Towards a holistic approach to security validation of construction machinery through HIL systems

Sheela Hariharan*†, Andreas Erséus*, Thomas Nolte†, Alessandro V. Papadopoulos†
* Volvo Construction Equipment, Eskilstuna, Sweden. {name.surname}@volvo.com
† Mälardalen University, Västerås, Sweden. {name.surname}@mdu.se

***Abstract*—The construction industry is increasingly equipping its machinery with sophisticated embedded systems and modern connectivity. Technology advancements in connected safety-critical systems are complex, with cyber-security becoming a more critical factor. Due to interdependencies and network connectivity, attack surfaces and vulnerabilities have increased significantly. Consequently, it is imperative to perform a risk assessment and implement robust security testing methods in order to prevent cyber-attacks on machinery segments. This paper presents a method for identifying potential security threats that also affect machine functional safety, facilitated by identifying threats in the threat modeling process and analyzing safety-security synergies. By identifying such risks, attack scenarios are created to simulate cyber-attacks and create test cases for validation. This approach integrates security testing into the current testing process by using penetration testing tools and utilizing a Hardware-in-the-Loop(HIL) test setup and it is verified with a simulated Denial of Service attack over a CAN network.**

I. INTRODUCTION

Construction equipment is transitioning from standalone, primarily electro-mechanical systems to integrated cyber-physical systems. With a future toward autonomy, this progress will allow for safe machine operation and the supply of extra aftermarket options. However, cyber-security needs to be much strengthened to develop secure, automated, and interactive operations that have to be run in various locations like construction sites, quarries, and other environments [1]. Recent years have seen the introduction of new attack vectors that show how cyber-security risks have expanded beyond discrete automobiles to include fleets, smart mobility APIs, EV charging infrastructure, and supply chains, in addition to the incidents the automotive sector has experienced over the past ten years. It is estimated that malicious hackers are responsible for 63% of all ongoing cyber-attacks [2], which is a substantial percentage. This assessment also highlighted that sectors with growing digital footprints are vulnerable to cyber-attacks that intrude on infrastructure, including off-road machines and public safety [3].

While off-road machines like construction equipment share similarities in ECU architecture, backend servers, and other connectivity protocols analogous to on-road vehicles, they differ significantly in other security aspects. For example, Original Equipment Manufacturers(OEMs) are responsible for specifying all requirements for their suppliers in the supply chain for on-road vehicles since there are only a limited number of variations between models. In construction machinery, there are considerable differences by model, including engine type, transmission type, and even control module type. As a result, the security of construction equipment is a shared responsibility between respective entities, including the construction equipment manufacturers, chassis builders, aftermarket suppliers, and fleet management owners, requiring more open security architectures [4]. These differences significantly increase the attack surfaces on construction machines, which also have strict functional safety requirements. According to [5], failure to maintain the system's safety compromises its security thus it is important to identify and mitigate cyber-security threats that directly impact the safety of the overall system. Security compromises have many impacts on construction machines, including safety hazards, downtime, and financial losses. Thus, cyber-security becomes a key factor in construction machinery products. To address these challenges in product security and have a secure product, extensive security analysis, and validation must be carried out by considering the threat modeling, requirements, and functional safety parameters using available test resources and additional tools. Thus, a question arises on '*How can we derive the secure validation concept and determine to execute the testing scenarios in the available test systems?*'

To address this question, this paper proposes a method for defining all known attacks for a particular system, in addition to identifying any adverse effects of a safety feature based on attack and damage scenarios. The mapping of these potential threats and risks into a database is then used to generate a shared library, which can be used to validate the existing system using testing environments such as HIL test setup, in conjunction with penetration testing tools. Based on this database, attack paths and test cases are developed for testers to execute. The next step involves simulating an actual attack and analyzing the system's behavior to determine its security and safety implications. A novel aspect of the proposed methodology is that it can be applied to the current stages of the testing life cycle in a way that can significantly benefit the industry in discovering security vulnerabilities. The main contributions of this paper are the following:

- A common and systematic threat and risks library creation for constructing attack scenarios that are continuously monitored and updated for newer vulnerabilities.

The work presented in this paper is supported by Volvo Construction Equipment, Mälardalen University, and the Swedish Knowledge Foundation (KKS) via the project ARRAY++.

- A holistic validation approach as risk-based to evaluate the created attack scenarios that can be adapted at different stages of the testing pipeline.
- Simulation of attacks by testers using HIL setup or equivalent with additional penetration testing tools.

This paper is organized as follows. Section II provides the necessary background and concepts of the proposed work and Section III outlines the related work. Section IV presents the holistic approach to validation utilizing attack scenarios, while Section V summarizes our findings and evaluation. Finally, Section VI concludes with a discussion of the implications of our work.

II. Background

In this section, we first describe the current and upcoming cyber-security standards in automotive and construction machinery as well as the threat modeling described in such standards. Following this, we outline the various security testing methods utilized to validate our derived attack scenarios.

A. Cyber-security standards

SAE J3061 [6] was the introductory standard to establish a set of rules and guidelines for cyber-security in vehicles. The ISO/SAE 21434 [7] standard was released later on to establish a set of high-level cyber-security principles, which was followed by UNCECE RN155 [8] and R156 [9] regulations. All these standards and regulations are to address the security aspects of automotive products and also define the methodology for threat modeling. Since these standards are recommended for on-road vehicles, the cyber-security standards for off-road construction machinery follow different ones. The European Commission machinery directive sets cyber-security requirements and policies for machinery products [10]. This directive created new acts to address security with an impact on safety like human-machine interaction and machinery with emerging technologies like Artificial intelligence(AI). Additionally, the European Commission has proposed a Cyber Resilience Act [11] which is designed for regulating cyber-security requirements for products containing digital components, which is primarily intended to strengthen security rules to ensure hardware and software security.

