ENIGMA: An explainable digital twin security solution for cyber–physical systems

Suhail, S., Iqbal, M., Hussain, R., & Jurdak, R. (2023). ENIGMA: An explainable digital twin security solution for cyber–physical systems. *Computers in Industry*, *151*, Article 103961. https://doi.org/10.1016/j.compind.2023.103961

Published in:
Computers in Industry

Document Version:
Publisher's PDF, also known as Version of record

Computers in Industry 151 (2023) 103961

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.sciencedirect.com/journal/computers-in-industry

ENIGMA: An explainable digital twin security solution for cyber–physical systems

Sabah Suhail [a,*], Mubashar Iqbal [b], Rasheed Hussain [c], Raja Jurdak [d]

[a] Vienna University of Economics and Business, Austria
[b] University of Tartu, Estonia
[c] University of Bristol, UK
[d] Queensland University of Technology, Australia

ARTICLE INFO

Keywords:
Cyber-physical system (CPS)
Cybersecurity awareness
Digital twins (DTs)
eXplainable AI (XAI)
Gamification
Industry 5.0

ABSTRACT

Digital Twins (DTs), being the virtual replicas of their physical counterparts, share valuable knowledge of the underlying physical processes and act as data acquisition and dissemination sources to Cyber–Physical System (CPS). Moreover, without obstructing the ongoing operations, DTs also provide an assessment platform for evaluating the operational behavior and security of the CPS. Therefore, they become a potential source of data breaches and a broad attack surface for attackers to launch covert attacks. To detect and mitigate security loopholes in DTs, one of the potential solutions is to leverage a gamification approach that can assess the security level of DTs while providing security analysts with a controlled and supportive virtual training environment. Artificial Intelligence/Machine Learning (AI/ML)-based approaches can complement the idea of security orchestration and automation in the gamification approach. However, AI/ML-based DTs security solutions are generally constrained by the lack of transparency of AI operations, which results in less confidence in the decisions made by the AI models. To address the explainable security challenges of DTs, this article proposes a gamification approach called s**E**curi**N**g d**I**gital twins through **GaM**ification **A**pproach (ENIGMA). While leveraging DTs as an offensive security platform, ENIGMA provides gaming scenarios to assess DTs' security and train security analysts. The game players within ENIGMA are humans (the attacker team) and AI agents (the defender team). Furthermore, ENIGMA is supported by an eXplainable AI (XAI)-based DT security assessment model that explains the decisions made based on the SHAP values by the AI model on attack vectors for the defender team, i.e., the AI agent. The SHAP values illustrate the contribution of different features towards predicting the outcome of attack vectors. This explanation can help security analysts to take security measures based on reasoned and trustworthy decisions. Finally, experimental validation has been carried out to demonstrate the viability of ENIGMA.

1. Introduction

The integration of enabling technologies with Cyber–Physical Systems (CPSs) has led to the realization of Industry 5.0 (Maddikunta et al., 2022). CPSs involve a dynamic threat landscape that requires them to adapt to emerging threats without negatively affecting the ongoing operations (Eckhart and Ekelhart, 2019). In the information security domain, this requirement has fueled research into Digital Twins (DTs). DT generally refers to a virtual (digital) representation of its corresponding physical counterparts, although more specific definitions of DTs target CPS security (Eckhart and Ekelhart, 2019). The implementation of DT depends on the available resources and technologies. Furthermore, it also depends on whether a physical system pre-exists or DT is envisioned before its physical counterpart. In the former case, DTs are leveraged during the operational and maintenance phases of the physical system, whereas in the latter case, DTs serve as a design model for the physical system. DTs utilize real-time and historical data from multiple sources (Tao and Zhang, 2017), such as sensor-based data, asset state data, system lifecycle data, contextual information, domain-specific knowledge, other DT instances, and so on. DT inspects for data inconsistencies through two of its services: (1) by simulating disjointed instances that are not connected to the physical system, referred to as *simulation mode*; and (2) by continuously mapping the physical and twin states of the system, referred to as *replication mode*. The optimized parameters are fed to the physical system to detect inconsistencies (Tao and Zhang, 2017), creating a closed loop between the physical and twin state of the system. Furthermore, instantiating and aggregating

* Corresponding author.
E-mail address: sabah.suhail@wu.ac.at (S. Suhail).

https://doi.org/10.1016/j.compind.2023.103961
Received 4 March 2023; Received in revised form 5 May 2023; Accepted 29 May 2023
Available online 14 June 2023

multiple DT instances can provide localized information on cascading inconsistencies, to determine which of the interconnected subsystems is causing the failure of the other subsystems.

To date, sophisticated attacks have targeted many critical infrastructures, such as energy and utilities, that caused severe service disruption (Lee et al., 2017). For instance, the *Stuxnet* (2010) malware that targeted the Iranian nuclear power plant (Langner, 2013), the *Industroyer* (2016) malware that targeted the Ukrainian power grid system (Lee et al., 2017), and the *Triton* (2017) malware that targeted the petrochemical plant in Saudi Arabia (Miller et al., 2019), are among the seminal examples of cyberwarfare in Industrial Control Systems (ICSs). The data from CPS, i.e., sensor-based data or asset state data, is one of the primary input data sources to the DTs. The malware-affected CPS produces erroneous data, and affects the quality and accuracy of DTs, thereby leading to invalid decisions provided by the DT services. This can be the first and foremost stage towards cascading failures.

DTs, as a *security-enhancing enabler*, are leveraged in two ways: data sharing; and state mirroring. First, DTs act as the primary source of data acquisition from and dissemination to various other data sources, creating a bidirectional asset-centric data flow between the physical system and its virtual counterpart. For example, physical assets, Knowledge Base (KB) that comprise process knowledge for DT modeling, and DT instance(s) rely on DTs as the data source. Moreover, a digital thread that links data across product lifecycle phases also obtains data from DTs. Process knowledge refers to the system's specification data, including the set of rules, connections, flows, states, and historical data for which the DT is modeled. Based on the data from the aforementioned sources, DTs further evolve and optimize their functionalities and eventually the services of the associated sources, such as physical assets with each lifecycle phase, i.e., engineering, operation, and decommissioning (Eckhart and Ekelhart, 2019). Second, based on the system's knowledge, DTs precisely mirror the states of their physical counterparts to provide DT services, including system testing, instance simulation, state replication, and predictive maintenance (Eckhart and Ekelhart, 2018b; Tao and Zhang, 2017). The aforementioned desirable characteristics of DTs make them a potential source of data breaches, leading to the abuse case of DT (Eckhart and Ekelhart, 2019). Attackers may exploit the deep knowledge about the physical process and devices accessible through DTs with a two-stage strategy: (1) place the key data acquisition and dissemination source, namely DT, into a malicious state, and (2) then through that state manipulate the underlying physical system's behavior covertly. Such strategy holds contrariwise, i.e., targeting CPS to attack DTs. For example, the evolution in the tradecraft of ICS-tailored malware represents that adversaries fully understand and codify the knowledge of physical industrial processes (Lee et al., 2017). With in-depth knowledge of physical industrial processes, the adversaries can easily infer and construct their knowledge about DTs. This can be the second stage towards cascading failures.

Among other security-enhancing use cases, such as intrusion prevention and detection and honeypots, leveraging DTs as a security-awareness learning platform has also been suggested (Eckhart and Ekelhart, 2019). For example, Vielberth et al. (2021), Zhou et al. (2022) proposed utilizing DTs as cyber ranges (which was conceptually proposed in 2018 by Bécue et al. (2018)) to provide hands-on cyber skills and security posture testing. However, without justifying the security and trustworthiness of DTs, it may raise the argument: *if DTs are themselves malicious, then using them as a solution to complement security solutions will be irrational*, which could be an ideal example of a DT abuse case.

The existing works on DT as a learning platform (Bécue et al., 2018; Vielberth et al., 2021; Zhou et al., 2022) have not realized the potential of harnessing DTs beyond the learning environment. The existing works on gamifying DTs lack the following elements: (i) have not investigated leveraging DTs as an offensive security platform (i.e., following an adversarial approach to protect the system), which undermines their use of DTs for security training and testing without considering the security aspects of DTs themselves, (ii) have not considered integrating Artificial Intelligence/Machine Learning (AI/ML)-based approaches on DT-based gamification platforms thus fails to integrate solutions that can provide security automation, and (iii) the absence of eXplainable AI (XAI)-based solutions in the gamification approach hinders the ability to justify security solutions. These research gaps have opened up a new aspect for gamifying DT security by using an offensive security approach where DT can support a multi-objective platform, including (i) a security-aware learning environment, (ii) automated security testing, and (iii) explainable DT assessment for security analysts.

To address the research gaps, we propose ENIGMA (s**E**curi**N**g d**I**gital twins through **G**a**M**ification **A**pproach) framework (as Fig. 1 shows). The DT part of Fig. 1 is, at least in part, the generic framework of CPS-DT also discussed in Eckhart and Ekelhart (2018a). ENIGMA leverages DT as an offensive security solution by launching overt attacks on DT instance(s) in a simulated, interactive, and controlled environment, hence called *gamification*. ENIGMA serves a two-fold purpose: (i) assess the security level of DTs and indirectly of the CPS, and (ii) provide security analysts with an explainable security-awareness training environment. Through an AI/ML approach, based on Deep Neural Networks (DNN), ENIGMA provides security orchestration in the DT-based gamification platform. Furthermore, ENIGMA supports XAI for security decisions using SHapley Additive exPlanations (SHAP) for post-hoc explanation, which identifies the features relevant to the outcome of the AI model. As a result, ENIGMA provides justifiable decisions by reasoning *what*, *why*, and *how* specific cybersecurity defense decisions are made in a gaming context.

