ADAM - An Adversary-Driven Attack Modelling Framework for Model-Based Security Testing

Tina Volkersdorfer, Hans-Joachim Hof
Security in Mobility
CARISSMA Institute of Electric, Connected, and Secure Mobility (C-ECOS)
Technische Hochschule Ingolstadt, Germany
tina.volkersdorfer@carissma.eu, hans-joachim.hof@thi.de

***Abstract*—ADAM (Adversary-Driven Attack Modelling) is a framework for model-based security testing. It is the foundation of a systematic and holistic attack modelling to support consistent and comprehensible penetration tests on model level. ADAM can be used for the automation of security testing in the early phases of software engineering (e.g., manual security reviews) as well as providing attack information for testing activities in later phases of the development lifecycle (e.g., penetration tests). By using ADAM, it is possible to continuously and consistently address security in software development, even if no running code is available. This paper focuses on the presentation of the concept of ADAM, describing the necessary components, their use and giving an insight into how the ADAM framework can be used in the context of a simulation environment. ADAM captures different perspectives of an attack, by the simulation of an adversary that executes multiple attacks to reach a given goal. Thus, ADAM supports not only the automation of model-based security tests but the whole security testing on model level, e.g., including test case generation. Our preliminary evaluation shows that it is possible to use ADAM in a wide range of domains and that there is potential reuse of modelled elements.**

Index Terms—attack model; adversary model; model-based testing; security testing; penetration test.

I. Introduction

In this extended paper, we present the Adversary-Driven Attack Modelling (ADAM) framework. This work expands the basic idea of a holistic attack modelling framework to support the model-based security testing presented in [1] by the following aspects:

- Specifying the adversary model utilising attributes for the adversary characteristic and the adversary goal.
- Giving an insight into how the Adversary-Driven Attack Modelling (ADAM) framework can be used and integrated, in the context of a simulation environment from the automotive domain, which represents the target model.

The ADAM framework is being developed in the research project “Modellbasierte Absicherung von Security und Safety für umfeldbasierte Fahrzeugfunktionen (MASSiF)” that addresses model-based safety and security testing in the automotive software domain. In the automotive domain, software engineers thoroughly use model-based safety engineering and model-based safety testing, e.g., to develop advanced driver-assistance systems [2]. However, to our best knowledge, there are currently no approaches for holistic attack-model-based security testing. This argument is also supported by [3]. Depending on what the use case requires, a suitable attack model of the existing multitude of isolated solutions is used. If the use case changes or new questions arise, the applied model may have to be updated, or further models may have to be used, e.g., the MITRE ATT&CK Framework [4] (used for details of a specific adversary profile) in contrast to attack trees [5] (focusing on the system security on identifying security improvements). Using different models or the constant development of new models is time-consuming and causes security to be inconsistent and untraceable, which in turn may have a negative impact on the quality of security testing. To close this gap, [1] introduced the basic idea of a holistic attack modelling framework to support model-based security testing. The concept provides an adversary-based and target-based foundation to automate security testing on model level.

Penetration tests can be used in different domains [6][7]. We will show that the idea of penetration testing can be applied in the early stages of software development even if there is not yet a running code. Instead of testing an implemented system ADAM tests a model of the system.

Penetration testing is a common means to evaluate implemented security controls [6]. However, penetration testing usually only takes place in the late phases of software development, when it is already very expensive in comparison to security fixes in the early phases of software development. Also, the effectiveness and efficiency of penetration tests depend on the skills of the tester [6]. Vulnerabilities could go unnoticed, hence the coverage of penetration tests is unclear.

In contrast, a holistic attack model that provides automatable mechanisms for generating and simulating an adversary’s attacking procedure could be applied in the early design phase. For example, in the automotive domain in contrast to other domains like web-application programming, executable system models are already used in the engineering process (e.g., for simulations of complex assistance systems [2]). ADAM can reuse these models. Hence, it mitigates some of the shortcomings of penetration testing by applying the idea of penetration testing already at model level. The automatable test execution is more cost-efficient, and weaknesses can be

This work is supported by the German Federal Ministry of Education and Research (BMBF) under the KMUinnovative program (Grant No. 16KIS0946).

detected earlier than in common penetration testing. However, our approach is a complement for penetration tests. The execution of attacks on models does not replace a necessary penetration test on the implemented system in later phases. In summary, the contribution by ADAM is

- Foundation for early, automatable security testing on model level (adversary-driven, step-by-step attempts to reach a specific goal) before running code is available.
- Domain-agnostic, holistic attack information basis supporting automatable security testing on model level.
- Potential for reuse of the framework elements, e.g., modelled attacks.

The primary focus of this extended paper is to provide more details on the necessary components of the ADAM framework.

The rest of this paper is structured as follows: Section II discusses related work on attack modelling. Section III states the requirements for the ADAM framework as a holistic attack modelling framework for security testing. Section IV presents our approach to attack modelling. This is followed by the presentation of the four main components of ADAM, adversary model in Section V, target model in Section VI, attack base in Section VII and the attack modelling from the process perspective in Section VIII. Section IX shows the preliminary evaluation of the ADAM framework. Section X concludes the paper.

II. Related Work

Several adversary and attack models exist. Depending on the perspective of the attack, there are various modelling concepts [8].

The process modelling approach focuses on representing the attack based on phases. For example, the Lockheed Martin Cyber Kill Chain [9] defines an attack with seven phases that have to be passed through by the adversary. The kill chain model intends to model advanced persistent threats and malware behaviour. Hence, an attack is seen as a linear process, and it does not represent information about the attack surface that is provided for an adversary. Testing requires exploring multiple attack techniques, so bare process modelling approaches are typically not sufficient for testing. Another standard method is graph-based modelling that uses attack graphs to represent various attack opportunities. Kaynar [10] presents examples of this class of adversary and attack models in the domain network security.

A specific graph representation of attacks is the attack tree [5]. An attack tree focuses on the primary goal of an adversary. This primary objective represents the root of the attack tree, the elementary attack steps to are the leaves, and the various associated subgoals link these nodes. Existing attack trees can easily be reused or combined to form more comprehensive attack trees for threat and risk analysis. Attack trees incorporate multiple paths adversaries may take, but they do not include any characteristics of an adversary or about an adversary's decision on the next steps in an attack. Efficient testing requires an approach that also takes into consideration realistic assumptions about attack paths. Our work uses tree structures in combination with adversary modelling and target modelling to overcome the shortcomings of attack trees.

Classification modelling approaches model attacks on different abstraction levels. For example, the MITRE Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) framework [4] enables attack modelling based on the adversary's perspective. Tactics, techniques, and procedures define adversary behaviour. MITRE ATT&CK can be used both to derive behaviour-based adversary scenarios and to establish relevant adversary profiles for an implemented system. It is suitable for testing and verifying the security of an existing software product. However, the MITRE ATT&CK framework is not designed for use in the early design phase to support model-based security testing based on a specific adversary strategy. Our work closes this gap.

However, some domains have specific requirements, that have to be met, e.g., in the automotive domain. Thus, [11] shows another classification modelling approach regarding attacks. Ponikwar et al. [11] focus on defining realistic adversaries for vehicular networking applications. Defined adversary characteristics provide a template that can be used to categorise adversaries. In the context of various attack scenarios, adversaries with different levels of strength can be considered, depending on the specific, targeted vehicular networking applications. The adversary profiles in [11] can be easily reused. However, it is designed as a foundation for threat and risk analysis, and for making implementation decisions of security controls, rather than to support the security testing process. Thus, ADAM intends to take advantage of an adversary characteristic, focusing on the rational behaviour of adversaries in keeping with a goal-oriented security testing procedure.

Another considerable approach regarding classification shows [8]. Sommer, Dürrwang, and Kriesten [8] define a uniform taxonomy to describe automotive security attacks by classification. However, the attacks are just descriptions (tabular form, listing) by the taxonomy, which are not linked to a model of the targeted system for automatable security testing. Sommer, Dürrwang, and Kriesten already compare attacks with successful penetration tests that can consist of several steps. The consideration of attacks into its steps has the main advantage that the attack's detailed information is taken into account, e.g., to recombine steps for unknown attack paths. This requires a collection of possible attack paths. However, [8] provides no method to find attack paths or to recombine steps for new paths. ADAM aims to close this gap. It considers the adversary's decision-making, including his opportunities for achieving the prime goal, that results in the identification of attack paths. Sommer, Dürrwang, and Kriesten mention that in further work, attack paths could be automatically generated based on this taxonomy. This is also the intention of ADAM, where the model-based approach (not limited to the automotive domain) is followed instead of representing attacks as a description (tabular form, listing).