B. Threat modeling – Threat Analysis and Risk Assessment (TARA)

The TARA methodology is used by most industries to assess the security risks associated with their products, services, and solutions. TARAs should cover all variants of a product that have been evaluated as cyber-security relevant. An effective TARA provides a good basis for identifying security vulnerabilities [12]. In TARA, the first step is ’*Item identification*,’ which identifies the assets, functions, or components that must be protected from security threats. Next, ’*Damage analysis*’ is an assessment of the potential impact of potential threats. This analysis evaluates the threat’s severity, consequences, and likelihood. It is described later in ’*Damage scenarios*’ how a threat may exploit a vulnerability in an item, resulting in damage or harm. An asset is then subjected to a ’*Threat analysis*,’ which involves identifying and analyzing potential threats to the asset and assessing the impact of potential threats. Lastly, an ’*Impact rating*’ indicates whether a given threat scenario is likely to negatively impact assets based on specific criteria. Fig. 1 explains the TARA process flow.

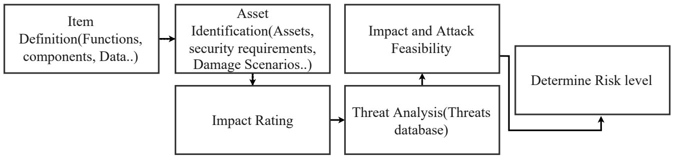

Item Definition(Functions, components, Data..)
Asset Identification(Assets, security requirements, Damage Scenarios..)
Impact and Attack Feasibility
Determine Risk level
Impact Rating
Threat Analysis(Threats database)

Fig. 1. TARA - Process flow.

Terminologies: An “*attack scenario*” occurs when an attacker tries to exploit weaknesses in a system to cause harm or gain unauthorized access. A *damage scenario* refers to adverse consequences or undesirable outcomes caused by compromising an asset’s cyber-security property(s) [7]. It describes the adverse effects of an attack on a machine, a machine function, or a service that affects a stakeholder’s interest. In addition, if a fault occurs in the item (in hardware or software) that affects the cyber-security properties of the primary or supporting assets [6], this is not considered to be a cyber-security damage scenario.

C. Denial of Service (DoS) attacks

DoS attacks and their various forms represent most attacks on availability attributes in cyber-security properties like Confidentiality, Integrity, and Availability defined in ISO/SAE 21434 [7]. Networks can be subjected to Dos attacks when they are unable to function as expected. It interrupts network services, such as routing services, for its authentic users when a DoS attack occurs. Dos attacks can partially or completely restrict legitimate users’ access to network resources, which will result in a degradation in service or a total denial of service [13]. A DoS attack could be the result of software bugs or environmental conditions, or it could be the result of an affected node refusing to cooperate with another node. There is a possibility that malicious nodes could refuse to forward packets to other ECU nodes, which can prevent safety-critical messages, such as braking and speed information, from reaching the engine controls. In addition, DOS may be accomplished by dropping malicious packets, which reduces network throughput and eventually results in service outages.

D. Security testing methods

Security testing methods are more established in the software engineering field, and they use a variety of different types of testing methods to identify security vulnerabilities. According to [14], security testing can be classified into Model-based testing, Code-based testing, Penetration testing, Regression testing, and Risk-based testing. These testing techniques have been adapted to secure the system development life-cycle. Few automotive industries use penetration testing or fuzzing as security testing methods, although more researchers are attempting to adapt these methods to the automotive industry.

a) Penetration testing (Pen-testing)

Penetration testing is an approach to security assessment that is used to detect security vulnerabilities by performing testing from the perspective of an attacker [15]. Security testers test the ability of vehicle software systems against malicious behavior and attempt to penetrate the System Under Test (SUT). Several studies have been conducted on various types of physical and remote attacks. Even though penetration testing produces the most significant results, it is often time-consuming and manual and requires deep domain knowledge. Automating known attacks is always an essential component of functional penetration testing. Through penetration testing, well-known issues and attacks can be detected, as well as the most likely and significant attacks. For vehicle software systems to be resilient, penetration testing alone is insufficient. Penetration testing tools reproduce cyber-attacks which are used to manipulate the SUT and simulate the attacks. In this paper, penetration testing tools like CanUtils [16], [17], Wireshark [18], and SavvyCAN [19] are used for simulating the attack scenarios.

b) Risk based testing

Risk-based testing is implemented to achieve two different techniques [20]. One is to optimize the overall security testing process with the results from threat and vulnerability analysis which provides the baseline for the test implementation. Other is to derive attack simulations that can be evaluated with SUT against its security specifications. The purpose of risk-based testing [21] is to facilitate security testing by assessing security risks based on indicators from a variety of artifacts obtained from the secure development lifecycle, such as the impact of requirements and software complexity.

c) Fuzzing

A fuzz test is a scalable testing method involving giving random inputs to the SUT to check for unexpected behavior. It is an automated test procedure used in cyber-security to simulate a potential 'cyber-attack' and identify vulnerabilities before launching a product. There are three types of fuzz testing: white box fuzzing, gray box fuzzing, and black box fuzzing [22]. Due to the complexity of automotive software, testers mainly perform black-box fuzzing, as white-box fuzzing requires more effort and time. In the automotive industry, where the supply chain is extensive, performing white box testing by the manufacturer will not be feasible because tier suppliers develop software.