ENIGMA framework comprises three key components, including (i) CPS-DT environment, (ii) gamification platform, and (iii) XAI-based DT security assessment module. The DT's building blocks comprise physical assets (device or process) as CPS, the corresponding DT environment that performs the digital-physical mapping between a physical asset and its digital replica, data sources, such as sensors and historical, DT modeling to build process knowledge of DTs, and DT services. The CPS-DT environment is vulnerable to attacks. The DT environment aids in instantiating re-producible interactive gaming scenarios for the gamification platform. The gamification platform provides a virtual training environment for security analysts. With Capture-The-Flag (CTF), i.e., the attack–defense approach, the gamification environment can emulate attack scenarios without affecting the ongoing CPS-DT processes. The game players, human and AI agents, are also provided with learning resources. Moreover, the gamification platform accesses gaming outcomes based on the XAI module that consists of conventional DNN learning stages followed by explanations of the outcomes to identify the most relevant features underpinning security decisions. To summarize, our main contributions are:

- analysis of CPS and DT attack targets,
- proposal of the ENIGMA framework for gamification of DT security with XAI, and
- implementation and validation of ENIGMA through a proof of concept, using DNNs and SHAP.

The rest of the paper is organized as follows. Section 2 presents the related work. Section 3 covers potential threats to CPS and DTs. Section 4 explains the components and working of the ENIGMA framework. Section 5 demonstrates a proof of concept implementation. Section 6 discusses limitations and future research directions. Finally, Section 7 concludes the paper.

2. Related work

As the deployment and maintenance of testbeds for analyzing dynamic cybersecurity threats and attacks are time- and cost-intensive (Antonioli et al., 2017), DTs have gained significant attention as cost-effective, reconfigurable, and reproducible simulation environments for security evaluation of CPS (Eckhart et al., 2019). To date, DTs as a

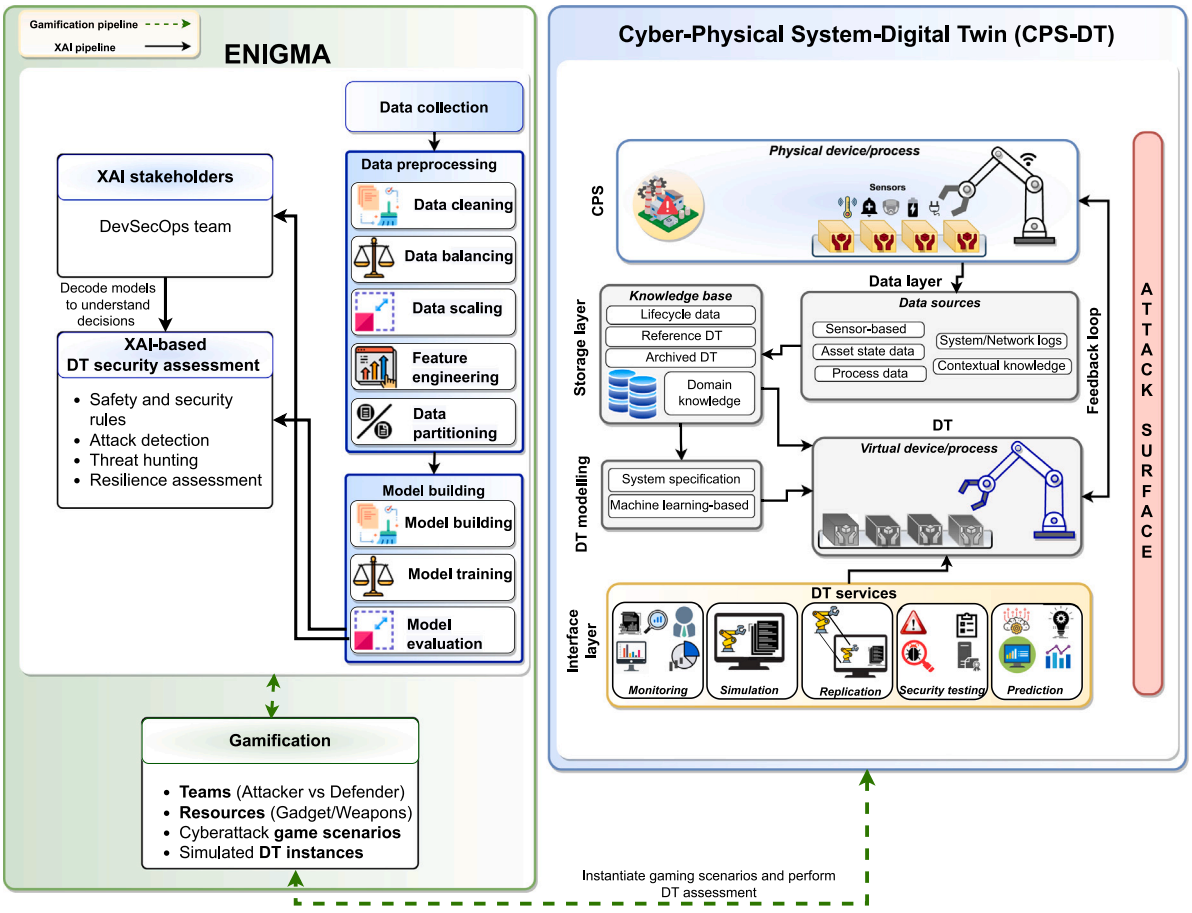


Fig. 1. ENIGMA: An explainable DT security solution for CPS through gamification approach.

security-enhancing enabler have been used in intrusion prevention and detection (Eckhart and Ekelhart, 2018b,a; Suhail et al., 2022b), or identified as a way forward in honeypots (Eckhart et al., 2019; Dietz and Pernul, 2020).

Another security-enhancing use case of DTs can be realized through DT-based cyber ranges. DT-based cyber ranges support gamification platforms to provide a training environment for cybersecurity professionals. The study (Bécue et al., 2018) conceptually proposed DT-based cyber ranges to evaluate DT's weaknesses and cyber-breaches. However, the proposed work lacks details on the realization of the gaming environment and hence practical implementation. Vielberth et al. (2021) provided a prototypical implementation for training security analysts. In this work, the authors use MiniCPS (Antonioli and Tippenhauer, 2015) to generate a virtual training environment to simulate attack scenarios of ICS. Zhou et al. (2022) also implemented DT-based cyber range for the Industrial Internet of Things (IIoT). However, the work does not focus on incorporating gaming elements. Table 1 summarizes the existing works based on features including gaming elements (such as objective-based games, teaming, resources, etc.), security-awareness learning, security automation incorporated through AI/ML, explainable decisions incorporated through XAI, and assuring the platform security incorporated through offensive security.

In the DT-based cyber range game scenarios, simulations can be directly instantiated from DTs or tailored to the cyber range (Vielberth et al., 2021). Such adoption fails to consider the abuse case of DTs (Eckhart et al., 2019) and leverages DTs as training platforms without ensuring the trustworthiness and reliability of the security awareness learning/training platform, i.e., DTs. More specifically, the existing works proposed DTs as a security-enabler and/or security-awareness solution without first confirming the security of DTs. To sum up, the existing works lack security-awareness learning environment, gaming elements, automated security testing, and explainable DT assessment for security analysts.

Exploiting DTs for launching attacks was mentioned as an open research (Eckhart and Ekelhart, 2019), and the study (Alcaraz and Lopez, 2022) provides a comprehensive survey of the potential threats to the DT paradigm. A potential solution to detect and mitigate security attacks on the CPS-DT ecosystem can be blockchain-based DTs (Suhail et al., 2022a). However, blockchain technology does not guarantee the trustworthiness of data at the data source (Dedeoglu et al., 2020).

By narrowing down our research to gamification-based DTs, we can conclude that the existing works on gamifying DTs have not investigated the potential of leveraging DTs as an offensive security platform. More specifically, such DT-based gamification approaches have used DTs for security training and testing without considering the security aspects of DTs. Furthermore, the existing gamification approaches lack XAI-based solutions to justify security solutions that enable security awareness and automation.

3. Security attacks on CPS-DT ecosystem

Like many other technologies, DT can become a double-edged sword where we can leverage DT to safeguard CPS against cyberattacks but on the other hand, DT itself can be an attractive target for cyberattackers. It is important to ensure that the benefits of using DT outweigh its risks as a target of cyberattacks. In this section, we discuss different attack strategies on the CPS-DT ecosystem. Usually, the attack strategies can be demonstrated in various ways, including Cyber Kill Chain (CKC) or attacks on different layers, such as the physical, data, network, or application layers. For instance, Alcaraz and Lopez (2022) carried out

Table 1
DT as a learning platform: A summary.

Research work	Concept/proof of concept (PoC): use case	Simulation tool	Features				
			Gaming elements	Security-awareness learning/ training	Security automation	Explainable decisions	Platform security assurance
Bécue et al. (2018)	Concept: Industrial Robot	–	×	◑	×	×	×
Vielberth et al. (2021)	PoC: Filling plant	MiniCPS	✓	✓	×	×	×
Zhou et al. (2022)	PoC: Petrochemical industry	Unity3D	×	◑	×	×	×
ENIGMA	PoC: Vehicle	ADT	✓	✓	✓	✓	✓

Legend: ✓ Covered; × Not covered; ◑ Partially

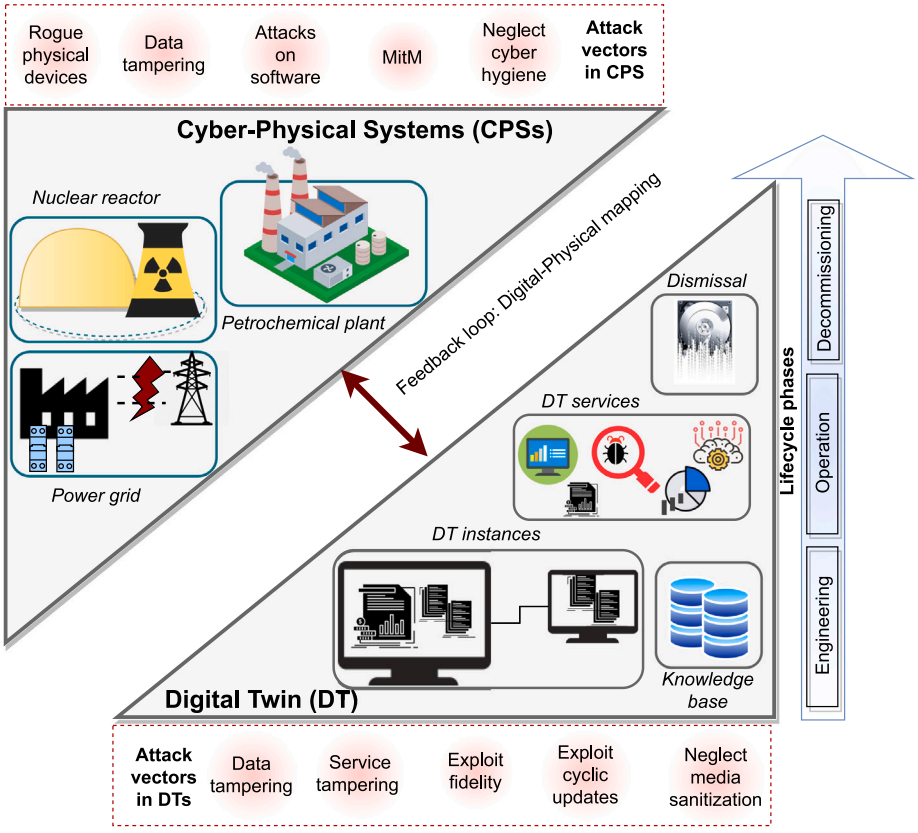


Fig. 2. Potential security attack on CPS-DT ecosystem.