There are also combined approaches to attack aspects shown above. Adepu and Mathur [3] present unified adversary and attack models with a focus on both security and safety aspects in the context of cyber physical systems. The relevant system information is part of an attack domain model. However, Adepu and Mathur limit the proposed framework by not considering the characterisation of an adversary, e.g., the adversary's current knowledge about the target. However, realistic assumptions about an adversary are necessary for comprehensible modelling of the strategic and tactical attack actions of the adversary.

The Security Abstraction Model (SAM) [7] and ADversary VIew Security Evaluation (ADVISE) [12][13] are the works most similar to our approach.

Zoppelt and Kolagari [7] provide SAM, a foundation for the early analysis and design phases of software architecture development. It focuses on the integration of a security model into the common system model of the automotive domain, under consideration the associated security challenges. SAM presents a structured approach to identify and categorise attacks. The resulting overview of the overall attack situation assists the collaboration of various experts in the automotive domain with deriving security requirements and with the decision-making of security controls. However, SAM supports security by design and does not focus on automatable, early security testing by considering the strategical adversary's perspective of attack-decision-making. ADAM intends to automate early security testing (adversary-driven, step-by-step).

The protection of a software architecture always requires compromises for defenders. Not only defenders but also purposeful adversaries trade effort against benefit. Hence, ADVISE addresses the structured and goal-oriented procedure of an adversary [12][13]. This method is based on an executable security model on the system level to generate security metrics. It can already be applied in the design process of a system. The application of ADVISE is neither limited to a specific domain nor a certain level of detail. ADVISE is proposed for repeatable usage in security engineering to support the evaluation of system security. However, this security analysis method is not designed to automate security testing on model level based on a specific adversary. The adversary's decision function of ADVISE for the simulated attack procedure does not include the different aspects of designing and launching an attack, e.g., reconnaissance actions. In contrast, the main aspect of ADAM is the adversary-driven approach that drives the attack modelling step-by-step, including actions to launch an attack for the automation of security testing on model level.

III. Requirements

We propose ADAM, a concept for holistic attack modelling to support the model-based security testing by simulating the strategic actions of an adversary in terms of traceability. The general basis for the requirements engineering is [14] by interpreting security testing or attacking as business processes. Concerning the modelling of dynamic behaviour, there are analogies between model-based testing, attack strategies, and tactics and models for business processes [8][14][15]. Using models, complex scenarios can be simulated. ADAM is intended to be used to decide on the next steps during attacking activities that can be interpreted as test steps, e.g., modelling the structured use of existing hacking tools (or penetration testing tools from the tester's point of view). Hence, relevant requirements for the design of our approach can be derived from [14]:

a) Model-based: The expectation is that applying a model-based perspective to an attack presents a suitable basis for formalisation similar to the formalisation of the software development process in IT that came with the introduction of model-based software engineering [16].
b) Expressive: The purpose is to model as many attacks as possible by ADAM. A generic attack modelling framework should express all necessary information regarding attack, adversary, and target. As already shown in Section II, most attack models only incorporate certain aspects of an adversary and the target. The area of application is a relevant factor for the choice of an attack model. Using ADAM, this holistic attack model for multiple use cases can involve less effort than the application of several different attack modelling techniques, and it offers widespread usage.
c) Reusable: ADAM should consist of reusable components to reduce the time-consuming modelling of new attacks. For example, already modelled elements of ADAM should be reusable for as many different use cases as possible (e.g., change of target, adversary, or attack). The requirements a) model-based and b) expressive support this requirement reusable.
d) Systematic: The proposed ADAM framework should ensure a systematic and continuous (re-)use of attack information in as many phases as possible of the software engineering process. From the experience of the authors, today's software development often lacks such a systematic and continuous reuse of information about attacks.
e) Consistent: ADAM should be consistent. A consistent model can be verified and validated. Formalisation and automated tool support require a consistent model.
f) Visualisable: ADAM should use visual means to model attacks. An appropriate visual graphic representation of attacks facilitates readability and understandability, especially of complex attacks. Visual illustrations are more intuitive for humans than prose text [15]. The use of visual elements supports the formalisation as it is missing the ambiguity and inaccuracy of prose. The aim is to achieve a concise expressiveness of ADAM. The connection of the individual attack modelling components in ADAM should be easily identifiable, such that security can be consistently verified and software quality increases.
g) Understandable: Software engineers that are no security experts should be able to use ADAM throughout the software development process. Hence, ADAM should be understandable, easy to learn and uncomplicated to use. Complex models tend to be difficult to understand [16]. This disadvantage should be avoided.

IV. Design of the ADAM framework

The ADAM framework provides an adversary-based and target-based foundation to automate security testing on model level, including test case generation. Aspects of the adversary as well as aspects of the target under attack affect the attack path. Accordingly, these aspects will be considered for a holistic attack modelling to drive the security testing on the model level. We postulate the following scope for this work. Future work will probably leverage some of these restrictions:

- The ADAM framework focuses on attacks on modelled systems. Therefore, an adversary can theoretically influence any software-enabled technology in various ways [17]. Attacks targeting humans (e.g., Social Engineering) are out of scope.
- Our model is limited to adversaries that follow a rational goal [11]. Random attacks are out of the scope of this work. The ADAM framework is limited to goal-oriented adversaries.
- The attack modelling is limited to the information that is defined in an attack base. Therefore, no exploits can be considered in the systematic attack modelling by ADAM, which are not included in the attack base. But by updates of a suitable attack base, the Zero-Day exploits can be quickly available for security tests.
- The focus of this paper is to identify the necessary conceptional elements for the suitable, holistic ADAM framework and giving an idea of how it can be used. Completeness, detailed specification and implementation of these elements are out of the scope of this paper.

A. Overview

By means of ADAM, we associate each attack with an adversary and the system under attack (target). An adversary plans, develops, and executes attacks against the target by using specific resources. The adversary attempts actions step-by-step in a certain order to reach the primary adversary goal. The target may provide one or more access points (AcPs). An AcP is a point of the target, that provides the adversary with the opportunity of executing attacks [8][18] (see Section VI). After each step, the adversary can gain new information about the target and expands his perspective in this regard. He has certain AcPs at his disposal. Different aspects, such as his skills (e.g., hacking skills), knowledge of the target domain (e.g., the ISO 26021 [19] for road vehicles), or his connection to the target (e.g., only remote) reduce the selection of available AcPs for the current attack attempt. Both for the initialisation of the ADAM framework (see the "for" section of the loop in Figure 1) and its application (see the "while" and "do" section of the loop in Figure 1), it requires this basis of the content in the context of attacks.

Figure 1 gives an overview of the ADAM framework. The framework consists of the adversary model, target model, and attack model. The adversary model characterises a specific adversary. Each adversary is defined by a set of descriptive attributes, the goal of his attack, and his current perspective of the target (called adversary perspective model in Figure 1). Based on [13], a modelled adversary is called adversary profile, consisting of the defined goal, initial perspective and determined characteristic. The target model simulates the system under attack and all necessary associated components of the environment to simulate the attack path of the adversary. Based on [8], the attack model is called attack base and includes the technical aspects, potential ways and means to attack, e.g., the possible actions with the associated target.

ADAM links all the necessary information of these models to consider the attack from a process perspective (see "Attack Modelling" in Figure 1). At the beginning (see "Initialise Elements" in Figure 1), the necessary elements (adversary model, attack model and target model) have to be initialised. To achieve the prime adversary goal, the so-called elementary attack iteration (EAI) is iteratively called in the attack simulation. The EAI consists of defined steps. With each iteration, the adversary must decide on which action to attempt next that is goal-directed. For example, depending on the current adversary's perspective, and skills, he attempts to exploit the target by an available AcP.

For this purpose, all necessary information from the attack base and adversary profile is used. The attack simulation on the target model provides the effect of the attack on the target. The use of a target model allows executing attacks on systems that do not yet exist. Each iteration step ends with an update of the adversary profile. For example, when the adversary reached his primary goal, the simulation terminates. Otherwise, see "Attack continues" in Figure 1, e.g., the next iteration starts. The outcome of the ADAM framework in the context of an attack simulation is an attack path. Based on [8][20], an attack path is one action or a set of sequential actions taken by an adversary for the adversary's goal achievement. In the simplest case, the adversary can theoretically achieve the prime goal by selecting and executing one action, i.e., the attack path consists of one action. Otherwise, the adversary's goal achievement is composed of several actions, i.e., the attack path is a sequence of several actions in a certain order.

ADAM can be used for model-based security testing. It provides a structured approach to simulate the strategical behaviour of a given adversary to attack a particular target. It takes into consideration the properties of the adversary as well as the perspective the adversary has of the target at a given time. The systematic attack modelling process of ADAM identifies actions that can be interpreted as test steps on model level [8]. Thus, ADAM provides a basis to automate security testing on model level, including test case generation. Hence, ADAM supports holistic model-based security testing in the early phases of the software development process (no running code available) [1]. Besides, e.g., the generated attack path by ADAM can support as guidance for tests in later phases of the software development lifecycle (running code available) [1].

