	invariant	a (repeated)	19,094
$REC_{EM1_{HE}}$	Steganographic Data	EM_1	heterogenous	$securware2024$ (repeated)	19,094
$REC_{EM2_{IV}}$	Steganographic Data	EM_2	invariant	a (repeated)	19,094
$REC_{EM2_{HE}}$	Steganographic Data	EM_2	heterogenous	$securware2024$ (repeated)	19,094
$REC_{EM3_{IV}}$	Steganographic Data	EM_3	invariant	a (repeated)	19,094
$REC_{EM3_{HE}}$	Steganographic Data	EM_3	heterogenous	$securware2024$ (repeated)	19,094

C. Analysis Environment

To achieve our goals (G_1 and G_2) presented in Section IV-A, we extract features from the network data sets (see Table I) to train a classifier to determine if we can distinguish between cover data and the different embedding approaches (G_1) and to analyze if we can distinguish between embedded message types (G_2). As mentioned, therefore we use the feature space [15] which performs a frequency analysis on the micro and nanosecond digits of network packet timestamps (as described in Section II-D). To do so, we extract a labeled feature vector (label based on embedding method and message type) from every recorded network data set. We iterate through every recorded network data set and extract a feature vector after 100 relevant packets, which results in 190 samples per data set (since 19,094 relevant packets are included in every data set). Based on the extracted feature space, we train a Multilayer Perceptron (MLP) as classification engine with the feature space as input layer followed by three hidden layers (with 64,32,16 neurons and rectifier activation function) because it is a powerful and modern classification algorithm known for its accurate performance. For evaluation, we perform 5-fold cross-validation to determine the classification accuracy (i.e., results) for G_1 and G_2 (visualized in Figure 5). For G_1, we train a MLP (MLP$_{4C}$) with 4 classes (cover, EM_1, EM_2, EM_3; represented as 4 neurons in the output layer of the MLP) because the feature vectors extracted from heterogeneous and invariant message types are merged here, because for G_1, we only want to distinguish between cover and embedding types. For G_2 a 7-class-MLP (MLP$_{7C}$) including all labeled feature vectors from all recordings is trained and the classes are represented as 7 neurons in the output layer of the MLP. An overview of the models and the included feature vectors are presented in Table II.

Figure 5. Analysis Environment for Evaluation.

TABLE II
FEATURE VECTORS (I.E., SAMPLES) INCLUDED IN MLP$_{4C}$ AND MLP$_{7C}$ FOR EVALUATION OF G_1 AND G_2

Name	Label of Vectors	extracted from:	Number of Vectors	Goal
In MLP$_{4C}$ included vectors:				
VEC_{CD}	CD	REC_{CD}	190	G_1
VEC_{EM1}	EM_1	$REC_{EM1_{IV}}$, $REC_{EM1_{HE}}$	380 (2x190)	
VEC_{EM2}	EM_2	$REC_{EM2_{IV}}$, $REC_{EM2_{HE}}$	380 (2x190)	
VEC_{EM3}	EM_3	$REC_{EM3_{IV}}$, $REC_{EM3_{HE}}$	380 (2x190)	
In MLP$_{7C}$ included vectors:				
VEC_{CD}	CD	REC_{CD}	190	G_2
$VEC_{EM1_{IV}}$	$EM1_{IV}$	$REC_{EM1_{IV}}$	190	
$VEC_{EM1_{HE}}$	$EM1_{HE}$	$REC_{EM1_{HE}}$	190	
$VEC_{EM2_{IV}}$	$EM2_{IV}$	$REC_{EM2_{IV}}$	190	
$VEC_{EM2_{HE}}$	$EM2_{HE}$	$REC_{EM2_{HE}}$	190	
$VEC_{EM3_{IV}}$	$EM3_{IV}$	$REC_{EM3_{IV}}$	190	
$VEC_{EM3_{HE}}$	$EM3_{HE}$	$REC_{EM3_{HE}}$	190	

V. EVALUATION RESULTS

In this section, the classification results of the 5-fold cross validation for G_1 and G_2 are presented and analyzed.