a comprehensive survey on the security threats in a layered structure, i.e., a data dissemination layer, data management and synchronization layer, data modeling layer, and data visualization layer. They particularly take into account the CPS/IoT, computing infrastructure, virtualization systems, computing techniques, and Human–Machine Interfaces (HMIs) and identify attacks such as software attacks, privilege escalation, Denial of Service (DoS), data extraction, and other similar attacks that are common throughout the proposed layers in Alcaraz and Lopez (2022). Due to the fact that we are leveraging DTs as a security-enhancing platform, we look into the security attacks from the perspective of the source of the data, the services provided by the DT, its design aspect, and the different stages of the DT lifecycle. Details are provided in the following subsections and summarized in Table 2.

To this end, the potential attacks on CPS-DT are divided into three classes, i.e., attacks on CPS, attacks on DT, and hybrid attacks. A pictorial representation of the attack points related to the CPS-DT ecosystem can be found in Fig. 2. Fig. 3 depicts a taxonomy of attacks on the CPS-DT ecosystem and Table 2 summarizes the attacks by identifying the targets and the source(s) of exploits. In the following, we provide details of the potential attacks on CPS (Section 3.1), attacks on DTs (Section 3.2), and hybrid attacks (Section 3.3).

3.1. Attacks on CPS

CPS is a complex-connected network of different devices and systems that have interfaces for control and configuration. More precisely, CPS has two main components, physical assets, and the cyber domain which enables interfaces to the physical assets. The attacks on the CPS have different scopes, ranging from only cyberattacks to physical and cyber–physical attacks as described in Kayan et al. (2022). Without loss of generality, we discuss the attacks launched by insiders and outsiders that cover the cyber and physical spaces of the CPS.

3.1.1. Attacks on CPS physical assets

The CPS-DT ecosystem encompasses the physical CPS, the communication backbone, and the digital infrastructure for the twin. One

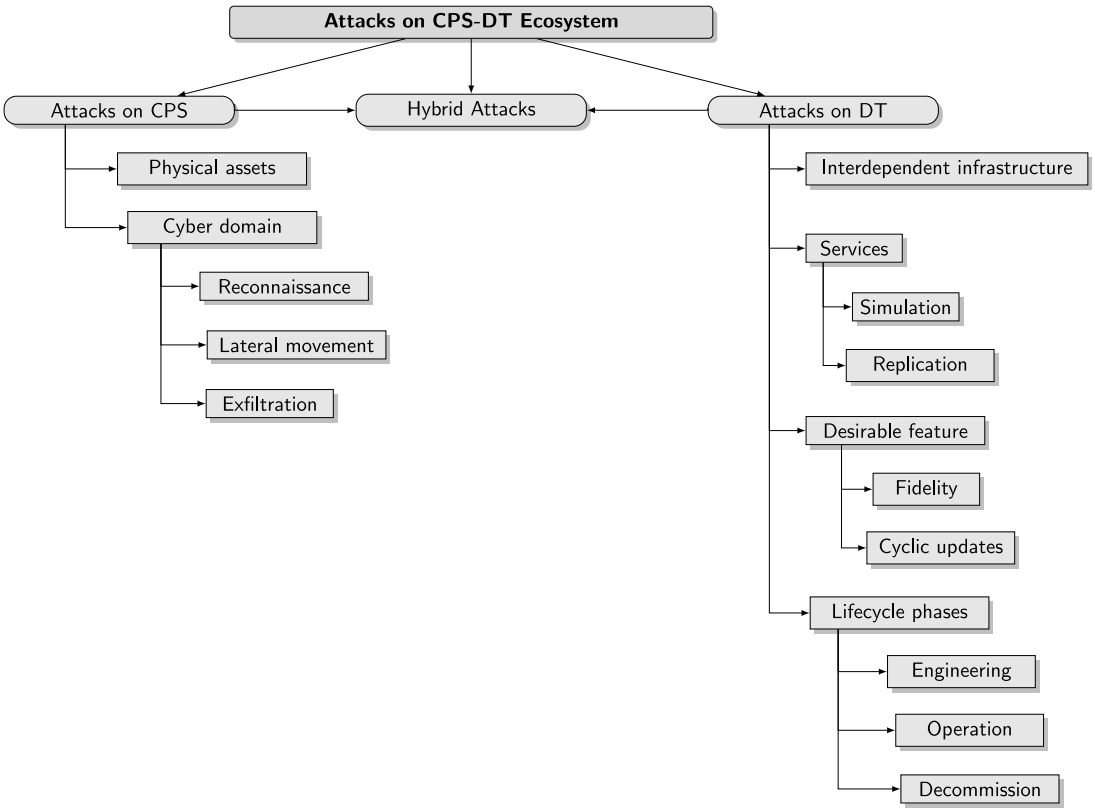


Fig. 3. Taxonomy of attacks on CPS-DT ecosystem.

Table 2
Summary of attack strategies on CPS-DT environment.

Target / Exploit	CPS-DT environment		Attack goal
	CPS	DT	
Data source	Sensors, gateways, PLCs		Compromise devices (e.g., rootkit in firmware and malicious update)
	Knowledge base		Tamper process knowledge, domain knowledge, or historical data
Services		Simulation	Learn system behavior by re(running) instances and manipulating user-specified parameters
		Replication	*Active:* Delay data/state synchronization to cause desynchronization
			Passive: Observe and learn system behavior
Design aspect		DT Modeling (fidelity)	Gain deep knowledge of the system/process, and exploit the degree of similarity between the physical system and its twin
Attack Stages	Engineering and/or Operation		Exploit digital thread, Gain control over high-value assets
	End-of-life/obsolete	Decommission	Retain knowledge about the system's life from archived DT by security breach and neglect of media sanitization

of the essentially available attack vectors on the CPS is the devices in the CPS environment where sensors, gateways, and Programmable Logic Controllers (PLCs) are modified or tampered with. More precisely, attackers could try to modify the logic of PLCs by compromising the interfaces, which may eventually cause these devices to deviate from the actual expected function. Furthermore, the attackers can inject or install malware and other malicious pieces of software into these devices. Sensor tampering at a very low level (for instance physically tampering with a sensor such as covering it with a material) may also contribute to the modification of the PLC logic, which eventually contributes to the dysfunction of the CPS.

3.1.2. Attacks on CPS cyber domain

Other essential elements involved (and then lead to) attacks on CPS include reconnaissance, privilege escalation, and data tampering in transit and/or at rest, to name a few.

Reconnaissance: Reconnaissance is the first step for the attackers that involves intelligence gathering. This is achieved through activities such as network scanning, exploiting zero-day vulnerabilities, and enumerating services to identify security loopholes in the underlying system. However, the attacker may go beyond conventional network reconnaissance in industrial ecosystems to achieve the desired objectives. Sophisticated malware can easily defeat isolation mechanisms

(as evident from real-world events), including air gaps, sandboxes, virtualization, and so on (Lee et al., 2017; Dietz and Pernul, 2020).

Lateral movement: After reconnaissance, the next natural move of the attacker is the lateral movement to dig deeper into the CPS network in the cyber domain. The attackers target privilege escalation to increase the level of control over devices or to gain access to more resources than they are normally entitled to. For instance, privilege escalation attacks can be caused by outsiders and insiders in the system by exploiting the system's misconfiguration, a bug, weak access control, or obtaining stolen credentials. These phenomena are common in generic CPS, and in reality, the edge devices connected to CPS are the perfect targets for such attacks. These attacks have a range of vectors, including social engineering, malware, software and hardware misconfiguration, and so on (Kayan et al., 2022).

Exfiltration: Data breach and exfiltration are often considered the last step in the attack chain. Depending on the scope of the attack, the attacker might be interested in stealing the sensor data passively without tampering, which could be leveraged for future malicious activities or espionage. On the other hand, the attacker might modify the data generated by different sensors and launch further attacks on the CPS. Furthermore, from the active attack's perspective, through reconnaissance, lateral movement, privilege escalation, and data tampering, the attackers may get hold of the physical system through the interfaces and both expose the CPS to further attacks and disrupt the normal CPS functionality. DT infrastructure may be hosted locally at the premises of the CPS or it can be hosted remotely under a different administrative domain. Therefore, the edge components of the CPS-DT ecosystem could be either at the CPS boundary, at the DT boundary, or both, depending on the underlying network infrastructure and the ownership of the CPS-DT ecosystem.

3.2. Attacks on DT

In this subsection, we discuss potential attacks on the DT and related infrastructure.

3.2.1. Exploiting vulnerabilities of interdependent DT infrastructure

Attacks on DT mainly target data and services. More precisely, these attacks are aimed at the core of DT, which includes KB. KB is essential for the DT modeling and design, and from the infrastructure standpoint, in principle, these attacks target both edge (at the DT) and the cloud that constitutes the DT. In the existing DT realization, cloud infrastructure is extensively used. In other words, to model a DT of an asset or a system, the size and scale of the asset demand sufficient resources. For instance, in the case of CPS, DT would need significantly large computing, storage, and communication resources. Therefore, cloud infrastructure with rich support for virtualization can be a natural choice. Thus, the cloud infrastructure in DT inherits all the attacks on the traditional cloud infrastructure. Therefore, it is of paramount importance to focus on data security pertaining to process knowledge in the cloud infrastructure. The edge and fog paradigms complement the use of cloud infrastructure for the end user. However, the convergence of edge–fog–cloud opens a plethora of other attacks such as malicious code and command injection, access control abuse, Distributed DoS (DDoS), and so on (Ometov et al., 2022).