V. Adversary Model

The adversary model provides the goal-oriented adversary for attack modelling. In the context of ADAM, the following

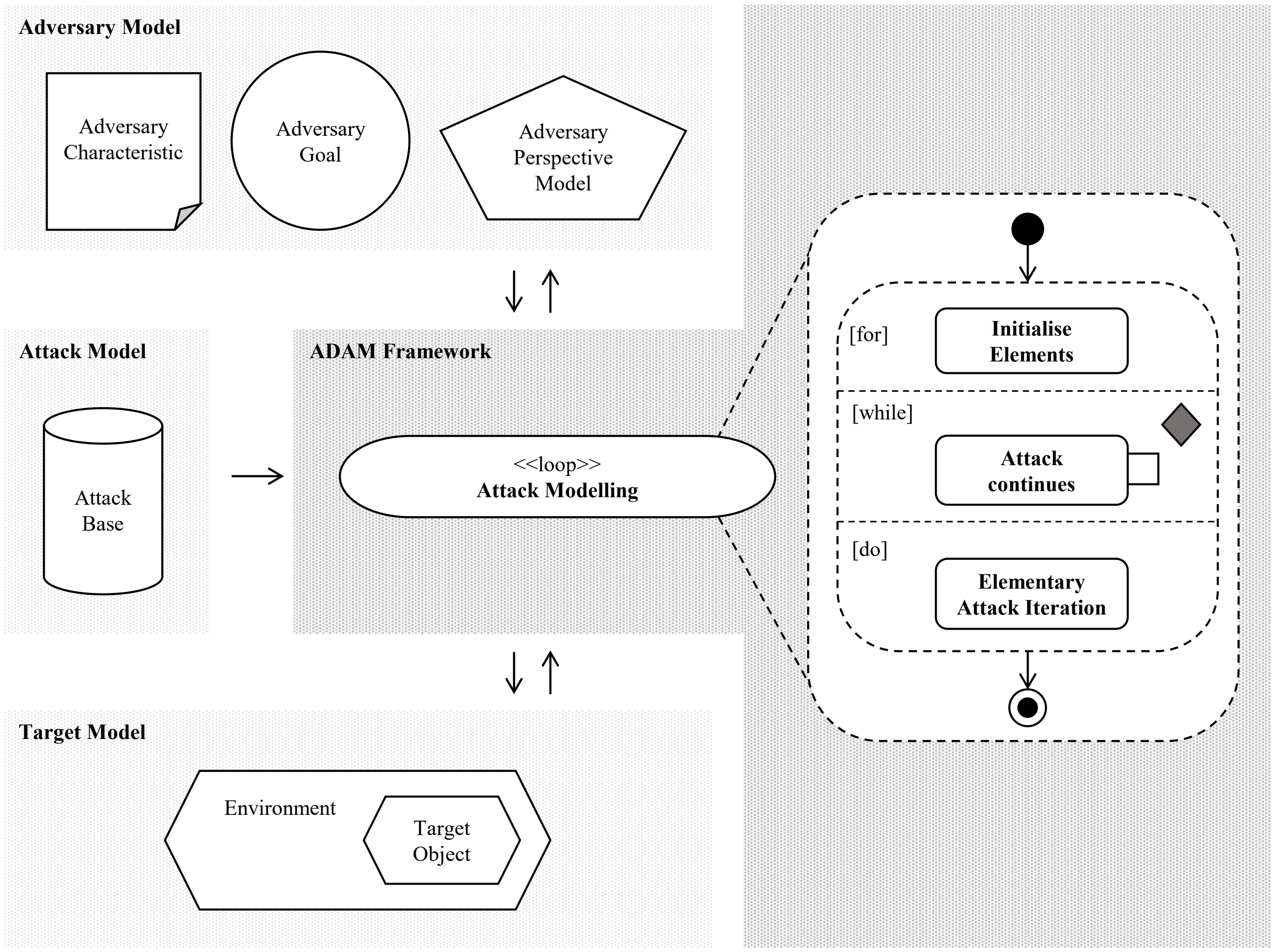

Fig. 1. Conceptual context of the ADAM framework based on [1].

requirements arise for the adversary model:

- An adversary should be modelled using defined, qualitative characteristics so that various realistic adversaries can be considered, independent of the target domain.
- The adversary model should state the prime goal of an adversary. The adversary goal is necessary to decide if the attack modelling is finished because the adversary goal is achieved. It is used for modelling the goal-oriented decision-making of a given adversary.
- The adversary model should represent the dynamic adversary's perspective of the targeted system, i.e., all information the adversary currently knows about the target that can be used to attack. This consideration enables representing an attack step-by-step, based on the adversary's preliminary, changing view of the target.
- A modelled adversary should be used for the process of attack modelling to generate the attack path.

Therefore, the adversary model consists of the three elements described in the following: adversary characteristic, adversary goal and the adversary perspective model [1] (see Figure 1).

A. Adversary Characteristic

The adversary is characterised by static attributes. This means, that the characteristic is initialised at the setup of the attack modelling (see "Initialise Elements" in Figure 1) and is not changed during its further use in modelling. Changing the characteristic means defining a different (type of) adversary for another simulation scenario. The total attributes represent the power of the modelled adversary and affect the adversary's attack path. It is assumed, e.g., an adversary with the expertise of general computer sciences, as the only skill, will search alternatives to actions of sophisticated hacking, unlike an adversary with professional security and hacking skills.

Figure 2 shows an exemplary collection of attributes for the adversary characteristic in support of ADAM. It can be expanded by arbitrary, which become necessary for the adversary's decision-making in the attack modelling regarding of future work, e.g., with the application of ADAM in further domains.

A unique ID (see characteristicID in Figure 2) is necessary to provide the individual identification of the modelled adver-

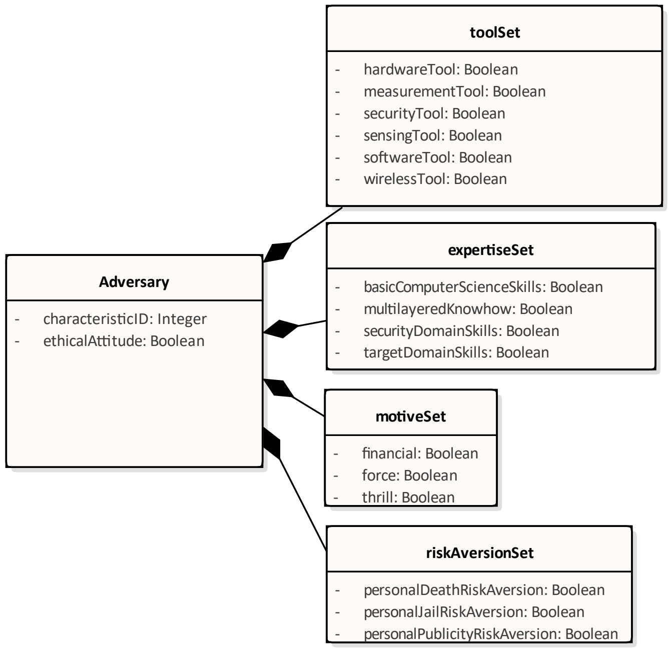
toolSet
- hardwareTool: Boolean
- measurementTool: Boolean
- securityTool: Boolean
- sensingTool: Boolean
- softwareTool: Boolean
- wirelessTool: Boolean
expertiseSet
- basicComputerScienceSkills: Boolean
- multilayeredKnowhow: Boolean
- securityDomainSkills: Boolean
- targetDomainSkills: Boolean
Adversary
- characteristicID: Integer
- ethicalAttitude: Boolean
motiveSet
- financial: Boolean
- force: Boolean
- thrill: Boolean
riskAversionSet
- personalDeathRiskAversion: Boolean
- personalJailRiskAversion: Boolean
- personalPublicityRiskAversion: Boolean

Fig. 2. Attributes for the Adversary Characteristic based on [8][12][13][21].

sary characteristic and thus its deliberate reuse in combination with another prime goal or initial adversary's perspective about the target.

The attribute ethicalAttitude in Figure 2 defines whether the adversary rejects personal injury as a matter of principle on the way to achieving his ultimate goal. For example, it is assumed that a penetration tester or hobbyist will not take actions that will clearly harm human lives to achieve their prime goal. Instead, these adversaries will seek other alternatives or give up. Thus, ethicalAttribute can be used for the adversary's decision-making on the next action of attacking out of available actions.