A. Results for G_1

In G_1, we want to determine, if the feature space from [15] in combination with a multilayer perceptron as classification engine (see Section IV-Cl MLP$_{4C}$) is able to distinguish between the selected embedding methods EM_1, EM_2, and EM_3. The presented confusion matrix in Table III shows that most of the samples can be correctly classified (76.84%) during a 5-fold cross validation.

TABLE III
CONFUSION MATRIX OF 5-FOLD CROSS VALIDATION OF MLP$_{4C}$ FOR G_1 (**BOLD**: CORRECT CLASSIFIED SAMPLES)

classified as $->$ / Actual	CD	EM_1	EM_2	EM_3
CD (190)	**12**	150	0	28
EM_1 (380)	78	**298**	0	4
EM_2 (380)	0	0	**380**	0
EM_3 (380)	27	21	0	**332**

Especially the results for EM_2 and EM_3 are accurate, MLP$_{4C}$ has correctly classified 100.0% of VEC_{EM2} (i.e., samples from EM_2) and 87.4% correctly classified VEC_{EM3}. Furthermore, we determined an accuracy of 78.4% for VEC_{EM1} (298 of 380 correctly classified). Thus, we can state that MLP$_{4C}$ can distinguish correctly between embedding methods for 88.6% of evaluated samples (1010 of 1140), see Figure 6.

TABLE IV
CONFUSION MATRIX OF 5-FOLD CROSS VALIDATION OF MLP_{7C} FOR G_2
(**BOLD**: CORRECT CLASSIFIED SAMPLES, *italic:* CORRECT CLASSIFIED EMBEDDING METHOD BUT MISCLASSIFIED MESSAGE TYPE)

classified as $->$ / Actual ($\sum$)	CD	$EM1_{IV}$	$EM1_{HE}$	$EM2_{IV}$	$EM2_{HE}$	$EM3_{IV}$	$EM3_{HE}$
CD (190)	**80**	7	8	19	20	39	17
$EM1_{IV}$ (190)	66	**18**	*28*	16	17	31	14
$EM1_{HE}$ (190)	58	*23*	**22**	16	16	38	17
$EM2_{IV}$ (190)	9	0	5	**126**	*35*	15	0
$EM2_{HE}$ (190)	2	0	4	*68*	**107**	9	0
$EM3_{IV}$ (190)	36	2	7	23	26	**62**	*34*
$EM3_{HE}$ (190)	38	1	7	29	22	*69*	**24**

However, the limitations of the approach can not be ignored. It is not able to differentiate between the cover data and EM_1. But this is reasonable, according to [5]. It is related to the size of analyzed packets (100 packets in our setup to extract a feature vector (sample), see Section IV-C), because the embedding becomes more and more unobtrusive the more packets are analyzed (in contrast: EM_2, and EM_3 become more obtrusive according to [15]).

Derived from this, we can state that the approach (feature space from [15] and MLP_{4C} as machine learning based classification engine) is well suited for a usage after a potential anomaly detection in network data for the classification of trained embedding methods to attribute the embedding method used by potential attackers for covered communication.

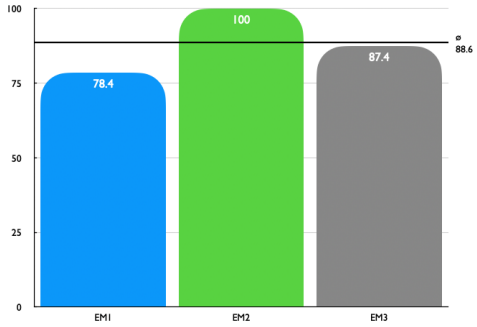

Figure 6. Percentage Share of correctly assigned Samples to Embedding Methods.