3.2.2. Exploiting DT services

DT can run in either simulation or replication mode. The replication mode of DT can operate as active or passive monitoring (Eckhart and Ekelhart, 2018a). The main difference between both modes is the state mapping time period, which depends on underlying application requirements (Minerva and Crespi, 2021). To run DT in a replication mode, twins and their physical counterparts must be continuously connected and synchronized through sensor measurements, network communication, process/device state data, or log files (Eckhart and Ekelhart, 2018b) to register any change. The replication mode may initiate direct cyclic updates to and from the DTs due to automatic feedback loops. However, to launch an attack on DT running in the replication mode, the attacker needs to stay active to avoid the problem of time-dependent state synchronization, i.e., consistency of state and data between the physical object and its different replicas. The state/data synchronization period can be defined according to the application requirements.

Simulation mode operates in an isolated environment without having a direct connection to the live systems (Eckhart and Ekelhart, 2018b). It requires user-specified settings and parameters as input to the DT for modeling the simulated environment of the physical asset. The simulation mode, being reproducible (Dietz and Pernul, 2020) with a broad range of trial-and-error learning mechanisms, can be directly used or tailored by the attacker according to the attacker's needs. In other words, the attacker can reveal emergent system behaviors by resetting and re-running the simulation until the insidious goals are achieved. In the worst case, it can target the theme of simulation mode—security-by-design (Dietz and Pernul, 2020) by reversing the defined configurations (i.e., sensor data thresholds) during security tests within the virtual environment. Furthermore, the attacker can passively learn the system state, such as gathering information about its configuration and operations. However, the attacker cannot trigger automated attacks on the CPS due to the absence of a direct feedback loop.

3.2.3. Exploiting desirable features of DT

The fidelity of DTs can also play a key role in the attacks on the CPS-DT ecosystem. A precise representation of DTs, i.e., fidelity, is essential in the modeling of DTs. With sufficiently high-fidelity twins, the attackers can compromise the overall design and functionality of the original asset. Considering the virtual representation in DTs that mimics the functionality of the corresponding physical asset (processes or devices) to a certain level of detail, reasonable feature generalizations or simplifications can occur as long as they stay context-aware (Minerva and Crespi, 2021), i.e., the level of fidelity will be variable. More precisely, the purpose of building DTs is to provide a cost-effective solution to test the physical system rather than replicating (in terms of simulation or emulation) the system. Nevertheless, accurate representation of DTs also exposes the system to the likelihood of a successful attack.

Another desirable feature of leveraging DTs is to support system automation. Note that the cyclic state updates are synchronously reflected from the DTs to the physical system or vice versa during the operation phase of the system. Though such actions aim to optimize the underlying operations, however, doing so, attacks instigated through DTs may have similar repercussions as that of directly attacking real field devices, especially under automated security testing use case (Eckhart and Ekelhart, 2019).

3.2.4. Exploiting data during lifecycle phases

During the engineering and design phase, process knowledge from multiple sources such as domain knowledge, data links, rules, historical information, and ML-based knowledge can be subject to attacks such as data poisoning, leakage, theft, and so forth. Similarly, a digital thread can also be subject to a manipulation attack, where the updates to the state of a DT could put the DT in a malicious state (Eckhart and Ekelhart, 2019). In other words, updates in one instance of a DT can have a cascading effect on other instance(s).

Usually, due to obsolescence or replacement of outdated hardware/software, the asset and its twin are destroyed during the decommissioning phase (Barricelli et al., 2019). Nevertheless, considering the fact that the knowledge about the predecessor system can be rehashed, DT data could be backed up and used by similar objects or domain experts to optimize the next generation of the system. No matter whether the DTs are destroyed or retained for future usage, attackers might exploit them (Eckhart and Ekelhart, 2019). For instance, DTs can be attractive targets of data breach incidents if they are not carefully

archived for future usage or disposed of while complying with proper media sanitization guidelines (i.e., removing data from storage media, such that the data may not be easily retrieved), for instance, NIST 800-88 guidelines.

3.3. Hybrid attacks

In addition to the attacks on CPS or DT in isolation, the attackers may target both CPS and DT concurrently through hybrid attacks. To do so, attackers either leverage the knowledge of the physical industrial process or exploit knowledge about the physical industrial process accessible through the DTs. Choosing a physical system or a DT (as targeted entities) to gain deep understanding depends on: (i) which entity (i.e., CPS or DT) exists first, (ii) whether it is easily susceptible to cyberattacks, or (iii) whether it provides ideal opportunities to live off the land. Depending on the motivation and capabilities of the attackers, which can also involve colluding with insiders, attackers can establish a persistent foothold at the physical system and DT simultaneously. Under such sophisticated settings, attackers are able to command and control both physical, i.e., CPS, and DT, which can be further strengthened due to automatic feedback loops enabled between CPS and DT. Following a joint attack strategy, attackers controlling the physical environment trigger the changes in the physical system, while concurrently, the attacker controlling the DT environment manipulates the pre-defined parameters. Therefore, even if they get caught while feeding adversarial data into the physical system, attackers may still exploit vulnerable entry points into DTs.

4. ENIGMA platform

A well-versed adversary can attack DTs to steer the CPS into an insecure state by, (i) manipulating the benign behavior of DTs, (ii) exploiting the desirable characteristics of DTs, or (iii) causing state inconsistencies between the digital and physical environments to affect the quality and accuracy of DTs. To thwart attacks on DTs, one potential solution is to assess the security level of DTs by launching attacks on them (Bécue et al., 2018). However, such DT assessments must be performed in an isolated environment without negatively affecting the operation of DT replication mode, where it is critical to register changes between the states of the twin and its physical counterpart to avoid invalid interpretations. To this end, we propose gamifying DT security by using an offensive security approach where DT can support a multi-objective platform, including a security-awareness learning environment, automated security testing, and explainable DT assessment for security analysts.

This section describes the concept behind DT-based gamification (Section 4.1), the components (Section 4.2), and operation (Section 4.3) of ENIGMA while mapping to the in-vehicle traffic data as the underlying use case of the simulated game scenario. The key acronyms are listed in Table 3.

4.1. DT-based gamification

Gamification is the process of incorporating game mechanics into non-game environments (Wood and Reiners, 2015). The gamification in the cybersecurity domain can provide security education to security professionals through a simulated yet interactive hands-on learning/training environment (Corallo et al., 2022). Mapping the gamification idea to CPSs security beyond learning objectives can help in investigating the reasons for service degradation or disruption, or assessing the resilience of the physical processes or devices against overt attacks. However, realizing the hypercritical nature of high-valued DT design artifacts and the need for trustworthy DTs for CPS, we choose to gamify DTs rather than CPS. Since the DT simulation mode supports instance(s) of virtual replicas of the physical processes or devices, the simulated gaming scenarios can be readily (re)produced and available for gamification. Moreover, instantiating the use cases from replication mode can help the DevSecOps team to simulate game scenarios about the system's emergent behavior. With the gamified nature of CTF, games can be conducted between attacker and defender (human and AI agents). The fact that most of the security solutions (e.g., attack detection, intrusion detection, malware detection, and so on) involve some form of AI techniques, we employ AI agents in the gamification approach. However, the decision taken by the AI agent must be understandable and explainable to the relevant stakeholders, i.e., security analysts in our case. Therefore, we rely on the algorithmic interpretability-driven explanation, also referred to as a post-hoc explanation in ENIGMA. To this end, we leverage SHAP as a post-hoc explanation mechanism in ENIGMA. The reason for using SHAP is to identify the features that are relevant to the outcome of the AI model.

4.2. Components of ENIGMA platform

The ENIGMA platform in Fig. 4 describes the details of high-level components (presented in Fig. 1) and their connections. We based our gaming platform on the work proposed in Vielberth et al. (2021) with significant differences in terms of game players (human and AI agents), offensive security strategy, and explainable decisions.

The ENIGMA platform comprises the following components: (1) management console, (2) teams, (3) gaming resources, (4) simulated gaming scenarios, and (5) XAI-based DT assessment. A management console (*step 1*) controlled by the DevSecOps team that defines objective-based game scenario(s), chooses the configuration of the teams made up of attackers or defenders and assigns them roles and resources. With the attack–defense CTF approach, there are two teams (T_H and T_{AI}), red team (the attackers) (*step 2*), and blue team (the defenders). In our use case, we consider humans (security analysts) as attackers (T_H) and AI agents (Generative Adversarial Networks (GANs)) as defenders (T_{AI}). However, the attacker–defender roles can be switched between human and AI agents based on the underlying security objective. Without loss of generality, the attacker can also be an AI agent that generates attack vectors automatically. In our case, we consider humans as attackers for ease of understanding. Teams are provided with cyber weapons and cyber gadgets (*step 3*) as gaming resources ($G_{resources}$) such as domain knowledge (videos, manuals, or instructional texts), services, utilities, and tools (e.g., testing and reconnaissance tools) to assist teams. For example, the attacker team can be provided with malware such as a backdoor and can infect firmware, external onboard diagnostics devices, Bluetooth, telematics, or exploit other connected devices. In our use case, the Controller Area Network (CAN) bus data of a vehicle can be infected by the attacker. The defender team can be provided with adversarial state information acquired from sources such as MITRE ATT&CK (Adversarial Tactics, Techniques, and Common Knowledge) (knowledge base of the adversary) and a runbook which is a step-by-step guide for completing complex operations. Both teams must have equal access to the domain knowledge, services, utilities, and tools.