The adversary's power depends on tools. The adversary is able to choose the attacking actions associated with them. Adapted from [8] we use the attributes securityTool, softwareTool, hardwareTool, sensingTool, measurementTool and wirelessTool (see Figure 2). The attribute securityTool defines whether the adversary owns security tools, e.g., a reverse engineering tool to be able to choose information gathering actions. The attribute hardwareTool defines whether the adversary owns hardware tools, e.g., a laptop to be able to collaborate with different other tools. The attribute softwareTool defines whether the adversary owns software tools, e.g., a debugger to be able to choose information gathering actions. The attribute wirelessTool defines whether the adversary owns wireless tools, e.g., a cellular tool to be able to take remote actions. The attribute sensingTool defines whether the adversary owns sensing tools, e.g., a radar tool to provide input for systems that expect radar data. The attribute measurementTool defines whether the adversary owns measurement tools, e.g., a logic analyser to be able to choose information gathering actions. This insight shows that different attacking actions require different tools.

In addition to the consideration of tools, an adversary can be endowed with helpful skills. A distinction is made between the adversary's skills (as a statically determined resource) and the current perspective of the target that the adversary (dynamically) expands during the attack until the achievement of the adversary's goal. Based on [12][13], we distinguish the skills between basicComputerScienceSkills, securityDomainSkills, targetDomainSkills and the attribute multilayeredKnowhow (see Figure 2). By means of these attributes, the adversary's capabilities are determined, which affects his attack path. For example, just because the targeted system provides the opportunity of, e.g., actions regarding timing attacks, an adversary without the necessary skills will take alternative actions. The attribute basicComputerScienceSkills defines whether the adversary has basic expertise in computer science, for example, expertise in the internet protocol family. The attribute securityDomainSkills defines whether the adversary has specific expertise in the domain of security and hacking, e.g., as a professional penetration tester [6]. The attribute targetDomainSkills defines whether the adversary has specific expertise in the target domain, for example, in the interface technology of controller area network (CAN) in the automotive domain [8]. Besides the various areas of expertise, the attribute multilayeredKnowhow defines, whether the adversary has extensive knowledge on various topics that can be required for the achievement of time-consuming or complex attacks [5][11]. For example, a group of experts of different topics can be defined as an adversary compared to an individual adversary who is limited to its skills. It is used to distinguish organised groups with different experts as an adversary in ADAM to less broadly based groups or individuals. It is assumed that organised groups, like advanced persistent threat groups, have access to a wide range of different experts and are well networked with each other in the group.

The reason for a particular behaviour is determined by the motive [21]. The adversary's motives affect his decisions during an attack, independent of the current prime goal. Based on [8], and adjusted for the adversary model, the motives of the adversary are defined by the following set of attributes: force, thrill and financial (see Figure 2). The attribute force defines whether the adversary's willingness to act is driven by the purpose of influencing, e.g., as a state actor [22]. The attribute thrill defines that the adversary's willingness to act is based on thrill, e.g., as a hobbyist or script kiddie [22]. The attribute financial defines whether the adversary has financial reasons that purposefully affect his willingness to act, e.g., as a criminal, or as a paid professional penetration tester [22].

The adversary's decisions are determined by his risk aversion [5]. Each adversary is willing to take a certain risk to achieve the individual ultimate goal. During the ongoing attack, in step 3 (see Section VIII) of the EAI, the adversary has to choose one action out of diverse options, whereby each action has a specific risk. If an action is associated with a risk that the adversary has an aversion to, it is assumed that the adversary prefers alternative actions or surrenders. Based on [5], the set of attributes for risk aversion consists of personalPublicityRiskAversion, personalJailRiskAversion and personalDeathRiskAversion (see Figure 2). The attribute personalPublicityRiskAversion defines whether the adversary has an aversion to the risk of publicity, e.g., a current employee [22] of the business of the targeted object (not to be confused with an angry or former employee). The attribute personalJailRiskAversion defines whether the adversary has an aversion to the risk of jail time, e.g., a hobbyist [22]. The attribute personalDeathRiskAversion defines whether the adversary has an aversion to the risk of death, e.g., an activist [22] who wants to call attention to a grievance.

This collection of attributes in Figure 2 exemplifies that various realistic adversaries can be presented using the adversary characteristic in the ADAM framework. The adversary can be characterised independently of the specific attack goal, and the current adversary's perspective of the target. Once an adversary is modelled by these attributes, the adversary characteristic can be reused in the context of, e.g., different adversary goals, initial adversary's perspective, targets, or new attack possibilities.

B. Adversary Goal

Another component of adversary modelling is the adversary goal (see Figure 1). The adversary goal is the primary goal to decide if the attack modelling is finished because this goal is achieved. It is used for modelling the goal-oriented decision-making of a given adversary. For example, the primary goal of an adversary may be the gain of financial data from a financial data transfer system. As well as the adversary characteristic, the adversary goal is static. This means, that the ultimate goal is initialised at the setup of ADAM (see "Initialise Elements" in Figure 1) and is not changed during its further use in the ADAM framework.

The adversary goal determines "what" is the attack goal. "How" the adversary goal is attempted to be achieved, by a certain sequence of actions (attack path) is only apparent from the application of ADAM (see attack modelling, Section VIII). Hence, the adversary's goal is used to derive the attack path of an adversary during an attack simulation. In the example above, the goal derives attractive data stores as target objects. The adversary tries to navigate from any AcPs available to the adversary to these data stores.

The following attributes illustrate an exemplary adversary goal modelling in support of the ADAM framework. A unique ID (goalID) is necessary to provide the individual identification of the modelled adversary goal and thus its deliberately reuse in combination with another adversary characteristic or initial adversary's perspective about the target. The attribute goalTarget defines the concrete target object of the adversary goal, e.g., a certain data set, data storage, interface, subfunction, function, or a complete system [8]. The attribute goalMotivation defines what is to be achieved concerning the goalTarget [8]. The goalMotivation always refers to the goalTarget. Based on [23], the goalMotivation represents the urge to act to achieve something by choosing useful actions. The following motivations can be distinguished to specify the adversary goal:

- Gaining (regarding information, financial or material) [7][8][24]
- Obtaining access [24]
- Affecting
- Controlling (in sense of the wittingly and goal-oriented utilisation)
- Modification or tuning (in sense of optimisation) [7][8]
- Damaging to property [8][24]

The attribute goalUndercover defines whether the adversary goal is associated with an as covert and unnoticed approach as possible [25]. It is assumed, e.g., to achieve the prime goal with a focus on industrial espionage, an adversary prefers to act as unnoticed and covertly as possible, i.e., the focus during the attack is also on actions to cover traces. The attribute goalAggressive defines whether the adversary goal is associated with an aggressive approach. For example, to achieve the ultimate goal with a focus on only finding vulnerabilities of a certain system (i.e., being not aggressive), an adversary prefers to act very carefully, not to cause any damage to the system. This scenario can be assigned to a typical penetration testing use case [25]. Whereas it is assumed, that, e.g., to achieve a prime goal with a focus on causing as much damage as possible (i.e., being aggressive), an adversary prefers choosing actions that cause harmful consequences [11].

These goal attributes exemplify how the adversary goal can be represented for the ADAM framework. The modelled adversary goal can be (re-)used in the context of, e.g., different adversary characteristics, adversary's perspective, targets, or new attack possibilities. The goal drives the step-by-step decision-making of a given adversary in ADAM and thus also affects the test case generation for model-based security testing.

C. Adversary Perspective Model

The third component of the adversary model for the ADAM framework is the adversary perspective model. It represents the adversary's state in the context of the progressing attack at a given time [12][13], from the beginning to the end of the attack simulation.

At the beginning of the attack modelling, the adversary perspective model shows the adversary's initial state of knowledge of the targeted object and its environment, including AcPs. An adversary can have already gained knowledge, e.g., from public sources, such as forums, manuals, standards, or as a result of previously executed attacks. For example, regarding the automotive domain, an adversary has already remote access to the CAN bus on a vehicle over cellular AcPs, which is demonstrated in [26]. Which initial adversary perspective is defined depends on what is the focus of the user of ADAM. If a tester uses ADAM to test a single function within a system, it can be assumed that the modelled adversary has already access to the system (defined as initial adversary perspective), in contrast to the test case where an entire system is to be attacked. Thus, ADAM can influence the security test case generation on model level with the help of the adversary perspective model. The possibility of individually defining the initial adversary situation enables a flexible setting of the focus of the attack modelling. For example, if the adversary already has remote access to the victim's laptop, the focus is on actions regarding access on credentials instead of actions to gain access on the laptop anyway.