III. Related Work

There has been a growing interest in product security fields in academia and industry, and many researchers have investigated security vulnerabilities in the on-road segments [23], [24], [25], [26], [27]. Different areas are susceptible to security vulnerabilities, among which are software, hardware, and communication networks. Considering the complexity of the supply chain, the involvement of fleet owners, the presence of aftermarket tools and suppliers, and the differences in security standards, product security in construction machinery is unique. Additionally, establishing security testing in an industry is difficult due to the wide variety of entities involved in producing machine products, and thus determining security requirements is challenging.

A. Automotive cyber-security testbeds

There is a substantial amount of work being done in the cyber-security testbed that works with HIL systems in industrial control systems [28], [29], [30]. Although industrial control system security vulnerabilities differ from those found in the machinery domain, these studies will provide a better understanding of the need for and methods of implementing security testing. There is also a considerable amount of work being done in academia in the development of automotive cyber-security test beds. There is a proposal that [31] CAN fuzzers can be used in the design phase of a system to provide security. The black box testing method and clear guidelines regarding the importance of security testing and the challenges associated with testing methods are presented in [32], which also identified the difficulties connected with supply chain management. A comprehensive test model that includes a variety of wireless protocols and attack tree surfaces were also incorporated. A security verification and validation process has been developed by the same group [33] which provides a good process for using existing security testing tools and facilitates a variety of verification and validation techniques. CAN fuzzers are developed that systematically define fuzzers, fuzzing configurations, and Oracle functions for testing automotive ECUs using CAN interfaces in [34]. An evaluation of fuzzing oracles for electronic control units was conducted with the development of a sensor harness. Fuzz testing process is recommended to be implemented into the continuous integration pipeline for automotive systems by [35]. Through the application of fuzz testing throughout development in a continuous integration pipeline, issues are detected at an early stage in the development process. The testbed is developed for remote cyber-security testing of heavy vehicles in [36]. Specifically, it supports J1939 networks commonly found in heavy vehicles, such as buses and trucks. For the testing setup, the authors used real ECUs and simulated node controllers based on Linux. The testbed offers several features that are useful for studying and manipulating network traffic Overall, the current literature study indicates that penetration testing, fuzzing, as well as model-based testing have been extensively covered.

B. Safety-security interplay

Several studies have been conducted in the area of security and safety correlation in the automotive industry. The systematic literature mapping study is performed in [37] that provides an overview of all the relevant studies related to safety and security conducted in the automotive sector, and [5] provides a summary of functional safety and cyber-security standards. A

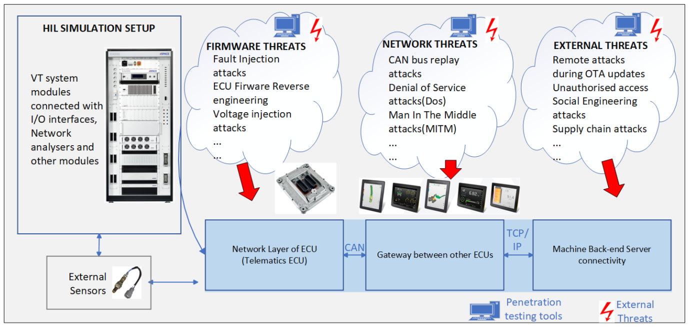

HIL SIMULATION SETUP
VT system modules connected with I/O interfaces, Network analysers and other modules
FIRMWARE THREATS
Fault Injection attacks
ECU Firware Reverse engineering
Voltage injection attacks
...
...
NETWORK THREATS
CAN bus replay attacks
Denial of Service attacks(Dos)
Man In The Middle attacks(MITM)
...
...
EXTERNAL THREATS
Remote attacks during OTA updates
Unauthorised access
Social Engineering attacks
Supply chain attacks
...
...
External Sensors
Network Layer of ECU (Telematics ECU)
CAN
Gateway between other ECUs
TCP/IP
Machine Back-end Server connectivity
Penetration testing tools
External Threats

Fig. 2. Validation setup with HIL systems.

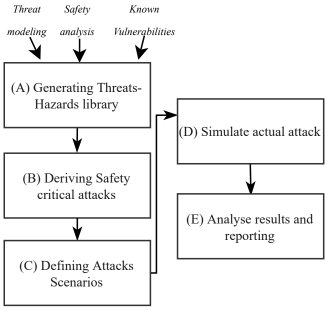

Threat modeling
Safety analysis
Known Vulnerabilities
(A) Generating Threats-Hazards library
(B) Deriving Safety critical attacks
(C) Defining Attacks Scenarios
(D) Simulate actual attack
(E) Analyse results and reporting

Fig. 3. An approach for deriving attack scenarios for validation.

security-focused safety validation method is proposed in [38]. An integrated approach is presented which enables systematic derivation of both safety and security constraints from the system safety case in [39]. The overview of the organizational challenges that must be solved to fulfill the requirements of industrial projects that also consider both functional safety and cyber-security [40]. An integrated approach combining safety with security is presented in [41] in the context of autonomous driving.