4.3. How does ENIGMA operate?

We denote a physical object (vehicle, in our case) as $P_o = \{p_1, p_2, \ldots, p_n\}$, where an object is fully characterized by a set of sub-objects (engine, steering system, suspension, wheels, brake, Electronic Control Units — ECUs, and sensors, etc.) along with their data values and/or states. P_o can be fully realized as a DT model T_o with (sub)instances, i.e., $\{t_1, t_2, \ldots, t_n\}$.

Definition 1 (*DT Model*).
A system specification-based instance $t_i \in T_o$ can be defined as a triplet:

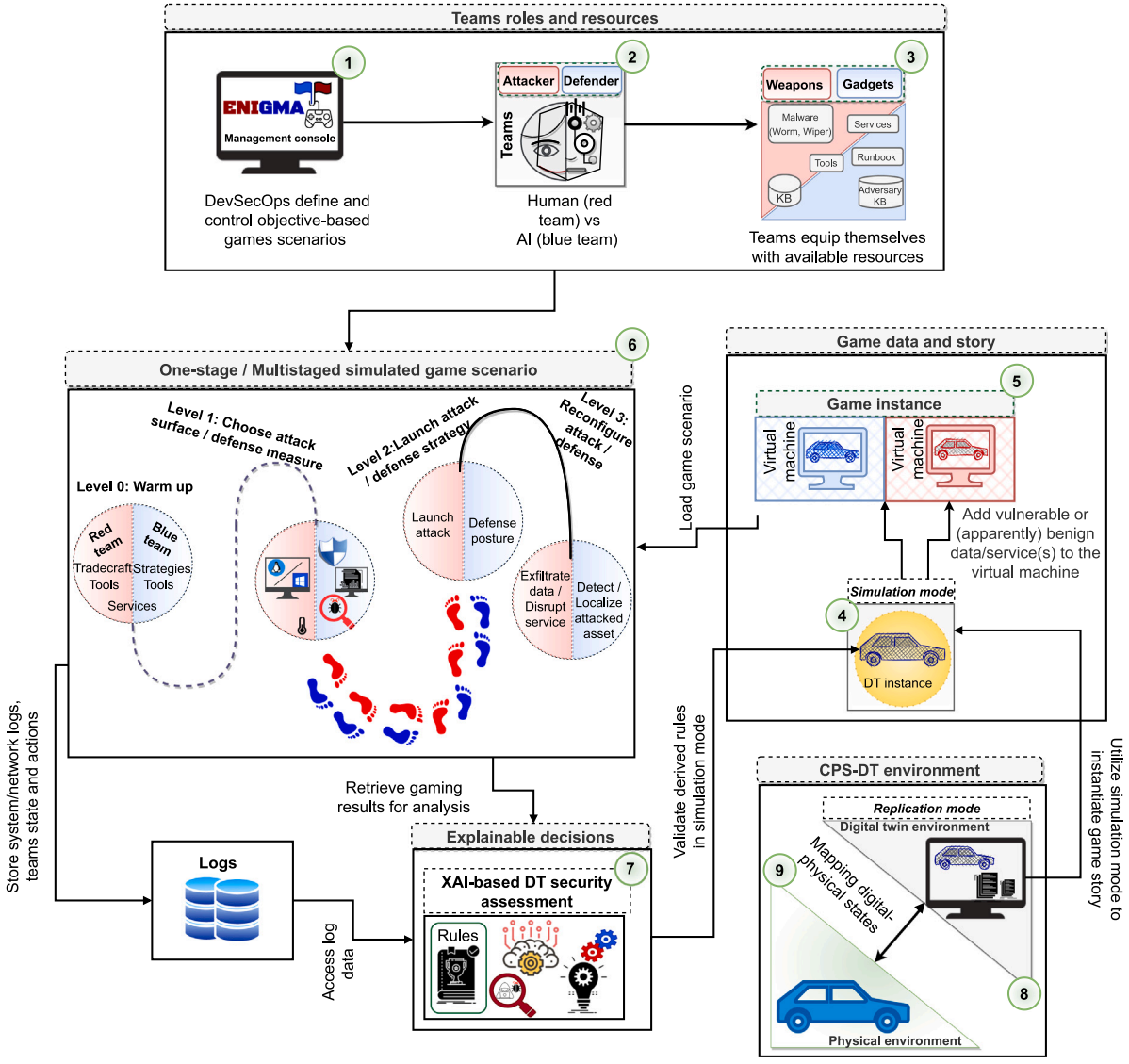


Fig. 4. Gamifying in-vehicle network traffic: ENIMGA use case.

Table 3
Notations and their explanation.

Notation	Explanation	Notation	Explanation
ID_{t_i}	(ith) Twin object ID	$D_{t_i,time}$	Timestamped data of (ith) twin object (sensor or device)
$State^{\omega}_{t_i}$	(ith) Twin object (current) state	f	function (steps: ingest, transform, feed)
P_o	Physical object	T_o	Twin object
χ_{config}	System configuration	χ_{spec}	System specification
T_H	Human team	T_{AI}	AI team
G_{story}	Game story	G_{data}	Game data
$G_{resources}$	Game resource	$G_{results}$	Game results
$logs$	logs of game	τ	Threshold
$X_{decision}$	Explainable decision	C	Set of challenges

$$t_i = \langle\, ID_{t_i},\ f\left(D_{t_i,time}\right),\ \chi_{spec}\ \rangle \tag{1}$$

D_{t_i} can be obtained either from digital-physical mapping, i.e., $p_i \mapsto t_i$ or telemetry data (in our case) through f (a three-step function) as follows:

$$\begin{aligned} f^{ingest}_{s_1} &: D_{telemetry}, && \text{obtain telemetry data} \\ f^{transform}_{s_2} &: D_{telemetry} \mapsto D_{t_i,time}, && \text{transform data} \\ f^{feed}_{s_3} &: t_i \leftarrow D_{t_i,time}, && \text{feed data to twin} \end{aligned} \tag{1a}$$

and χ_{spec} is:

$$\chi_{spec} = \langle State^{\omega}_{t_i}, \tau^{\mathbf{max}}_{t_i}, \tau^{\mathbf{min}}_{t_i}, classifier \rangle \quad \text{specification data} \tag{1b}$$

$classifier$ (binary) denotes $\{Attack, N\text{-}Attack\}$.

Definition 2 (*Game Instance*)**.**
A $G_{instance}$ initiated from $t_i \in T_o$ can be formulated as:

$$G_{instance} = \langle G_{story}, G_{data}, \chi_{config} \rangle \tag{2}$$

G_{data} can either be a publically available dataset or directly obtained from t_i.

$$G_{story} = \langle C_i, timespan \rangle \tag{2a}$$

G_{story} contains C_i to be accomplished for a given *timespan* by respective (T_H, T_{AI}) teams.

$$\chi_{config} = \langle AzureADTcredentials, ADTinstanceURL, AzureEventGrid, Storage \rangle \tag{2b}$$

χ_{config} represents DT system's configurations during which $G_{instance}$ is active.

Definition 3 (*XAI-Based DT Security Assessment*)**.**
$X_{decision}(\mathbf{SHAP}_{values})$ can be explained as:

$$\mathbf{SHAP}_{values} \leftarrow G_{results}, logs \tag{3}$$

$$classifier : X_{decision}(\mathbf{SHAP}_{values}) \mapsto \{Attack, N\text{-}Attack\} \tag{3a}$$

where $X_{decision}$ depends on $\mathbf{SHAP}_{values}$ values to classify the decision as $Attack$ or $N\text{-}Attack$.

To initiate the gaming environment (as *step 6* in Fig. 4 shows), a DT instance can be instantiated from the DT simulation mode (*step 4*). A DT instance serves as a gaming instance ($G_{instance}$) and is loaded into the Virtual Machine (VM) (*step 5*). t_i is instantiated from the DT model (T_o) that represents the complete virtual replica of the physical object (P_o). $G_{instance}$ comprise game story (G_{story}) and game data (G_{data}). A game story (G_{story}) defines a set of challenges (C_i) to be accomplished by respective teams, i.e., T_H and T_{AI}. The game data (G_{data}) consists of datasets with intentional vulnerabilities or apparently benign data. Given the fact that ENIGMA can be used to evaluate the security of DTs, the DT instance(s) can be directly loaded to a VM, i.e., without adding any vulnerabilities. This strategy serves the purpose of penetration testing. The purpose of using VM is to emulate the functionality of the desired t_i subjected to $G_{instance}$ without affecting ongoing processes, i.e., digital-physical mapping in the replication mode.

The next step is to start the game, which may consist of one-stage or multistage simulated gaming scenarios (*step 6*). Depending on the security objectives and players, such as security analysts from industry or researchers from academia, defined in the game story, the stages may vary. For example, a multistage game may consider multi-rounds with increasing complexity. A one-stage game may consist of a data or service tampering attack and a respective defense game. In general, the games can be engaging and interactive with additional options such as scoring (rewards or penalties) and manipulating defensive or adversarial strategies. The first two stages, i.e., *Level 0* and *Level 1* must hold for one or multistage games. In Level 0, a warm-up phase, both teams gather the necessary resources. In Level 1, the attacker team chooses the potential attack surface(s), such as exploiting vulnerabilities in the unpatched software through reconnaissance, whereas the defender team chooses defensive measures, such as using anti-malware tools. For the sake of simplicity, we choose a one-stage game where the dataset with vulnerabilities is loaded to the virtual machine for the defender team to play their part. The events are logged in data storage, from where the data can be utilized as an input data source for the next game or further analysis. To interpret the AI-based decision for attack ($Attack$) and non-attack ($N\text{-}Attack$) classification, XAI-based DT assessment (*step 7*) is used. The main objective of this module is to enable the DevSecOps team to comprehend and retrace the results made by AI agents (defender team, in our case).

Moreover, integrating XAI mitigates the mistrust caused by the black-box nature of AI/ML algorithms and establishes more understanding and confidence in the decisions made during DT security assessment. The derived results, such as updated rules ($Rules$) or identified vulnerabilities, are first verified in the simulation mode. The reason for this additional step is to make sure that new configurations will not affect the physical environment. The results are fed into the DT environment (Replication mode) (*step 8*) and eventually mapped to the physical environment (*step 9*). Please note that due to limited resources, i.e., the absence of a physical environment and paid subscription to instantiate multiple DT instances in ADT, steps 8 and 9 are not simulated in this work. We anticipate that the existing state replication mechanisms, such as Eckhart and Ekelhart (2018a) can be used to realize the fully operational Digital-Physical mapping, and we plan to extend the current version in future work. The steps for generating a DT model, setting a gaming environment, and explaining game results are discussed in Algorithm 1, Algorithm 2, and Algorithm 3 respectively.