During the attack simulation, each EAI increases the adversary's knowledge, thus updating the adversary perspective model. For example, the adversary may get access to further AcPs after the first attack iteration, which he can use for an attack attempt in the next attack iteration. As long as the adversary's current knowledge is not sufficient to achieve his primary goal, the adversary tries to expand his knowledge in the appropriate direction through further attack attempts. This means that not only the adversary's initial perspective (by the adversary perspective model) but also each successive state during the attack attempt is relevant for attack modelling with ADAM. The adversary has a certain perspective of the target at a given time. Using the adversary perspective model, the results of the attack by the adversary is logged. Regardless of whether the simulated action was successful for the adversary or not, each result is a new takeaway. Hence, the initial perspective of the adversary is updated with each selected and simulated action during the attack simulation. It is assumed, that the adversary may get a wrong idea of the target. For example, the adversary could be fooled by means of security controls of the target. Thus, the adversary perspective model may keep incomplete or blurry details on the target. It represents the current, preliminary view an adversary usually has on the target. In contrast to the adversary perspective model, the target model (see Section VI) holds only correct information.

The adversary perspective model is necessary for the adversary-driven procedure of the step-by-step attack modelling. Hence, ADAM provides traceable attack attempts for the support of security testing automation on model level.

VI. Target Model

During an attack simulation with the ADAM framework, an adversary has to decide his next action for the achievement of his ultimate goal. Once the adversary has chosen an action, it is applied to the target model.

The target model represents the target, composed of both the target object and the environment of it, which an adversary wants to exploit and/or utilise for his attack attempt to achieve the ultimate goal. The target object corresponds to the concrete object, with which an adversary wants to act according to his ultimate goal (see the attribute goalTarget of the adversary goal). For example, the target object in the target model can be a certain data set that the adversary wants to delete. Depending on what is to be modelled or tested with a defined adversary, the target object is, e.g., data set, interface, subfunction, function or a complete system. The environment contains all necessary information to simulate the generated attack path by the EAI to get down to the target object, e.g., by means of exploited AcPs. From the set of all existing interfaces of the target object and its environment under consideration, the adversary has particular points available at a certain time, called AcPs. An AcP, based on [8][18], is an available point to the modelled adversary at the current time and provides him unintended access or unintended information disclosure. The AcP is either the target object or part of the surrounding environment of the target object. During an attack iteration, an adversary can utilise available AcPs to achieve his prime goal.

This outcome of the applied action to the target model influence the adversary's profile. For example, the output of a simulated action is a credential that is accordingly included in the current adversary perspective model of ADAM.

The attack simulation on a target model enables to show the attack and its effect on the target on model level. This implies that attacks can be tested in the early phases of the development of a product (target) that is not yet implemented. The target model simulates the target object and all relevant aspects of the environment of the target object that can be used for the attack attempts by the adversary.

VII. Attack Base

The attack base contains all necessary technical details for the attack modelling, e.g., all actions that an adversary could possibly execute during the attack simulation. Various works (see Section II) exist with a lot of information about known attacks. For example, based on known, past attacks [27][28], or from associated reports and analysis [29][30], actions can be derived to fill the attack base.

An action is the elementary element of the attack from the technical perspective. Based on [8], the action represents an elementary attack (e.g., modify the calibration update) that can be one specific step of a composite attack (e.g., controlling a certain electronic control unit (ECU)). As an intermediate step of a more comprehensive attack, the step does not necessarily have to be a stand-alone, typical attack action. For example, using search engines can be an action of the composite attack reconnaissance.

Several actions can be pooled to an AcP, which is also provided as information in the attack base. E.g., based on [28], the on-board diagnostics (OBD) as AcP (of the automotive domain) is associated with the following actions:

- Action of capturing CAN messages
- Action of replay CAN message
- Action of sending the standard command for disabling the CAN communication
- Action of firmware extraction of the telematics ECU
- Action of sending the standard command for flashing the telematics ECU

The actions can be linked to each other utilising preconditions and postconditions [8][10]. The action of capturing CAN messages has preconditions, e.g., that access to the CAN bus is given and captured CAN messages as a postcondition, which in turn acts as a precondition for further actions in this regard.

The attack base includes the technical aspects of attacks. For the holistic attack consideration, the handling of this foundation of technical attack information is described in Section VIII. The attack modelling loop is responsible for selecting specific actions, step-by-step, from all available actions in the attack base.

VIII. Attack Modelling

The ADAM framework provides various perspectives of an attack as shown in Figure 3.

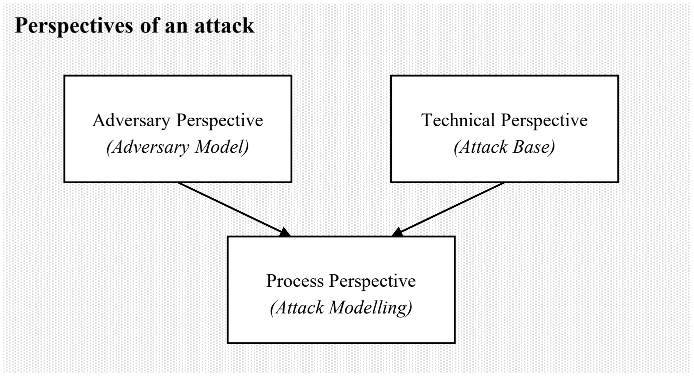

Fig. 3. Linking of the considered perspectives of an attack by the ADAM framework.

The adversary model provides the goal-oriented adversary perspective of attacks (see Section V). This perspective represents the scope of the attack modelling from the adversary's point of view, i.e., the adversary characteristic, the adversary goal and the adversary's current, dynamic knowledge about the target (adversary perspective model).

The technical perspective of attacks focuses on the actions provided by the attack base (see Section VII), which can lead to the adversary's goal achievement in proper order.

The process perspective focuses on the execution of an attack. By means of defined steps, the technical perspective of an attack is considered in a structured way, i.e., the attack modelling by the ADAM framework brings together both the goal-oriented adversary perspective of attacks and the technical perspective of attacks, as outlined in Figure 3.

The attack modelling loop simulates each attack. First, the necessary elements, i.e., the adversary model, attack model, and target model, have to be initialised (see "Initialise Elements" in Figure 1), which provide input for the attack modelling loop. As a result, a specific adversary profile, target model and attack base are available. Next, their content will be used in a structured way according to the attack modelling loop.

As long as the attack continues, for example, the adversary goal is not achieved and he does not surrender, the EAI will be called iteratively. We call such an iteration an elementary attack iteration (EAI), as shown in Figure 1, as it constitutes the smallest attack unit possible from a process perspective of attacks. Each EAI includes the five steps:

1) Identify available AcPs
2) Select an AcP
3) Select an action
4) Probe the target
5) Update the adversary profile

Within each iteration, the modelled adversary decides of which action to attempt next for achieving his primary goal. Thus, the adversary first chooses an AcP over which to execute the attack attempt. For this purpose, e.g., information about his current perspective of the target object and its environment is used (provided by the adversary perspective model). In addition to the prerequisites concerning the target (from the adversary's perspective), aspects regarding the adversary (characteristic) can also be presupposed, e.g., necessary tools and skills (see Subsection V-A) to select a certain AcP, and next, a specific action.

For example, the adversary wants to modify a function in a vehicle (adversary goal). Thus, he selects the AcP OBD connector because he has the necessary skills and tools for the automotive target domain and the OBD connector is target-aimed to the adversary goal. From all available actions associated with the AcP, the adversary chooses one goal-directed action based on the adversary's profile. In the example above, the adversary selects an action of code injection, because the adversary already knows, the OBD connector is active and vulnerable, and actions of override did not work (e.g., as results of a previous attack action, represented in the adversary perspective model) and the action is in compliance with the adversary characteristic. For example, he has targetDomainSkills and basicComputerScienceSkills as appropriate skills, and, e.g., due to the acceptance of the risk of death, the adversary selects and executes an action of code injection. The chosen action will be applied to the target model that provides the effect of the (attack) action on model level. Finally, the adversary profile will be updated, e.g., the adversary perspective will be expanded by the result of the action (successful, or not) and, as appropriate, by the recent AcPs that has been made available for the next EAI. To achieve the adversary's primary goal, usually, several elementary iterations are necessary. The chosen actions by the adversary profile during the attack modelling, by the sequential selection, provide the attack path as the output of the applied ADAM framework. These selected actions can be interpreted as test steps on model level [8]. Hence, ADAM supports not only a foundation to automate model-based security tests but the whole security testing on model level, including test case generation. Besides, the generated attack paths can give hints that are worth considering in other tests, like penetration tests in later phases of the software development process.