B. Results for G_2

For evaluation goal G_2, we analyze if our machine learning based approach is able to distinguish between two types of messages (invariant and heterogeneous). As the results in Table IV show, the approach is able to distinguish between embedding methods for most of the extracted feature vectors (samples) which confirms the results from G_1, but the message type can only be distinguished for EM_2 for the majority of samples (61.3% 233 of 380 samples). For EM_1 and EM_3 the message types are mostly misclassified, which means that our used feature space is not able to distinguish it, we also tried it with other classifiers (for example Decision Trees or a Support Vector Machine) but the results were comparable. If we look at the formalization of EM_1 (see Figure 2) and EM_3 (see Figure 4), the classification results are not surprising because the formalizations let us assume that the type of the embedded message should not result in statistically significant differences. Thus, it is plausible that the selected feature space has no discriminatory power to distinguish between embedded message types for these embedding methods. Additionally, the results also show (also from G_1) that our framework is well suited to confirm specific assumption about specific embedding methods and to determine possibilities and limitations which can be taken into account while elaborating novel defense mechanisms.

VI. CONCLUSION AND FUTURE WORK

In this paper, we introduce a novel analysis framework for steganographic network data in ICS. Information hiding techniques embedded with steganographic methods can be used to implement malware, communicating in ICS network traffic. The framework offers the possibility to compare and analyze multiple steganographic embedding methods and to derive defense mechanisms based on the analysis. In this work, we exemplarily analyze three embedding methods based on a straightforward laboratory setup with the novel framework and derive an approach, which is able to distinguish between the evaluated embedding methods after detection with an accuracy of 88.6%. Additionally, we were able to differentiate between invariant and heterogeneous message types for embedding method EM_2 for the majority of samples (61.3%).

These results confirm that the novel framework presented in this paper is well suited to analyze different steganographic embedding methods and it offers an easy and fast possibility to confirm use case specific assumptions about the behaviour of covert information hiding channels, which are used for stealthy malware to elaborate potential detection and defense mechanisms, and to additionally attribute potential attackers.

In future work, the framework will be used to analyze other embedding methods with other payload and embedding places (e.g. sensor values like in [21]). Additionally, we would like to analyze the opportunity to differentiate between message types more accurately with, for example, a novel handcrafted feature space. Furthermore, it should be analyzed if the used machine learning based approach (handcrafted feature space from [15] with MLP as classification engine) is able to attribute more

attackers that use different types of steganographic embedding methods and message types that are, for example, not included into training data.

ACKNOWLEDGEMENTS

The research in this work has been performed in context of the project ATTRIBUT (https://omen.cs.uni-magdeburg.de/itiamsl/deutsch/projekte/attribut.html) jointly by a teaching project at "Brandenburg University" in term 2023/2024. This comprises in particular the conceptional design of the experimental analysis framework and embedding method EM_3, software realization in Python of all algorithms for embedding and feature extraction in Section III. It was further supported by the evaluation dataset (Section IV-B) generously contributed by the project "SYNTHESIS", funded by the German Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (BMUV, project no. 1501666B) in the framework of the German reactor safety research program.

REFERENCES

[1] MITRE-ATT&CK, "Data obfuscation: Steganography", *https://attack.mitre.org/versions/v14/techniques/T1001/002/, last access: 19/09/24*, 2020.

[2] D. Kushner, "The real story of stuxnet", *https://spectrum.ieee.org/the-real-story-of-stuxnet, last access: 19/09/2024*, 2013.

[3] R. M. Lee, M. J. Assante, and T. Conway, "Analysis of the cyber attack on the ukrainian power grid", SANS Institute, Tech. Rep., 2016.

[4] I. Dragos, "Assessment of reported malware infection at nuclear facility", *https://www.dragos.com/blog/industry-news/assessment-of-reported-malware-infection-at-nuclear-facility/*, 2019.