Algorithm 1: Algorithm 1 describes a twin instance (t_i) derived from a DT model (DT_Model). A DT_Model consists of context, Id, type, schema, and contents. To instantiate (t_i), we define the systems' specification (χ_{spec}), including twin current state ($State^{\omega}_{t_i}$), maximum and minimum thresholds ($\tau^{\mathbf{max}}_{t_i}, tau^{\mathbf{min}}_{t_i}$) for example, speedometer range, and a classifier ($classifier$) to classify $Attack$ or $N\text{-}Attack$ scenarios. The IoT devices are registered and authenticated as discussed in Section 5.2. To generate timestamped sensor data ($D_{t_i,time}$), a three-step function f is applied as follows: (i) $f^{ingest}_{s_1}$ obtains telemetry data from Azure IoT Hub, (ii) $f^{transform}_{s_2}$ transforms the data to the required format of the DT instance, and (iii) finally, $f^{feed}_{s_3}$ feeds data to t_i. The data we considered for the DT instance is real-time (online) generated through an IoT device simulator. Through ADT explorer, t_i is then made available for visualization.

Algorithm 1 A twin instance derived from a DT model

Input: $ID_{t_i}, D_{t_i,time}, DT\_Model, \chi_{spec}$
Output: $t_i \in T_o$
1: Define $DT\_Model = \langle Context, Id, Type, Schema, Contents \rangle$
2: Define $\chi_{spec} = \langle State^{\omega}_{t_i}, \tau^{\mathbf{max}}_{t_i}, \tau^{\mathbf{min}}_{t_i}, classifier(Attack, N\text{-}Attack) \rangle$
▷ System specification
3: Obtain registered IoT devices from Azure IoT Hub
4: Timestamped object (sensor) data $D_{t_i,time}$ from Azure IoT Hub by three-step function f
a: $f^{ingest}_{s_1} : D_{telemetry}$
b: $f^{transform}_{s_2} : D_{telemetry} \mapsto D_{t_i,time}$
c: $f^{feed}_{s_3} : t_i \leftarrow D_{t_i,time}$
5: $t_i \in T_o$ accessible at ADT explorer

Algorithm 2: Algorithm 2 presents Phase-I of DT-based gaming environment. We define the teams, i.e., human team as attackers (T_H) and AI agents as defenders (T_{AI}). Based on the designed set of challenges (C_i), the roles ($Roles$) are assigned to each team. Similarly, resources ($G_{resources}$) are also allocated to each team. The next steps include instantiating a game instance ($G_{instance}$) from t_i (as discussed in Algorithm 1). The $G_{instance}$ is based on three main components (as discussed in Eq. (2)), including game story (G_{story}), game data (G_{data}), and system configurations (χ_{config}). We based our G_{story} on a set of challenges (C_i). G_{data} can either be a publically available dataset or directly obtained from t_i. Finally, the $G_{instance}$ is loaded into a Virtual Machine (VM). Now, based on the G_{story}, the respective teams will participate in the game by performing actions such as choosing an attack surface and

Algorithm 2 DT-based gaming environment (Phase-I)

Input: T_H, T_{AI}, $Roles$, $G_{resources}$, t_i, $G_{instance}$, χ_{config}, VM
Output: $logs$, $G_{results}$
1: Define teams T_H (attacker) and T_{AI} (defender)
2: Assign $Roles$ to each team (attacker: T_H, defender: T_{AI}) subjected to C_i
3: Allocate $G_{resources}$ to teams (T_H, T_{AI})
4: Derive a required DT model $t_i \in T_o$ referred to as $G_{instance}$ ▷ Call Algorithm 1
5: Based on $G_{instance}$, describe G_{story} and a set of challenges C_i
6: Load $time_{assigned}$ for selected C_i
7: Prepare game data G_{data}
8: Load system configurations χ_{config} ▷ Record χ_{config} during active $G_{instance}$
9: $G_{instance} = \langle\ G_{story},\ G_{data},\ \chi_{config}\ \rangle$
10: Load game $VM \leftarrow G_{instance}$
11: **do**
12: T_H and T_{AI} equip with $G_{resources}$
13: Based on G_{story} :
a: T_H choose attack surface and launch attack
b: Following T_H action, T_{AI} choose defense measure and trigger defense strategy
c: T_H and T_{AI} (re)configure attack/defense ▷ Works under multi-staged game settings
14: $C_{i,result} = (f(Accomplished(G_{story}[C_i]))$
15: Record and store data in $logs$
16: Record $time_{spent}$ to accomplish C_i
17: **while** ($C_{i,result}$)
18: $timespan = time_{spent} - time_{assigned}$
19: **if** ($timespan > 0$) **then**
20: Record $timespan$
21: **end if**
22: Call Algorithm 3

Algorithm 3 XAI-based DT security assessment (Phase-II)

Input: $logs$, $G_{results}$
Output: $X_{decision}$
1: Obtain $G_{results}$ from Algorithm 2
2: Access T_H and T_{AI} actions and states data from $logs$
3: Derive $\mathbf{SHAP}_{values}$
4: Derive $X_{decision}(\mathbf{SHAP}_{values})$
a: $Attack \leq \mathbf{SHAP}_{values} \leq N\text{-}Attack$
5: Instantiate a DT instance from t_i
6: Test and validate $G_{results}$ in DT instance ▷ Peformed in isolated simulation mode
7: Update $Rules$
8: $t_i \mapsto p_i \in P_o$ ▷ Map states and data from digital to physical object

launching an attack are carried out by T_H whereas triggering defense measure by T_{AI} (as discussed in step 12 of the algorithm). This process will continue until all challenges are accomplished. For each C_i, we are recording time assigned ($time_{assigned}$) to accomplish a challenge (C_n) by the defender team. Moreover, we are also recording time spent ($time_{spent}$) by the defender team to accomplish the same challenge (C_n). Upon the completion of all challenges defined in C_i, we compute $timespan$. If more time is required, (i.e., $timespan > 0$), we log this activity. The purpose of keeping track of $time_{assigned}$ and $time_{spent}$ time intervals can help the security analysts in two ways: (a) to set the difficulty level of the challenges for future $G_{instance}$, and (b) to estimate how much time could be invested if the physical system or the DT is under similar attacks.

Algorithm 3: Algorithm 3 discusses Phase-II of the game, i.e., assessment of the gaming result. After retrieving the required information from logs ($logs$), the gaming results ($G_{results}$) obtained from Algorithm 3 are explained through $X_{decision}(\mathbf{SHAP}_{values})$. $\mathbf{SHAP}_{values}$ provides reasoned decisions of $Attack$ and $N\text{-}Attack$ scenarios. The rules ($Rules$) which may include safety policies, patch system, or other configuration settings. Finally, the validated and verified instances (from steps 4 and 5) are mapped to the physical system (p_i).

5. Implementation and evaluation

We present a prototypical implementation to demonstrate and validate ENIGMA. Our experimental setup includes a brief overview of the selected DT platform (Section 5.1), a detailed description of the vehicle DT (Section 5.2), and datasets (Section 5.3). Based on the available resources, we validate the proposed framework with reasonable assumptions (Section 5.4).

5.1. Choosing a DT platform

We start with choosing a publicly available DT platform that can facilitate the software implementation of a vehicle DT model (our use case) by representing the physical counterpart. There are multiple simulation platforms, including Microsoft Azure Digital Twins (ADT), Simio, Simul8, Simumatik, and Ditto, each having its pros and cons (Pavlov, 2022). Considering our use case requirements and assumptions (i.e., data models and sensor data with predefined thresholds), we choose the ADT platform where Azure IoT Hub is used to generate sensor data, a DT explorer to visualize a vehicle DT, and an Azure function as data ingestion.

The ADT is a platform-as-a-service based on IoT that renders a DT environment of a physical representation. The ADT collects data from IoT devices via Azure IoT Hub, which serves as a cloud gateway, and uses Azure function for data ingestion, transformation, and processing platform (Microsoft, 2022). Moreover, the ADT platform facilitates the users to design DT graphs and their vocabulary using the Digital Twin Definition Language (DTDL).

5.2. Vehicle DT creation and deployment

In our use case, the vehicle DT environment encapsulates the AI models and establishes a feedback loop with a physical system. Using DTDL, we create the vehicle DT model that observes static details (e.g., physical properties of the system) and dynamic information (e.g., sensor data). More specifically, the process knowledge of DTs is based on the system's specification data. In addition, the vehicle DT comprises sensor models (e.g., speedometer, indicator, coolant) connected with registered IoT devices on Azure IoT Hub and receives data from the vehicle sensors through the CAN bus (as Fig. 5 shows). The vehicle DT source code is available on GitHub.[1]

Creating a vehicle DT on ADT involves setting up several components and processes. A step-by-step approach is as follows. The process starts with configuring the vehicle's ECUs and communicating with the CAN bus. Three sensors, i.e., speedometer, indicator, and coolant are then installed on respective ECUs positions to emit data to the vehicle DT. The sensor devices are registered on the Azure IoT Hub and are authenticated using Shared Access Signatures (SAS) token authentication (Microsoft, 2023). The Azure IoT Hub receives data from the sensors through the CAN bus. To route the data to the vehicle DT, the event grid is used that allows subscription to an Azure function. The Azure function acts as an on-demand compute functionality, serving as a data ingestion, transformation, and feeder to the DT. The Azure function receives the data and transforms it into JSON format using JsonPatchDocument (Fig. A.7), which is readable by the vehicle DT.