IV. Holistic method for security validation

In this paper, we implement a holistic approach by creating attack scenarios that can be applied later in an existing testing environment to simulate attacks based on those scenarios, as illustrated in Fig. 2. The approach is designed as a holistic process and can be adapted at any stage of the testing pipeline following the testing requirements. As an example, unit testing can be used to test attack scenarios that can only be executed on a particular system, and HIL testing can be used to test scenarios that require external interfaces. As a result of this approach, safety requirements are verified in parallel with security requirements, and potential safety concerns are validated without compromising system security. This enables an understanding of the system's vulnerabilities on a comprehensive level by validating both safety and security concerns.

Following are the **5 steps** proposed in this approach, as illustrated in Fig. 3.

Step1 - (A) Generating Threats-Hazards library:

The threat-hazard library is generated by combining the threat database from threat modeling and the risk database from risk assessment. The key objective of this initial step is to formulate critical security risks and general scenarios that affect safety and security for the base development of the library. In addition, this library is continually updated with known exposures from databases like NVD [42] and also with revisions of threat modeling. In the later stages of this approach, assets are identified, and related threats and risks are mapped according to their attack types. This is the common library that will be augmented and used in simulating attack scenarios for testing several SUT or applications. Following are examples of two machinery-related scenarios and their sub-scenarios as evidence for developing a threat library.

Scenario 1: Communication – Inter-/intra-vehicle communication

Externally, machines communicate with other systems and devices via cellular networks, WiFi networks, USB connection, camera interfaces, or Bluetooth. In addition, it has in-vehicle communication via SAE J1939, CAN, and LIN. Diagnostics can be done through On-Board-Diagnostics(OBD) and with after-market tools. Each of these communications affects safety and security functions, which act as attack vectors.

Sub-scenario:

- Hardware and software components of a machine can be vulnerable to external connections, raising safety concerns.
- Connectivity and communication between OBD and aftermarket tools present security and safety concerns that can be exploited.
- Connected machines that send diagnostic information to the backend office.

Scenario 2: Working environments – Construction machinery sites, autonomous quarries

Construction machines are operated in different environments like mining, quarries, and building sites. There is a risk of harm to operators or users in this operation, which has a high level of functional safety. *Sub-scenario:*

- Safety of machines and users operating in these working environments.
- Autonomous machines working on sites along with manually operated machines.

Scenario 3: Diagnostics – Fault tracing, software updates, and fleet management

Technicians and service engineers mostly do fault tracing as machines cannot be driven to workshops. Sometimes, fleet management owners can use different aftermarket options or fault tracing procedures other than OEMs. *Sub-scenario:*

- Troubleshooting of machines by technicians whose tools may have been infected with malware.
- A fleet owner's machine may impact an operator's privacy information, job orders, and machine status.

As an example, the above scenarios illustrate how the library will function when creating tests for chosen scenarios.

Step2 - (B) Deriving safety-critical attacks:

Initially, scenarios and sub-scenarios are created by assembling information from threat and risk libraries. Using this library as a basis, the next step is identifying assets, asset types, security analyses, and attack types for each sub-scenario. The information obtained from these asset collections and analyses determines safety critical risks. After the formulation of the library has been completed, it is possible to conduct safety analyses, taking into account the inputs from the risk assessment. This will enable us to determine the most significant risk level for safety, which can then be categorized as a safety critical risk. Several assets influence a particular scenario. To simplify the analysis, we take scenario 1 from the previous step into account when determining assets and safety-critical risks.

Scenario 1: Communication – Inter-/intra-vehicle communication

1) **Asset** – ECU, gateway
2) **Asset group** – Hardware, software
3) **Safety risk severity** – Severity level S1 (High)
4) **Security attack feasibility rating** – Critical (4)
5) **Safety critical goal** – Critical, securing all communications properly in order not to get access to the communication network. By developing this matrix, it is possible to identify which risks are safety-critical in all scenarios and also determine how the machine reacts when a security threat occurs. These ratings are performed following ISO/SAE 21434 [7] and ISO 19014 [43].

Step3 - (C) Defining attacks scenarios:

During this step, we will create threat and attack scenarios and map them according to their assets, threat types(s), and attack type(s) identified in earlier steps.

1) *Creating threat scenarios:* Threat scenarios can be generated utilizing threat modeling techniques such as TARA [44], as well as our existing library of threat scenarios. The boundless possibilities of threat scenarios are obtained from any of these threat models, and it isn't easy to initially accept all of these inputs. A threat scenario can be identified more easily if asset types and asset interests are taken into consideration. In this way, it might be possible to understand what threat scenarios are necessary and required for the system. For example, we will take scenario 1: Communication - Inter-/intra-vehicle communication to create threat scenarios.
   Scenario 1: Communication – Inter-/intra-vehicle communication
   Possible threat scenarios include:
   a) Spoofing of messages sent and received over the CAN bus.
   b) Tampering with remotely operated functions like external mobile application, remote diagnostics, and connection to backend office.
   c) Denial of service when certain services are requested as UDS commands.
   d) Elevation of privilege when there is unauthorized access to certain functions for example through USB or wireless protocols.
2) *Creating attack scenarios:* By creating attack scenarios, a comprehensive understanding of how and what to test for security attacks can be gained. This step is primarily planned to describe potential attacks and to demonstrate that implemented security controls prevent the type of attacks that were performed. It is also intended to establish that there were no violations of safety risks during this attack scenario. We create attack scenarios by taking the example of Scenario 1.
   Scenario 1: Communication – Inter-/intra-vehicle communication
   Possible attack scenarios include:
   a) Attacker gains unauthorized privilege by doing insider attacks.
   b) Attacker attempts to perform a man-in-the-middle attack by eavesdropping over the CAN network.
   c) Attacker tries to spoof the network by scanning the aftermarket services and tools.