[5] M. Hildebrandt, K. Lamshoeft, J. Dittmann, T. Neubert, and C. Vielhauer, "Information hiding in industrial control systems: An opc ua based supply chain attack and its detection", IH&MMSec 2020, pp. 115–120, 2020. DOI: 10.1145/3369412.3395068.

[6] T. Neubert, C. Kraetzer, and C. Vielhauer, "Artificial steganographic network data generation concept and evaluation of detection approaches to secure industrial control systems against stegano- graphic attacks", *In The 16th International Conference on Availability, Relia- bility and Security (ARES 2021), August 17–20, 2021, Vienna, Austria. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3465481.3470073*, 2021.

[7] K. Lamshoeft, T. Neubert, J. Hielscher, C. Vielhauer, and J. Dittmann, "Knock, knock, log: Threat analysis, detection & mitigation of covert channels in syslog using port scans as cover", *Digital Investigation 2022 (DFRWS EU 2022)*, 2022.

[8] S. Wendzel *et al.*, "A generic taxonomy for steganography methods", Jul. 2022. DOI: 10.36227/techrxiv.20215373.v1.

[9] W. Mazurczyk, S. Wendzel, and K. Cabaj, "Towards deriving insights into data hiding methods using pattern-based approach.", *ARES 2018, 13th International Conference on Availability, Reliability and Security; Hamburg, Germany, August 27 - August 30, ISBN: 978-1-4503-6448-5*, 2018.

[10] OPC-Foundation, "Unified architecture", *https://opcfoundation.org/about/opc-technologies/OPCUA/, last access: 19/09/24*, 2008.

[11] ACROMAG-Incorporated, "Introduction to modbus tcp/ip", *https://www.prosoft-technology.com/kb/assets/intro_modbustcp.pdf, last access: 19/09/24*, 2005.

[12] S. Wendzel *et al.*, "A revised taxonomy of steganography embedding patterns.", *In the Proceedings of 16th International Conference on Availability, Reliability and Security (ARES 2021), Article No.: 67, Pages 1 - 12, August 17–20, 2021, Vienna, Austria. ACM, New York, NY, USA, 12 pages. https://doi.org/10. 1145/3465481.3470069*, 2021.

[13] T. Neubert, B. Peuker, L. Buxhoidt, E. Schueler, and C. Vielhauer, "Synthetic embedding of hidden information in industrial control system network protocols for evaluation of steganographic malware", *Tech. Report, arXiv, https://doi.org/10.48550/arXiv.2406.19338*, 2024.

[14] L. Caviglione, "Trends and challenges in network covert channels countermeasures", *Applied Sciences*, vol. 11, Feb. 2021. DOI: 10.3390/app11041641.

[15] T. Neubert, A. J. C. Morcillo, and C. Vielhauer, "Improving performance of machine learning based detection of network steganography in industrial control systems.", *In the Proceedings of 17th International Conference on Availability, Reliability and Security (ARES 2022), Article No.: 51, pp. 1 - 8, August 23– 26, 2022, Vienna, Austria. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3538969.3544427*, 2022.

[16] Wireshark-Foundation, "About wireshark", *https://www.wireshark.org/about.html, last access: 19/09/24*, 2024.

[17] M. Hall, "The weka data mining software: An update.", *In SIGKDD Explorations*, 2009.

[18] J. Demšar *et al.*, "Orange: Data mining toolbox in python", *Journal of Machine Learning Research*, vol. 14, pp. 2349–2353, 2013.

[19] M. Abadi *et al.*, *TensorFlow: Large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015.

[20] F. Chollet *et al.*, *Keras*, https://keras.io, last access: 19/09/24, 2015.

[21] K. Lamshoeft, C. Kraetzer, J. Dittmann, T. Neubert, and C. Vielhauer, "Information hiding in cyber physical systems: Challenges for embedding, retrieval and detection using sensor data of the swat dataset", *In Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security (IHMMSec '21), pp. 113 - 124, June 22–25, 2021, Virtual Event, Belgium. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3437880.3460413*, 2021.