[1] https://github.com/mubashar-iqbal/enigma-digital-twin

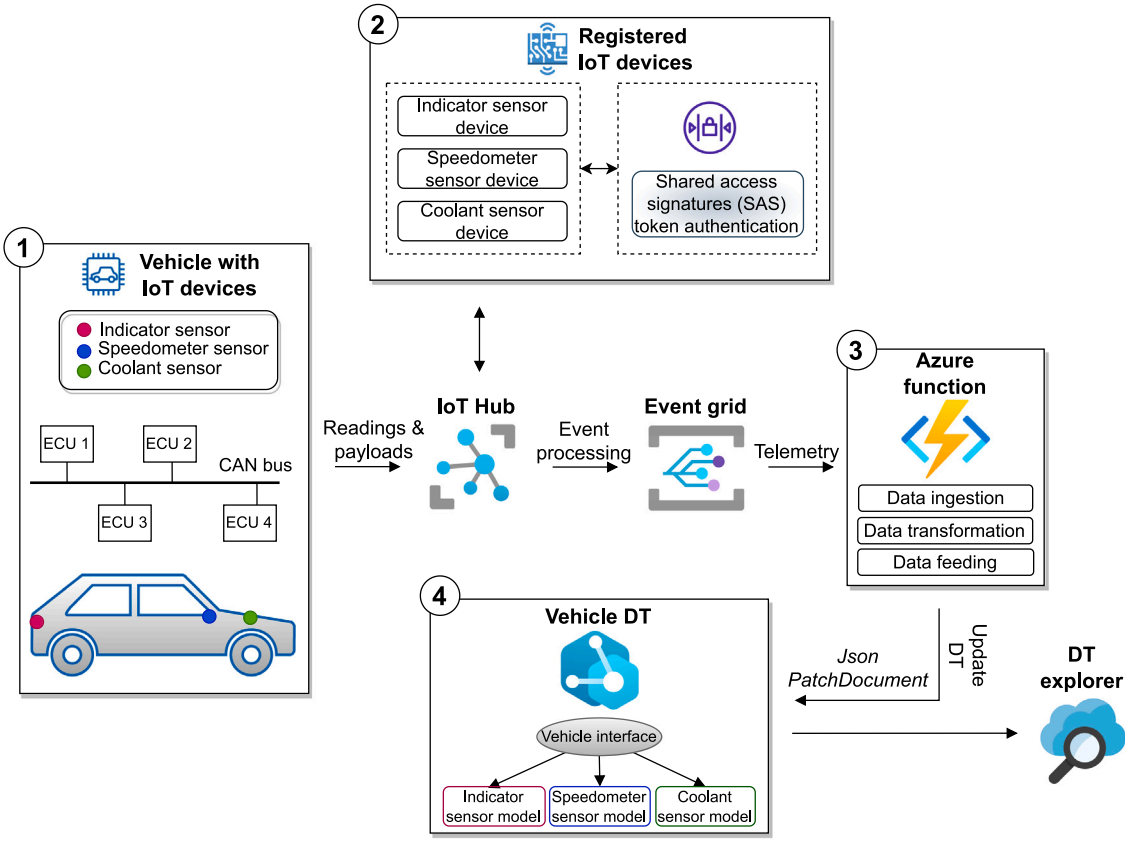


Fig. 5. An overview of ENIGMA architecture realized with Microsoft Azure Digital Twin.

The information gathered by the Azure function is then used to update the vehicle DT.

Finally, the ADT explorer allows users to explore the twins in a graphical view to visualize and interact with the DT and the data streams received from the sensors (Fig. C.9). The ADT explorer loads the vehicle DT using the ADT instance URL.[2] One of ADT Explorer's key advantages is the ability to create twin graph and vocabulary for twined IoT devices by using user-defined models in the DTDL. In our vehicle DT, we have a vehicle interface model (e.g., VehicleInterface.json) and its relationships (e.g., rel_has_speedometer_sensor, rel_has_indicator_sensor, rel_has_coolant_sensor) with three sensor models (e.g., SpeedometerSensor.json, IndicatorSensor.json, CoolantSensor.json). We make use of the DTDL validator to verify the correctness of the models and guarantee that the designated schema follows the DTDL guidelines and specifications.

The sensor models consist of several attributes crucial for vehicle DT functionality. A speedometer sensor model (SpeedometerSensor.Json) is shown as an example (Fig. B.8) to illustrate the elements, such as schema, properties, relationships, and content, which describe the type of DT in our DT environment. For instance, "@context" defines the namespace, and "@id" specifies a unique identifier for the DT model. The "@type" field describes the information contained within the model. The "displayName" field provides a user-friendly name for the model, making it easier for users to identify it. Finally, the "contents" field contains the DT readable structure for the sensor readings.

The "contents" field of the model is responsible for holding information about the various attributes related to the sensors. Within this field, several key elements are defined. The first of these elements is the sensor ID, which is used to identify the sensor as it is registered on Azure IoT Hub. The sensor state indicates the functional and non-functional state of the sensor to understand the current condition and operation of the device. Another important aspect of this field is the exposition of the CAN bus payload. This refers to the readings obtained from the speedometer sensor through the CAN bus. These readings are critical in understanding the operation of the speedometer. Moreover, the "contents" field includes the classification of the request, which indicates whether it is an Attack or N-Attack request. For example, in the gamification mode, we create an instance of the DT that continuously receives data through sensors relayed to AI models. The AI agent leverages DNN to generate a model trained and tested on the provided data following steps including data collection, data preprocessing, model building, learning, and evaluation (Fig. 1). In our case, we utilize the two publicly available datasets, Survival Analysis Dataset (SAD) (Han et al., 2018) and Oak Ridge National Laboratory (ORNL) (Verma et al., 2020), which provides the classification of Attack and N-Attack data. We utilize the XAI-assisted assessments for the vehicle DT security using the datasets to comprehend the decisions made to AI stakeholders in the event of a security attack.

5.3. Datasets

We consider standard in-vehicle network traffic public datasets, i.e., SAD (Han et al., 2018) and ORNL (Verma et al., 2020). Both datasets are based on CAN data and feature injection attack scenarios (intrusion, man-in-the-middle, fuzzing, fabrication, targeted fabrication, masquerading, flooding, and malfunction, etc.) that can potentially impede in-vehicle functionality or exacerbate the degree of damage. An in-depth explanation of the SAD and ORNL datasets is available in Han et al. (2018), Verma et al. (2020), respectively.

[2] https://EnigmaDigitalTwin.api.eus.digitaltwins.azure.net *(accessible through ADT)*.

Table 4
In-vehicle network traffic dataset: Summary of features.

Notation	Description
TS	Timestamp of the collected CAN bus data
CAN_PL_0 - CAN_PL_7	Data where CAN_PL_0 is the first byte, and CAN_PL_7 is the last byte of the payload from CAN bus data
ID_MF	ID of the message frames
TD_TS_MF	Time difference between timestamps of message frames that refers to the difference in time between the arrival of consecutive messages on a CAN bus
TD_ID_MF	Time difference between CAN ID for individual message frames that refers to the time elapsed between the transmission of two different messages on a CAN bus
DLC	Data length code specifies the number of bytes of a CAN bus message

5.4. Experimental validation

To comprehend gaming outcomes, the XAI-assisted DT security assessment module retrieves the gaming results for analysis (as step 7 of Fig. 4 shows). This module provides a post-hoc explainability of the decisions taken by the AI agent in the prediction of an Attack or N-Attack during the simulated game. To do so, SHAP graphs in Fig. 6 demonstrate how various feature values contribute to the prediction of Attack or N-Attack. Table 4 summarizes the feature values used in ORNL and SAD datasets. To make informed decisions for securing CPS-DT environment, the XAI stakeholder can leverage the interpretation provided by the DT assessment module to determine the contribution of each feature for classifying Attack or N-Attack (as Fig. 6(a) shows). Higher SHAP values indicate a stronger contribution to the model in decision-making. As shown in Fig. 6(a), the contribution of SHAP values for Attack (red) or N-Attack (blue) decisions have an equal distribution (i.e., it implies that each feature is contributing similarly and equally to both types of predictions). For example, if the features are not equally relevant or important to both types of predictions, it may reflect that the model is not effectively capturing the relationships between features and target variables. As an illustration, the TD_ID_MF and ID_MF with SHAP values of 0.225 and 0.169 respectively reflect the impact on the contribution to the model decision. Among the features with higher SHAP values, we may observe that the identity and the time difference between CAN messages play a pivotal role in the attacks. The TD_ID_MF feature, to be more precise, indicates a Sybil attack. On the other hand, identity masquerading and fuzzing attacks used for zero-day exploits are possible through the provided dataset belonging to the ID_MF feature. The SHAP values make it explainable to the stakeholders (security analysts) to take correct and timely preventive measures.

Given that, the feature TD_ID_MF is of higher significance; therefore, we utilize the Partial Dependency Plot (PDP). The PDP illustrates the impact of a single feature (E[f(x)|TD_ID_MF] in our case) on the model's prediction (E[f(x)]) that helps to gain further insights into the predicted outcome. For example, Fig. 6(b) shows the PDP, where the x-axis represents the values of a chosen feature (i.e., TD_ID_MF), and the y-axis displays the influence of that feature in predicting whether a CAN message would be classified as an Attack or N-Attack. The CAN message with TD_ID_MF values less than −0.8 is most likely to be classified as an Attack, while the CAN message with values greater than −0.8 is classified as N-Attack. At the same time, the gray bar indicates the distribution of the feature values in the dataset and helps in interpreting the relationship between the feature and the predicted outcome. For instance, the height of gray bars on the x-axis in PDP characterizes the magnitude of the contribution of the TD_ID_MF feature to the prediction made by the XAI model in a specific range of TD_ID_MF feature values.

6. Limitations and future directions

The potential future research directions of ENIGMA's design stance lie at the intersection of game-based learning and offensive security dimensions. In the following, we discuss the limitations of the prototypical implementation of ENIGMA (Section 6.1) and provide an outlook on the future research directions of ENIGMA (Section 6.2).