   As shown in Table I, assets, threat types, and possible scenarios are mapped to set up the boundary for performing validation.

Step4 - (D) Simulate actual attack:

Many security testbeds have been developed and methods proposed for testing automotive security [45] each with advantages and disadvantages. The main challenge with respect to these security testbeds is testing in an actual vehicle environment fulfilling functional safety risks. The following measures should be considered to validate the attacks on the SUT:

1) The tester must have a complete understanding of what attack scenarios to simulate and how to simulate them.
2) SUT must be prepared with all necessary tools and preconditioning parameters.
3) Expected measures to be taken in the event of a successful attack. For example, certain functions may become unavailable, or the ECU may crash as a result of the attack.
4) It is essential that the tester is aware of situations in which the attacks that have been created might fail, as well as the reasons for the failure.

The SUT should be adequately prepared in advance for the simulation of attacks, taking into consideration the tools, additional hardware, and interfaces. In addition, the tester should be aware of the expected behavior of the attack scenarios.

This step focuses on creating a suitable test setup for performing attacks based on the attack scenarios created in the earlier steps. This step is further described in the following sections.

1) *Setting up the test environment:* Since security testing in construction machinery is a relatively novel testing methodology, a dedicated security testing setup could

TABLE I
SCENARIO 1: COMMUNICATION – INTER-/INTRA-VEHICLE COMMUNICATION - MAPPING OF ASSETS TO THREAT AND ATTACK SCENARIOS.

Asset	**Threat scenario**	**Threat type (STRIDE)**	**Attack type**	**Attack scenarios**	**Examples for possible attacks**
ECU	Code injection, Message fault injection, reprogramming through remote / USB / Aftermarket tools.	Tampering.	Replay of messages on CAN bus.	An attacker who attempts reconnaissance in the network by scanning its services and tools.	Injection of communication data (Protocol attacks in the form of corrupting payload).
Communication protocols	External interfaces connected like USB, and ethernet ports communicate with ECU through CAN protocol.	Spoofing.	Sending false messages.	An attacker attempts to hinder interfaces by inserting malicious values into protocol requests and responses.	USB connection to the infotainment malware and sending fault messages on the bus.
Diagnostics tools	Diagnostics commands sent from aftermarket tools.	Denial of service.	Rejection of UDS services.	An attacker attempting to send random UDS services to unlock the ECU for example.	Injection of diagnostics data through OBD (Physical attacks).
Gateway	Insider attacks by using privileges.	Elevation of Privilege.	Successful unauthorized access.	An attacker attempts to gain access to the ECU by finding the potential attack vector.	Develops or technicians unexpectedly installing malware when using (Type of social engineering attack).

TABLE II
SECURITY TOOLS.

Open source tools	**Purpose**
Instrument cluster simulator (ICSim)	Used for CAN bus reversing and works with Virtual CAN devices configured on Linux.
Kayak	Java-based tool for CAN traffic analysis, extended to use for GPS tracking.
Caring Caribou	Designed as Nmap tool for vehicle hacking like brute forcing UDS services.
Octane CAN bus sniffer	Open source CANbus sniffer runs on Windows for CAN bus transmission and reception.
Wireshark	Network monitoring tool for CAN network that can be used with Linux candump/can-utils.
Nmap	Used with Linux for scanning open ports.

prove challenging and cost-prohibitive. In our holistic approach, this testing step serves the purpose of reusing tools and environments already available for testing purposes. Therefore, HIL is used to set up the test environment for the selected attack scenarios during the design phase. It is possible to monitor network communication utilizing network analyzers. Furthermore, we require the installation of a virtual machine to use the available Linux security testing tools for conducting penetration testing. A few well-known tools [46] that can be used for simulating attacks are listed in Table II. Although these tools are open source and not specifically developed for automotive, they are sufficient to execute most of the scenarios since HIL has other interfaces and network analyzers that can be used for monitoring test behavior.

2) *Generating tests based on derived attack scenario:* The test cases were written by the type of attack to be simulated. As an example Table I, if wireless interfaces were being attacked, test cases would have to be generated based on the attack on the Bluetooth protocol. Test cases are derived from the database of security requirements and risk assessments. In addition, test cases based on attack scenarios that directly impact safety must be developed. This procedure creates an appropriate test model as a 'black-box model' to be tested using HIL systems. Although it is possible to generate test cases automatically, it isn't straightforward at this stage of development since the test environment is being prepared for the first time. As a result, we will perform manual tests on the scenarios taken from the database and with the tools identified from the previous stage.

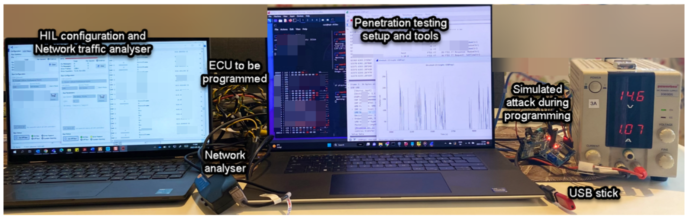

HIL configuration and Network traffic analyser
ECU to be programmed
Penetration testing setup and tools
Simulated attack during programming
Network analyser
USB stick

Fig. 4. Experimental setup.