IX. Preliminary Evaluation

In this section, we evaluate if the ADAM framework meets the requirements of Section III and we give an idea of how ADAM can be integrated to support security testing. We evaluated ADAM under the following restrictions (future work will leverage some of these restrictions):

- The application of the modelling is limited in each case to one attack iteration.
- The attack scenarios under consideration focus on the first actions of an attack, comparable to reconnaissance of the Lockheed Martin Cyber Kill Chain [9].
- The proposed ADAM framework is applied to two significant attack scenarios by way of example. The first example incorporates vulnerabilities of the Open Web Application Security Project (OWASP) Top Ten 2017 [31], hence is highly relevant in the domain of web application. In contrast, the second example stems from the automotive domain. We use the idea of UML class, object and activity diagrams [32] and attack tree [5] for our examples. We choose UML as it is common in many relevant application domains.
- The decision on which adversaries are relevant for modelling with the help of ADAM is not the focus of this paper. For the evaluation, we use the widespread method of threat and risk analysis in the context of the research project MASSiF. As a consequence, the relevant type of adversary is organised criminals.

A. Evaluation Criteria

The following criteria were identified for the evaluation:

1) Model-based: The criterion refers to the extent to which the ADAM framework is based on a model.
2) Relevant attacks: The criterion refers to the extent to which relevant attacks can be modelled using ADAM.
3) Application domain independence: The criterion refers to the ability to model different attacks independently of the application domain.
4) Reusable elements: The criterion refers to the extent to which the modelled contents and elements of ADAM can be easily reused in conjunction with other attack scenarios.
5) Systematic structures: The criterion refers to the extent to which there is a systematic approach to the structure and procedure of the proposed attack modelling concept so that an attack can be modelled in a comprehensible and repeatable way.
6) Visual elements: The criterion refers to the extent to which the ADAM framework has graphic elements or can be illustrated visually at a glance.

A consistent model is a requirement for the use of automatism [14] and thus, a suitable basis for supporting the automation of security testing on model level. Therefore, a detailed specification of the EAI, including a proper syntax and semantic for the necessary elements have to be defined. The specification of the individual elements of the proposed concept is out of the scope of this paper. Therefore, we omit the evaluation of the model consistency. Moreover, the specification of the EAI is required to make appropriate statements about the understandability. The understandability of a model helps to evaluate its usefulness. Also, we omit the evaluation of the requirement understandability. We will survey relevant stakeholders to assess the understandability of the model in the further course of the still running research project MASSiF.

B. Exemplary Application of ADAM

We iterated through the proposed ADAM framework, based on two exemplary attack scenarios, and integrated it in an exemplary way. For the sake of briefness, we only present an extract of exciting findings in the following Figure 6, Figure 8 and Figure 9, concerning the elementary attack step 3 of the EAI (see Section VIII).

In the first scenario, we model a criminal adversary who wants to steal an identity on a social media platform. This scenario incorporates attacks from the OWASP Top Ten 2017 [31]. The characteristic of the criminal is shown in Figure 4. The defined adversary accepts injury to people as a matter of principle on the way to achieving the ultimate goal (see “ethicalAttitude=false” in Figure 4). The adversary has an aversion to actions that are associated with a risk of the adversary’s death (see “personalDeathRiskAversion=true” in Figure 4). For example, Ponikwar et al. [11] mention criminals who act goal-oriented, like a business. Hence, the criminal is assigned with the financial and force motive. He is well equipped with skills and tools. The adversary is defined as an organised criminal group, which is reflected in the attribute “multilayeredKnowhow=true” in Figure 4.

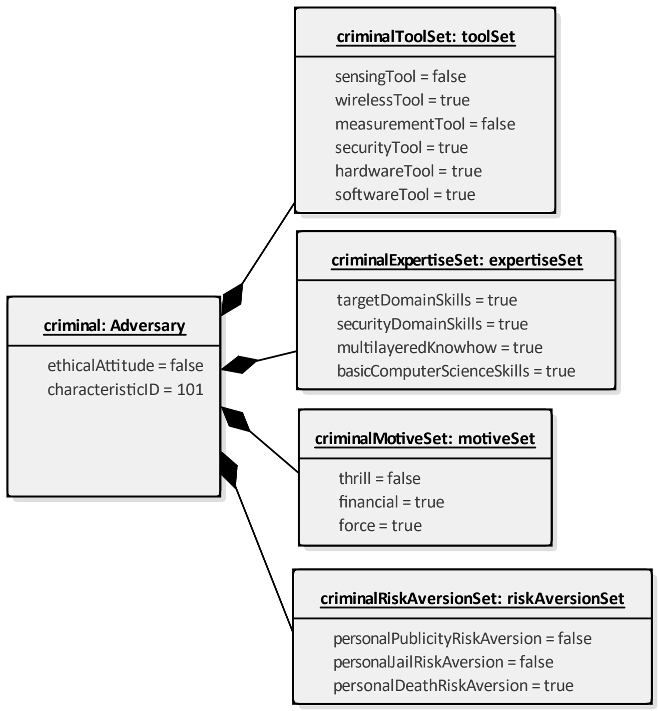
criminalToolSet: toolSet
sensingTool = false
wirelessTool = true
measurementTool = false
securityTool = true
hardwareTool = true
softwareTool = true
criminalExpertiseSet: expertiseSet
targetDomainSkills = true
securityDomainSkills = true
multilayeredKnowhow = true
basicComputerScienceSkills = true
criminal: Adversary
ethicalAttitude = false
characteristicID = 101
criminalMotiveSet: motiveSet
thrill = false
financial = true
force = true
criminalRiskAversionSet: riskAversionSet
personalPublicityRiskAversion = false
personalJailRiskAversion = false
personalDeathRiskAversion = true

Fig. 4. Characteristic of the modelled criminal as an adversary.

Figure 5 describes the goal of the criminal for the first scenario. In this scenario, the criminal adversary not only wants to gain identity information but also aims at personating someone else on a social media platform. In doing so, the goal is linked to acting as unnoticed as possible and not destroying the function of the social media platform.

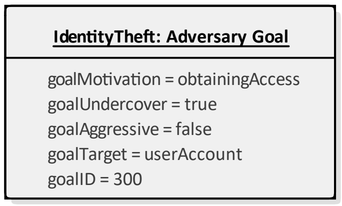

Fig. 5. Identity theft as an adversary goal.

The alternative actions for the criminal (regarding identity theft as the ultimate goal) from the AcP user input field of a web application are shown in Figure 6. Using tree structures, the result of the criminal's decision is visualised. The adversary chose an action in the context of Credential Stuffing.

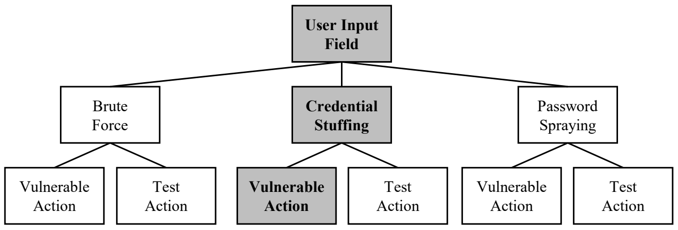

Fig. 6. Selected action based on the AcP "User Input Field".

The second scenario is taken from the research project MASSiF. The adversary characteristic from the first scenario is reused. In the second scenario, the defined criminal adversary with the characteristic as shown in Figure 4, attempts to disrupt the function of an ECU in a vehicle. In this use case, the "targetDomainSkills=true" refers to the automotive domain. For example, it is assumed the criminal knows the relevant standards of the automotive, such as the ISO 26021 ("Road vehicles – End-of-life activation of on-board pyrotechnic devices") [19]. For the goal achievement, the defined criminal could start an attempt by utilising actions based on his knowledge about the standards of automotive.

Figure 7 represents the adversary goal for the second scenario. The criminal wants to disturb an ECU functionality. Within the scope of this goal, the criminal does not care whether the victim notices the attack attempts (see "goalUndercover=false"). Attribute "goalAggressive=true" shows that the adversary accepts that the goal achievement may entail personal injuries.

Figure 8 illustrates the alternative actions for the criminal (with the prime goal of ECU functionality disturbance) from the standardised interface OBD connector in a vehicle. At the very beginning of the attack, the criminal needs information about the target. Thus, the adversary selected an action regarding extraction information using the OBD connector.

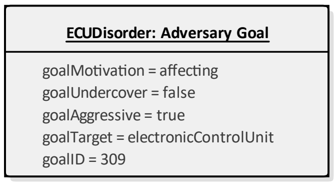
ECUDisorder: Adversary Goal
goalMotivation = affecting
goalUndercover = false
goalAggressive = true
goalTarget = electronicControlUnit
goalID = 309

Fig. 7. ECU disturbance as adversary goal.

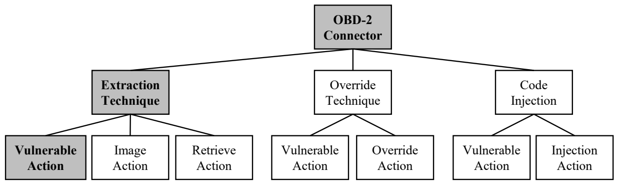

Fig. 8. Selected action based on the AcP "OBD connector".