3) *Selecting security test methods:* The background section provides an overview of different security testing methods, such as penetration testing, fuzzing, and code analysis. Depending on the attack scenario that will be tested, the appropriate testing procedure will be selected. In the same scenario of a Bluetooth attack as described in the step for generating test cases, pen testing may be an accurate testing method. Pen testing and fuzzing can be used for CAN-related attacks in another scenario. For an example with Scenario1: Communication – Inter-/intra-vehicle communication, the selection of test methods could be:
   a) Simulating spoofing and message reply attacks: Penetration testing
   b) Gaining unauthorized access to breaking in UDS commands: Fuzzing

4) *Validity of test cases, tools and performing tests:* Before executing planned tests, validating the test cases and tools should be performed to determine where the tests fail. It is sufficient to validate test cases and toolsets before the tests are conducted by the standard procedure. When the

SUT is successfully configured, for example, with HIL systems, the test is performed based on the potential outcome of the test. Test engineers should monitor the post-attack conditions to determine whether the attack was successful or unsuccessful. The two states of 'success' and 'failure' should clearly distinguish how the safety goal is violated in each case. It might be necessary to produce a detailed report on how attacks are deployed and what behavior is observed in SUT for further evaluation.

Step5 - (E) Analysis results and reporting:

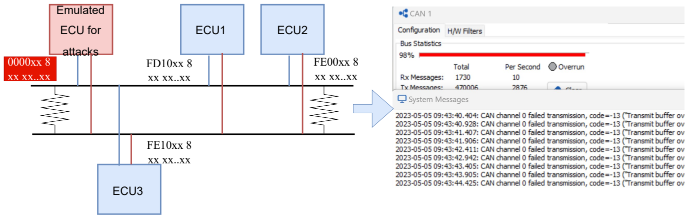
It is essential to report the test results and behavior of the SUT to security specialists, who will determine the mitigation methods for each simulated attack. In this paper, mitigation methods are not discussed as we are primarily concerned with the validation of attack scenarios.

V. Evaluation and Results

Experimental setup: Fig. 4 shows the experimental setup for testing the proposed validation method. This uses a hybrid setup with a HIL system for simulating other ECUs on a network, with the SUT being an actual ECU and an emulation board for delivering attacks. A HIL system uses VT cards (VT2004, VT2848) to simulate sensor signals, input signals for ECUs, voltage simulations, and digital I/O modules which are controlled by Vector VTeststudio and Canoe network analyzer. The real-time ECU comes with a powerful SA8155P Qualcomm Adreno automotive cockpit platform, which runs on Linux and Android operating systems. It also has a Rubus-based MPC57x NXP microcontroller, a CAN protocol transceiver, TJA1043T, and a dual high-speed CAN transceiver, TJA1048. An emulation board Az-Delivery is powered by an ATmega328P chip programmed in Arduino IDE, and the MCP2515 chip is used for simulating Denial of Service(DoS) attacks on the CAN bus. HIL configuration is controlled by one computer, and penetration testing tools are used on another computer running Kali Linux in a virtual machine. The Kvaser USB CANprofessional captures the traffic between the emulation board and the actual ECU which is through a special breakout box to read the ECU pins directly.

Testing tools: Kali Linux [17] is installed over VMware workstation player [47] on a test computer for bus monitoring and simulating attacks on real ECU. Kvaser USBCAN pro is connected to the Linux computer, where the connection is established to capture CAN traffic. Kali Linux packages SavvyCAN [19] and CAN-utils [16] are used to simulate the attacks on CAN network.

Attack simulation (CAN DoS attacks): For validating the attack scenarios created by the proposed approach, in this use case, the Denial-of-Service attack type is selected to be simulated on the test setup that has been designed. DoS attacks on CAN affect the availability property of the cyber-security triad. DoS attacks as shown in Fig. 5 on CAN buses aim to disrupt or disable services between ECUs. The simulated attack involves sending random messages with a high priority bit in the CAN ID continuously on the CAN bus by loading the bus with messages of high frequency. As a result, the functionality of ongoing messages is disrupted on the bus, leading to discrepancies between the communication of the ECUs. In this scenario, the CAN ID 0x00 is sent continuously at 10ms over the bus to increase the busload.

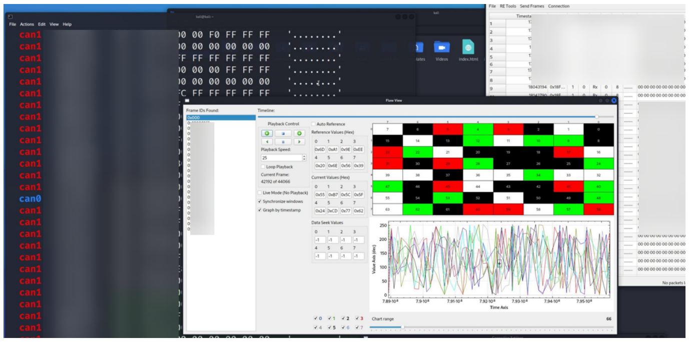


Bus Statistics
98%

	Total	Per Second
Rx Messages:	1730	10
Tx Messages:	470006	2876

```
2023-05-05 09:43:40.404: CAN channel 0 failed transmission, code=-13 ("Transmit buffer ov
2023-05-05 09:43:40.928: CAN channel 0 failed transmission, code=-13 ("Transmit buffer ov
2023-05-05 09:43:41.407: CAN channel 0 failed transmission, code=-13 ("Transmit buffer ov
2023-05-05 09:43:41.906: CAN channel 0 failed transmission, code=-13 ("Transmit buffer ov
2023-05-05 09:43:42.411: CAN channel 0 failed transmission, code=-13 ("Transmit buffer ov
2023-05-05 09:43:42.942: CAN channel 0 failed transmission, code=-13 ("Transmit buffer ov
2023-05-05 09:43:43.405: CAN channel 0 failed transmission, code=-13 ("Transmit buffer ov
2023-05-05 09:43:43.905: CAN channel 0 failed transmission, code=-13 ("Transmit buffer ov
2023-05-05 09:43:44.425: CAN channel 0 failed transmission, code=-13 ("Transmit buffer ov
```

Fig. 5. CAN-DoS attack representation.



```
can1   00 00 F0 FF FF FF   '........'
can1   00 00 00 00 00 00   '........'
can1   FF FF FF FF FF FF   '........'
can1   00 00 00 FF FF FF   '........'
can1   00 00 00 00 00 00   '........'
can1   FC FF FF FF FF FF   '........'
```

Frame IDs Found: 0x000

Playback Speed: 25

Current Frame: 42192 of 44066

Reference Values (Hex)

0	1	2	3
0x6D	0xA1	0x9E	0xEE
4	**5**	**6**	**7**
0x20	0x6E	0x56	0x39

Current Values (Hex)

0	1	2	3
0x55	0xB7	0x5C	0x5F
4	**5**	**6**	**7**
0x24	0xCD	0x77	0x62

Data Seek Values

0	1	2	3
-1	-1	-1	-1
4	**5**	**6**	**7**
-1	-1	-1	-1

Fig. 6. Message frames and network disruption captured from pen-testing environment.

Results: As part of this penetration test, an enormous number of CAN packets are sent to the actual network via the SavvyCAN tool through the emulation board to produce a DoS attack. According to Fig. 6, the DoS attack increases the CAN bus load and disrupts other CAN messages on the bus. The graphical representation shows how normal communication is affected by having random high priority on the bus. It is noticed that the CAN ID frames transmitting critical parameters, such as hydraulic functions, are delayed on the bus. It will affect the machine's performance and cause harm to the operators and other humans at the construction site. But, DoS attacks using physical access can only be produced on a limited scale. In contrast, a similar attack can be carried out remotely through a wireless network that has access to the main CAN bus of the network and can be simulated by using other penetration testing tools. It is intended to demonstrate the feasibility of simulating these attacks without special test equipment and the effectiveness of the validation method by using an ordinary scenario. The simple mitigation measure is to have the proper filtering and monitoring measures of the transmitted and received CAN messages over the network. The results of this experiment demonstrate that this holistic approach can be used to derive attack scenarios and simulate the tests on available setups with the assistance of additional penetration testing tools.

VI. Conclusion

In this paper, we proposed and validated a holistic approach for security validation through attack scenarios, taking safety criticality into account, primarily for the context of construction machinery. Our approach identifies the adverse effects of security issues that also detect safety impacts. We proposed a testing process that could be adapted to the product development cycle and used the existing testing environment with a few open-source tools added on. This is later then validated against the simulated CAN DoS attack of the actual ECU connected in the network using the HIL setup. We have presented some attack vectors and created attack scenarios from the common library derived from Threat modeling.

For future work, we aim to improve this process by incorporating licensed tools, creating risk models and behavioral models of the SUT, and then running the validation as model-based security testing. The results of modeling will enable the testing framework automated and compatible to run the tests iteratively to find more security vulnerabilities.

References

[1] C. Jichici *et al.*, "Effective intrusion detection and prevention for the commercial vehicle SAE J1939 CAN bus," *IEEE Trans. Intelligent Transportation Systems*, vol. 23, no. 10, pp. 17 425–17 439, 2022.

[2] Upstream security, "Upstream's 2023 global automotive cybersecurity report," Tech. Rep., 2023. [Online]. Available: https://upstream.auto/reports/global-automotive-cybersecurity-report/

[3] Y. Burakova *et al.*, "Truck hacking: An experimental analysis of the SAE J1939 standard," in *USENIX Work.Offensive Technologies (WOOT)*, 2016.

[4] G. D'Anna, *Cybersecurity for Commercial Vehicles*. SAE International, 2018.

[5] G. Macher *et al.*, "An integrated view on automotive SPICE, functional safety and cyber-security," in *SAE Technical Paper Series*, 2020.

[6] SAE International, *J3061: Cybersecurity guidebook for cyber-physical vehicle systems*. [Online]. Available: https://www.sae.org/standards/content/j3061_201601/

[7] ISO Standard, *ISO/SAE 21434 - Road Vehicles – Cybersecurity engineering*. https://www.iso.org/standard/70918.html.

[8] UNECE Regulation R155, *R155*. [Online]. Available: https://unece.org/transport/documents/2021/03/standards/un-regulation-no-155-cyber-security-and-cyber-security

[9] UNECE Regulation R156, *R156*. [Online]. Available: https://unece.org/transport/documents/2021/03/standards/un-regulation-no-156-software-update-and-software-update

[10] Regulation of the European Parliament and of the Council on Machinery Products. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52021PC0202

[11] Cyber Resilience Act, European Commission. [Online]. Available: https://digital-strategy.ec.europa.eu/en/library/cyber-resilience-act

[12] Itemis security analyst: Threat and risk assessment tool. [Online]. Available: https://nvd.nist.gov/

[13] W. A. Al-Dhuraibi *et al.*, "Securing vehicular ad-hoc networks: a ddos case study," in *Int. Conf. Computation, Automation and Knowledge Management (ICCAKM)*, 2021, pp. 112–117.