The respective application of the actions in the scenarios leads to new information for the adversary, e.g., the specific AcP is vulnerable. During the next iteration, the adversary can select the next action based on the new information the adversary gained from the previous attack iteration.

In the context of the research project MASSiF, we integrated ADAM in an exemplary way, as shown in Figure 9. As MASSiF focuses on the automotive domain a model-in-the-loop (MiL) simulation environment is used to integrate. A MiL is common for safety testing of complex assistance systems in the early development phases [2]. In the MiL test, an executable model is the test object. The test object and an associated environment are integrated into a control loop to represent and test their interactions as realistically as possible [2].

The ADAM framework consists of the adversary and attack model, which are connected with the target model through the attack modelling loop (see Figure 1). The target model consists of the target object and its surrounding environment. Based on [15], a simulated environment model displays the boundary conditions and required interfaces of the environment for the system model. In this case, in Figure 9, the system model is the target object, e.g., the defined adversary wants to disturb a certain system, modelled as a system model.

The simulated environment provides required input data for the simulation of the system. For example, a system model processes the required input data and resends some output to another component in the simulated environment. But instead of a direct link between the simulated environment and the system model (target object), the data flows through the ADAM framework, as shown in Figure 9. During an EAI of the attack modelling loop, the defined adversary may or may not have certain possibilities (based on the adversary perspective model as well as his characteristic, e.g., skills) to take actions to affect the data flow from the simulated environment to the target object (system model) to achieve his ultimate attack goal. For example, the adversary chooses an action to replay manipulated sensor data to the system model (target object)

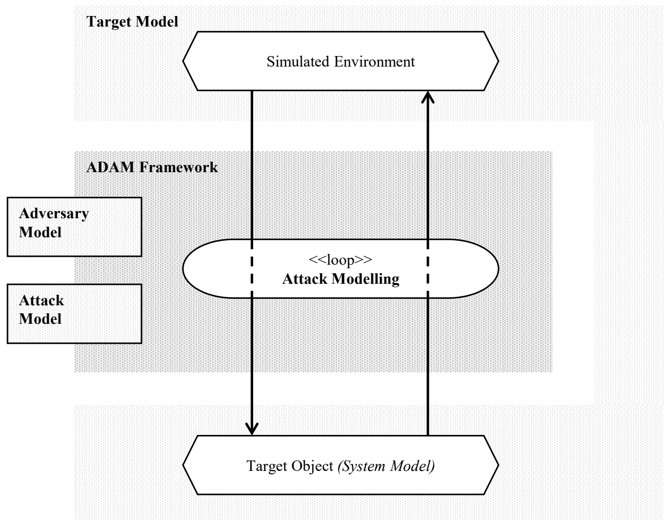

Fig. 9. Conceptional idea for the integration of the ADAM framework.

because he has already met the associated preconditions (such as successful access) as a result of previous actions. To get the result of the selected action, the data flow from the system model (target object) to the environment will be read out (as part of the EAI step 5 "Update the adversary profile", see Section VIII).

C. Interpretation and Discussion

Using the example of tree structures and UML class, object and activity diagrams, model-based elements can be used systematically for attack modelling. The criterion model-based can be confirmed insofar as ADAM provides a suitable foundation for different modelling approaches.

The criterion relevant attacks can be confirmed to the extent that we were able to model two representative examples from very different application domains. It is assumed, that the attacks have already been integrated as actions in the attack base to be considered using the ADAM framework. In our opinion, the proposed concept provides a suitable basis for modelling attacks, independent of the domain. Likewise, utilising the adversary model, different adversaries can be represented, for example, the adversary characteristic is extendable accordingly. Consequently, the ADAM framework accomplishes the criterion application domain independence. However, it is still an open question to what extent the specific characteristics of individual domains must or can be captured.

The defined actions of the attack base, as well as the EAI process itself, are exemplary representatives of reusable elements. Likewise, and regarding the criterion reusable elements, a modelled adversary (adversary profile) and its components (e.g., adversary goal), can be reused repeatedly and adapted, for example, with a different goal and/or initial adversary's perspective about the target for the attack modelling.

The EAI of the ADAM framework represents the basic, systematic, adversary-driven guideline for attack modelling from the process perspective of attacks. Likewise, the adversary model, target model and attack model represent a suitable foundation for a systematic deployment, representation and reuse of attack information. In this respect, the criterion systematic structures is accomplished.

The exemplary use of tree structures, UML activity, class and object diagrams shows that the attack modelling concept provides a suitable basis for the integration of graphical model elements. In this respect, the criterion of visual elements is accomplished.

The attributes of the adversary characteristic in their current form are not yet sufficient for the attack modelling by the ADAM framework. The Boolean expression, e.g., for the tool set and skill set of the adversary characteristic is not sufficient to filter out an action from a set of similar actions. For example, the actions of vulnerability scanning, fuzzing, and reverse engineering all require security tools. If securityTool (see adversary characteristic in Section V) is only expressed by a Boolean value, the filtering does not reduce the set of these available actions to one action. Another example is the attribute set of risk aversion (see adversary characteristic in Section V). For example, a dependency on the legal situation of the respective country is still important for choosing actions according to the personalJailRiskAversion.

The advantage of the conceptional idea in Figure 9 is that the integration of the ADAM framework can be used flexibly. In our integration example, the underlying MiL tests should be performed with or without the use of ADAM. If the ADAM framework is integrated, it passes the data onto the target model. It happens whether the modelled adversary took actions with this data or not. The data stream should be able to be passed through easily. Nevertheless, there can be a real-time requirement for the simulated environment regarding simulating safety-critical functions. Further work is required for this challenge.

We evaluated five out of seven requirements. In the context of the criteria, ADAM meets the requirements model-based, expressive, reusable, systematic, and visualisable. The requirements e) consistent and g) understandable can only be meaningfully evaluated in a later stage of the research project MASSiF. Hence, we did not evaluate these requirements.

It is being assumed, that further details and tighter definitions of the action, adversary goal, adversary characteristic and adversary perspective model are required because they are closely related to the specification of the EAI in future work. For example, a differentiated classification of the adversary characteristic values (compared to a simple Boolean consideration) may be necessary to enable a suitable cost-benefit analysis for implementing the adversary attacking decisions within the EAI.

By the integration of ADAM in the already existing work to model-based testing utilising environment model and system model [15], it is attempted to reduce effort and to use the already established knowledge. The extent to which domain-specific requirements can be met by the exemplary integration idea has to be considered in future work.

X. Conclusion and Future Work

In this extended paper, we present the ADAM framework for model-based security testing. The framework addresses security throughout the software engineering process. The main goal of ADAM is to provide a foundation to automate security testing in the early phases of software engineering (e.g., manual security reviews), which is the focus of this paper. Domains in which models are used at an early stage in software development will especially benefit from this approach. One example is the automotive industry, in which executable models are already used in the design phase, such as in MiL-methods for safety testing. ADAM can reuse these executable models. Besides, the results of ADAM can be used as a basis for assisting security testing activities in the later phases of the software development lifecycle, e.g., the generated attack paths can give hints that are worth considering in other tests like penetration tests. The central part of the ADAM framework is the attack modelling loop. During a loop, ADAM provides several perspectives on attacks. An attack is executed against a target simulation (target model). Using a system model as a part of the target model, in the context of the MiL-method, allows simulating attacks on software systems that are not implemented yet. The primary goal that the adversary wants to achieve drives the simulation and offers multiple paths of attacks. The preliminary evaluation of ADAM shows that the framework is expressive, reusable, systematic, visualisable and model-based. Future work will focus on detailed specification, implementation of the proposed elements, particularly the attack modelling loop, attack base and the testing part of the ADAM framework.

Acknowledgment

This work is part of the project "Modellbasierte Absicherung von Security und Safety für umfeldbasierte Fahrzeugfunktionen (MASSiF)". It is supported by the German Federal Ministry of Education and Research (BMBF) under the KMUinnovative program. Many thanks to our colleagues, Sanjana Biank and Matthias Meyer.

Federal Ministry of Education and Research

References

[1] T. Volkersdorfer and H.-J. Hof. "A Concept of an Attack Model for a Model-Based Security Testing Framework". In: *The Fourteenth International Conference on Emerging Security Information, Systems and Technologies Securware 2020* (Valencia, Spain, Nov. 21–25, 2020). IARIA, 2020, pp. 96–101. URL: https://www.thinkmind.org/articles/securware_2020_2_130_30046.pdf (retrieved 11/18/2021).

[2] H. Winner, S. Hakuli, F. Lotz, and C. Singer. *Handbuch Fahrerassistenzsysteme. Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*. 3rd ed. Springer Verlag, 2015. Chap. 8, 33. DOI: 10.1007/978-3-658-05734-3.

[3] S. Adepu and A. Mathur. "Generalized Attacker and Attack Models for Cyber Physical Systems". In: *2016 IEEE 40th Annual Computer Software and Applications Conference. 10-14 June 2016, Atlanta, Georgia : proceedings*. 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC) (Atlanta, GA, USA, June 10–14, 2016). Ed. by S. Reisman. Piscataway, NJ: IEEE, 2016, pp. 283–292. ISBN: 978-1-4673-8845-0. DOI: 10.1109/COMPSAC.2016.122.

[4] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas. *MITRE ATT&CK. Design and Philosophy*. Ed. by The MITRE Corporation. The MITRE Corporation. 7515 Colshire Drive, McLean, VA 22102-7539, 2018. Chap. 1, 2, 3, 4. URL: https://www.mitre.org/sites/default/files/publications/pr-18-0944-11-mitre-attack-design-and-philosophy.pdf (retrieved 09/25/2021).

[5] B. Schneier. "Attack Trees". In: *Dr. Dobb's Journal* 24.12 (1999), pp. 21–29. URL: http://macs.citadel.edu/baniks/427/Homework/attacktrees.pdf (retrieved 10/18/2021).

[6] K. Scarfone, M. Souppaya, A. Cody, and A. Orebaugh. *Technical Guide to Information Security Testing and Assessment*. Special Publication 800-115 800-115. Gaithersburg, MD 20899-8930: National Institue of Standards and Technology, Sept. 2008. URL: https://doi.org/10.6028/NIST.SP.800-115 (retrieved 10/05/2021).

[7] M. Zoppelt and R. T. Kolagari. "Reaching Grey Havens: Industrial Automotive Security Modeling with SAM". In: *International Journal on Advances in Security* 12.no. 3 & 4 (2019), pp. 223–235. ISSN: 1942-2636. URL: https://www.iariajournals.org/security/sec_v12_n34_2019_paged.pdf (retrieved 09/29/2021).

[8] F. Sommer, J. Dürrwang, and R. Kriesten. "Survey and Classification of Automotive Security Attacks". In: *Information* 10.4 (148 2019). URL: https://doi.org/10.3390/info10040148 (retrieved 11/15/2021).

[9] E. Hutchins, M. Cloppert, and R. Amin. "Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains". In: *Leading Issues in Information Warfare & Security Research* 1 (Jan. 2011). URL: https://www.researchgate.net/publication/266038451_Intelligence-Driven_Computer_Network_Defense_Informed_by_Analysis_of_Adversary_Campaigns_and_Intrusion_Kill_Chains (retrieved 04/01/2021).

[10] K. Kaynar. "A taxonomy for attack graph generation and usage in network security". In: *Journal of Information Security and Applications* 29 (2016), pp. 27–56.

ISSN: 2214-2126. DOI: https://doi.org/10.1016/j.jisa.2016.02.001.

[11] C. Ponikwar, H.-J. Hof, S. Gopinath, and L. Wischhof. "Beyond the Dolev-Yao Model: Realistic Application-Specific Attacker Models for Applications Using Vehicular Communication". In: *The Tenth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE 2016)* (Nice, France). July 2016. URL: https://arxiv.org/abs/1607.08277v1 (retrieved 03/18/2020).

[12] E. LeMay, W. Unkenholz, D. Parks, C. Muehrcke, K. Keefe, and W. Sanders. "Adversary-Driven State-Based System Security Evaluation". In: *Proceedings of the 6th International Workshop on Security Measurements and Metrics* (Bolzano, Italy). MetriSec 10 Article 5. New York, NY, USA: Association for Computing Machinery, Oct. 2010, pp. 1–9. DOI: 10.1145/1853919.1853926.

[13] E. LeMay, M. D. Ford, K. Keefe, W. H. Sanders, and C. Muehrcke. "Model-based Security Metrics Using ADversary VIew Security Evaluation (ADVISE)". In: *2011 Eighth International Conference on Quantitative Evaluation of SysTems*. 2011, pp. 191–200. DOI: https://doi-org.thi.idm.oclc.org/10.1109/QEST.2011.34.

[14] A. Drescher, A. Koschmider, and A. Oberweis. *Modellierung und Analyse von Geschäftsprozessen. Grundlagen und Übungsaufgaben mit Lösungen*. ger. De Gruyter Studium. München, Wien: De Gruyter Oldenbourg, 2017. Chap. 2, 3. DOI: 10.1515/9783110494532.

[15] M. Winter, T. Roßner, C. Brandes, and H. Götz. *Basiswissen Modellbasierter Test. Aus- und Weiterbildung zum ISTQB Foundation Level - Certificated Model-based Tester*. 2nd ed. dpunkt.verlag GmbH, 2016. Chap. 3, 4. ISBN: 97833864902970.

[16] J. M. Borky and T. H. Bradley. *Effective Model-Based Systems Engineering*. Cham: Springer International Publishing, 2019. DOI: 10.1007/978-3-319-95669-5.

[17] The MITRE Corporation, ed. *CAPEC Glossary*. 2019. URL: https://capec.mitre.org/about/glossary.html (retrieved 06/10/2021).

[18] J. Bryans, L. S. Liew, H. N. Nguyen, G. Sabaliauskaite, S. Shaikh, and F. Zhou. "A Template-Based Method for the Generation of Attack Trees". In: *Information Security Theory and Practice*. IFIP International Federation for Information Processing 2020. Ed. by M. Laurent and T. Giannetsos. Cham: Springer International Publishing, 2020, pp. 155–165. DOI: 10.1007/978-3-030-41702-4_10.

[19] *Road vehicles - End-of-life activation of on-board pyrotechnic devices. Part 1: General information and use case definitions*. ISO 26021-2:2008(E). May 2008.

[20] S. Moskal, S. J. Yang, and M. E. Kuhl. "Cyber threat assessment via attack scenario simulation using an integrated adversary and network modeling approach". In: *The Journal of Defense Modeling and Simulation* 15.1 (2018), pp. 13–29. DOI: 10.1177/1548512917725408.

[21] G. W. Maier, F.-R. Esch, and M. Kirchgeorg. *Motiv*. Ed. by S.-V. GmbH. Feb. 16, 2018. URL: https://wirtschaftslexikon.gabler.de/definition/motiv-39694/version-263096 (retrieved 08/03/2021).

[22] Bundesamt für Sicherheit in der Informationstechnik - BSI, ed. *Register aktueller Cyber-Gefährdungen und -Angriffsformen*. Bonn, 2018. URL: https://www.allianz-fuer-cybersicherheit.de/SharedDocs/Downloads/Webs/ACS/DE/BSI-CS/BSI-CS_026.pdf?__blob=publicationFile&v=1 (retrieved 07/03/2021).

[23] G. W. Maier and M. Kirchgeorg. *Motivation*. Ed. by S.-V. GmbH. Feb. 19, 2018. URL: https://wirtschaftslexikon.gabler.de/definition/motivation-38456/version-261879 (retrieved 07/09/2021).

[24] J. D. Howard and T. A. Longstaff. *A Common Language for Computer Security Incidents*. Tech. rep. Oct. 1998. URL: https://www.osti.gov/biblio/751004 (retrieved 05/06/2021).

[25] *Durchführungskonzept für Penetrationstests*. Studie. Bonn: Bundesamt für Sicherheit in der Informationstechnik-BSI, 2003. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Studien/Penetrationstest/penetrationstest.pdf?__blob=publicationFile&v=2 (retrieved 09/23/2021).

[26] C. Miller and C. Valasek. "Remote Exploitation of an unaltered passenger vehicle". In: *Black Hat USA* (2015). URL: http://www.illmatics.com/Remote%20Car%20Hacking.pdf (retrieved 06/04/2021).

[27] The MITRE Corporation, ed. *CAPEC*. URL: https://capec.mitre.org/ (retrieved 08/11/2021).

[28] F. Sommer and J. Dürrwang. *IEEM-HsKA/AAD: Automotive Attack Database (AAD)*. 2019. URL: https://github.com/IEEM-HsKA/AAD (retrieved 06/24/2021).

[29] The MITRE Corporation, ed. *CWE*. URL: https://cwe.mitre.org/ (retrieved 08/11/2021).

[30] The MITRE Corporation, ed. *CVE*. URL: https://cve.mitre.org/index.html (retrieved 08/11/2021).

[31] The OWASP Foundation, ed. *OWASP Top 10 - 2017. The ten most critical web application security risks*. Bel Air, MD 21014 US, 2017. URL: https://github.com/OWASP/Top10/raw/master/2017/OWASP%20Top%2010-2017%20(en).pdf (retrieved 07/18/2021).

[32] Object Management Group, ed. *Unified Modeling Language*. 2.5.1. Needham, MA 02494, USA, 2017. Chap. 15, 16. URL: https://www.omg.org/spec/UML/2.5.1/PDF (retrieved 07/22/2021).