[14] I. Pekaric *et al.*, "Applying security testing techniques to automotive engineering," in *Int. Conf. Availability, Reliability and Security*, 2019.

[15] S. Mahmood *et al.*, "Systematic threat assessment and security testing of automotive over-the-air (OTA) updates," *Vehicular Communications*, vol. 35, p. 100468, 2022.

[16] https://github.com/linux-can/can utils. (2020) Socketcan utilities. [Online]. Available: https://elinux.org/Can-utils

[17] KL. (2014) Kali linux. [Online]. Available: https://www.kali.org/

[18] Wireshark. (2018) wireshark and usbcap. [Online]. Available: https://www.wireshark.org/

[19] https://github.com/collin80. (2018) Canfuzzing and analysis tool. [Online]. Available: https://www.savvycan.com/

[20] I. Schieferdecker *et al.*, "Model-based security testing," *arXiv preprint arXiv:1202.6118*, 2012.

[21] I. Pekaric *et al.*, "Applying security testing techniques to automotive engineering," in *Int. Conf. Availability, Reliability and Security*, 2019, pp. 1–10.

[22] L. J. Moukahal *et al.*, "Vulnerability-oriented fuzz testing for connected autonomous vehicle systems," *IEEE Trans. Reliability*, vol. 70, no. 4, pp. 1422–1437, 2021.

[23] C. Miller *et al.*, "A survey of remote automotive attack surfaces," *black hat USA*, vol. 2014, p. 94, 2014.

[24] M. Wolf *et al.*, "Security in automotive bus systems," in *Work.Embedded Security in Cars*, 2004, pp. 1–13.

[25] S. Checkoway *et al.*, "Comprehensive experimental analyses of automotive attack surfaces." in *USENIX Security Symposium*, vol. 4, no. 447-462, 2011, p. 2021.

[26] M. Cheah *et al.*, "Towards a systematic security evaluation of the automotive bluetooth interface," *Vehicular Communications*, vol. 9, pp. 8–18, 2017.

[27] R. Falk *et al.*, "Electric vehicle charging infrastructure security considerations and approaches," *Proc. of INTERNET*, pp. 58–64, 2012.

[28] H. Christiansson *et al.*, *Creating a European SCADA security testbed*. Springer, 2008.

[29] C. Queiroz *et al.*, "Building a scada security testbed," in *Int. Conf. on Network and System Security*, 2009, pp. 357–364.

[30] C. Davis *et al.*, "Scada cyber security testbed development," in *North American Power Symposium*, 2006, pp. 483–488.

[31] D. S. Fowler *et al.*, "A method for constructing automotive cybersecurity tests, a can fuzz testing example," in *IEEE Int. Conf. Software Quality, Reliability and Security Companion (QRS-C)*, 2019, pp. 1–8.

[32] S. Marksteiner *et al.*, "A model-driven methodology for automotive cybersecurity test case generation," in *IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, 2021, pp. 129–135.

[33] M.Stefan *et al.*, "A process to facilitate automated automotive cybersecurity testing," in *IEEE Vehicular Technology Conference (VTC2021-Spring)*, 2021, pp. 1–7.

[34] T. Werquin *et al.*, "Automated fuzzing of automotive control units," in *Int. Work.Secure Internet of Things (SIOT)*, 2019, pp. 1–8.

[35] D. K. Oka *et al.*, "Integrating fuzz testing into a CI pipeline for automotive systems," Tech. Rep., 2022.

[36] K. Jost *et al.*, *Towards a Cyber Assurance Testbed for Heavy Vehicle Electronic Controls*, 2016, pp. 67–77.

[37] E. Lisova *et al.*, "Safety and security co-analyses: A systematic literature review," in *IEEE Annual Computer Software and Applications Conference (COMPSAC)*, vol. 13, no. 3, 2019, pp. 2189–2200.

[38] C. Wolschke *et al.*, "Saseval: A safety/security-aware approach for validation of safety-critical systems," in *IEEE/IFIP Int. Conf. Dependable Systems and Networks Workshops (DSN-W)*, 2021, pp. 27–34.

[39] E. Troubitsyna, "An integrated approach to deriving safety and security requirements from safety cases," in *Int. Computer Software and Applications Conf.*, 2016.

[40] J. Link *et al.*, "Current challenges of the joint consideration of functional safety & cyber security, their interoperability and impact on organizations: How to manage RAMS + S (reliability availability maintainability safety + security)," in *Int. Conf. Reliability, Maintainability, and Safety, (ICRMS)*, 2018.

[41] M. Koschuch *et al.*, "Safety & security in the context of autonomous driving," 2019.

[42] National vulnerability database(nvd). [Online]. Available: https://www.itemis.com/en/products/itemis-secure/documentation/user-guide/threatanalysisandriskassessment

[43] ISO Standard, *ISO 19014:Edition 1 - Earth-moving machinery – Functional safety*. [Online]. Available: https://www.iso.org/standard/70715.html

[44] S. Marksteiner *et al.*, "Integrating threat modeling and automated test case generation into industrialized software security testing," in *Central European Cybersecurity Conference*, 2019.

[45] D. S. Fowler *et al.*, "Towards a testbed for automotive cybersecurity," in *IEEE Int. Conf. Software Testing, Verification and Validation (ICST)*, 2017, pp. 540–541.

[46] C. Smith, *The car hacker's handbook: a guide for the penetration tester*. San Francisco:No Starch Press, 2016.

[47] VMware. (2014) Hypervisor. [Online]. Available: https://www.vmware.com/products/workstation-player.html