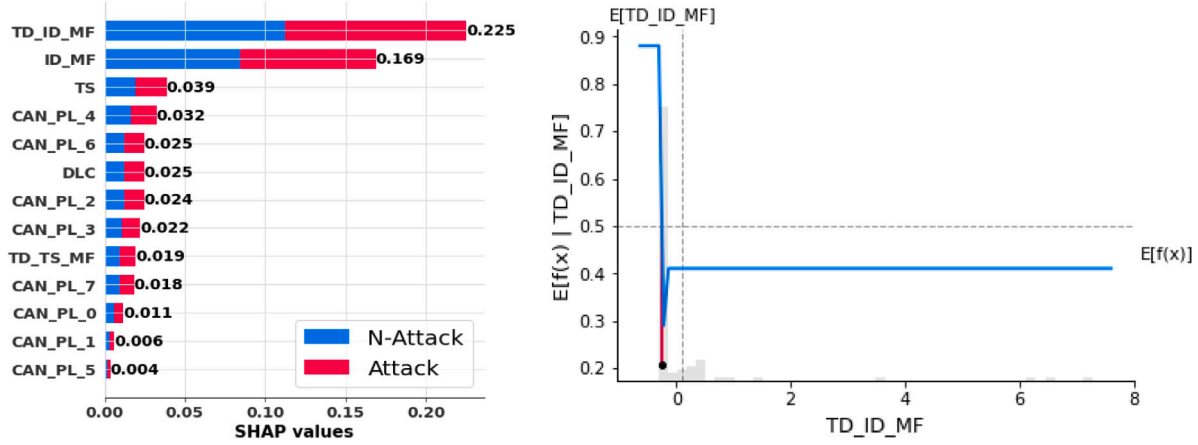
6.1. Threats to validity

The preliminary prototypical implementation of the ENIGMA aims to showcase a proof of concept for the gamification of DT. We identify the following threats to the validity of our experiments. First, developing a DT that closely replicates the functionality and behavior of the use case is a complex task. Therefore, based on the context, feature simplifications or generalizations can be adopted, as suggested in Minerva and Crespi (2021). Second, ENIGMA is tested and validated on two datasets, namely SAD and ORNL, featuring attacks on in-vehicle data as our underlying CPS use case. However, the presented validation must be generalized across datasets from different domains. Third, to demonstrate the generation and collection of sensor data (indicator, speedometer, coolant) of the prototype, we rely on Azure IoT Hub, which acts as a common source for device data to be used in ADT (Microsoft, 2022). However, in practical settings, such data collected from real operating devices can be ingested into ADT through the Azure function. Thus, without conducting experiments that require a complicated setup, including a real system generating real-time data, free access to the services of the simulator, and setting up the connection between the twin and its physical counterpart, we cannot draw a solid conclusion.

6.2. Potential extensions to ENIGMA

In the following, we discuss the future research directions pertaining to the ENIGMA framework.

- *Hybrid attack scenarios*: An interesting direction for future work is to explore the potential security threats incur due to hybrid attacks. More specifically, more research focus is needed on how the ENIGMA platform can effectively detect and localize the faulty entity in a complex environment that involve implicit interactions between CPS-DT. The offensive operations could be applied in both physical and twin environments to gamify security evaluation.
- *Engaging gameplay*: The challenges based on gamification and offensive security enhance engagement, motivation, and knowledge of the participating teams (Mink and Greifeneder, 2010). Therefore, for complex environments, to make the gaming environment more comprehensive, the current single-stage gaming environment can be expanded to a multi-stage gaming environment where attackers and defenders can compete in a continuous multi-round engagement. This setting could provide more insights into attackers' and defenders' strategies, and hence, (re)adjust the level of game complexity.



(a) Features importance for DNN.

(b) Partial dependence plot for DNN.

Fig. 6. Explanation of features used in ORNL and SAD datasets through SHAP.

- *Realism*: To support an interactive and immersive game-based learning experience in cybersecurity games, a potential research direction is to move ENIGMA from a DT-based environment to a metaverse-supported environment for experiential learning. Such transition can be realized due to the confluence of enabling technologies including Virtual Reality (VR), Augmented Reality (AR), 5G/6G networks, and wearables (Lim et al., 2022) with high-fidelity DTs. While the importance of cyber-threat intelligence has been highlighted for the metaverse (Dunnett et al., 2023), the use of metaverse-coupled DT's for security training remains an open issue. Given that, gaming is the mainstream component in a metaverse (Wang et al., 2023), ENIGMA in a metaverse-supported platform can provide security analysts with an increased level of realism through VR training and testing of the underlying use cases in addition to engaging and adaptive gameplay. However, ENIGMA, with the notion of a beyond-learning environment in the metaverse, requires further investigation from security and privacy aspects.
- *Role of ENIGMA in Industry 5.0 vision*: ENIGMA's platform can be further explored to realize the vision of Industry 5.0 (European Commission and Directorate-General for Research and Innovation et al., 2021) as follows: (i) *human-centric approach*: lifelong learning and training of workforce through an interactive gaming environment, (ii) *sustainable*: value creation by identifying anomalous behavior and timely troubleshooting, and (iii) *resilient*: monitoring and analyzing DT instances to predict attacks before it can cause long-term consequences.
- *Adaptive gamification*: ENIGMA can be further evolved to an intelligent adaptive gamification platform, where the game orchestration platform can be designed to deploy on-demand adaptive games for different scenarios pertaining to different security incidents. This will enable both training the personnel and taking necessary measures for attack mitigation and prevention in the underlying systems.
- *DT fidelity vs. security and efficiency*: The current ENIGMA framework considers the DT of a CPS in its entirety without taking into account the complex scenarios. However, in order to consider proactive security for CPS, variable fidelity is essential. The level of security provided by variable-fidelity DTs and their efficiency and sustainability is subject to further investigation. Leveraging variable fidelity DTs (preferably low-fidelity) can be served as a deception platform.

7. Conclusion

This paper proposed a gamification approach called ENIGMA for CPS-DT ecosystem security. By simulating attack–defense scenarios, without risking the critical infrastructure, ENIGMA reaps the following benefits: (i) train security analysts by defining context, environment, and learning objectives to gain practical knowledge and skills during an exercise or challenge, and (ii) evaluate the security of DTs and eventually the physical assets. Furthermore, ENIGMA is supported by an XAI-assisted security solution to provide an explanation of the decisions made by the DT security assessment model. The XAI-assisted DT security assessment model can help AI stakeholders to take security measures based on reasoned and trustworthy decisions. Finally, experimental validation has been carried out to demonstrate the viability of ENIGMA.

CRediT authorship contribution statement

Sabah Suhail: Conceptualization, Data curation, Investigation, Methodology, Validation, Visualization, Writing – original draft, Writing – review & editing. **Mubashar Iqbal:** Conceptualization, Data curation, Investigation, Methodology, Validation, Visualization, Writing – original draft, Writing – review & editing. **Rasheed Hussain:** Conceptualization, Data curation, Investigation, Methodology, Supervision, Validation, Writing – original draft, Writing – review & editing. **Raja Jurdak:** Supervision, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. JSON patch document

See Fig. A.7.

Appendix B. Speedometer sensor model

See Fig. B.8.

```
var updateTwinData = new JsonPatchDocument();

// sensor details
updateTwinData.AppendAdd("/Id", deviceId);
updateTwinData.AppendAdd("/SensorState", sensorState);

// can bus payload mapping to speedometer DT
updateTwinData.AppendAdd("/CanBusPayload/TS", ts);
updateTwinData.AppendAdd("/CanBusPayload/ID_MF", id_mf);
updateTwinData.AppendAdd("/CanBusPayload/CAN_PL_DATA_BYTES", can_pl_data_bytes);  // comma separated can bus payload bytes
updateTwinData.AppendAdd("/CanBusPayload/TD_TS_MF", td_ts_mf);
updateTwinData.AppendAdd("/CanBusPayload/TD_ID_MF", td_id_mf);
updateTwinData.AppendAdd("/CanBusPayload/DLC", dlc);

// classifier: Attack or N-Attack
updateTwinData.AppendAdd("/Classifier", classifier);

// update speedometer DT
await client.UpdateDigitalTwinAsync(deviceId, updateTwinData).ConfigureAwait(false);
```

Fig. A.7. JSON Patch Document for DT readable format.

```
{
    "@context": "dtmi:dtdl:context;2",
    "@id": "dtmi:vehicle:speedometer_sensor;1",
    "@type": "Interface",
    "displayName": "Speedometer Sensor Interface Model",
    "contents": [
        {
            "@type": "Property",
            "name": "Id",
            "schema": "string",
            "description": "Speedometer Sensor Id",
            "writable": true
        },
        {
            "@type": "Property",
            "name": "SensorState",
            "schema": "string",
            "description": "Funcational and Non-Functional State of Speedometer Sensor",
            "writable": true
        },
        {
            "@type": "Property",
            "name": "CanBusPayload",
            "description": "Speedometer Sensor Can Bus Payload",
            "schema": {
                "@id": "dtmi:vehicle:speedometer_sensor:payload;1",
                "@type": "Object",
                "fields": [
                    {
                        "@id": "dtmi:vehicle:speedometer_sensor:payload:timestamp;1",
                        "name": "TS",
                        "schema": "integer"
                    },
                    {
                        "@id": "dtmi:vehicle:speedometer_sensor:payload:id;1",
                        "name": "ID_MF",
                        "schema": "integer"
                    },
                    {
                        "@id": "dtmi:vehicle:speedometer_sensor:payload:bytes;1",
                        "name": "CAN_PL_DATA_BYTES",
                        "description": "CAN_PL_0 is the first byte, and CAN_PL_7 is the last byte of the payload from CAN bus",
                        "schema": "string"
                    },
                    {
                        "@id": "dtmi:vehicle:speedometer_sensor:payload:timestamp_difference_mf;1",
                        "name": "TD_TS_MF",
                        "schema": "integer"
                    },
                    {
                        "@id": "dtmi:vehicle:speedometer_sensor:payload:timestamp_difference_id;1",
                        "name": "TD_ID_MF",
                        "schema": "integer"
                    },
                    {
                        "@id": "dtmi:vehicle:speedometer_sensor:payload:dlc;1",
                        "name": "DLC",
                        "schema": "integer"
                    }
                ]
            }
        },
        {
            "@type": "Property",
            "name": "Classification",
            "schema": "boolean",
            "description": "The classification tells about the request is Attack or Non-Attack",
            "writable": true
        }
    ]
}
```

Fig. B.8. Speedometer sensor model defined in JSON-LD using DTDL specifications.

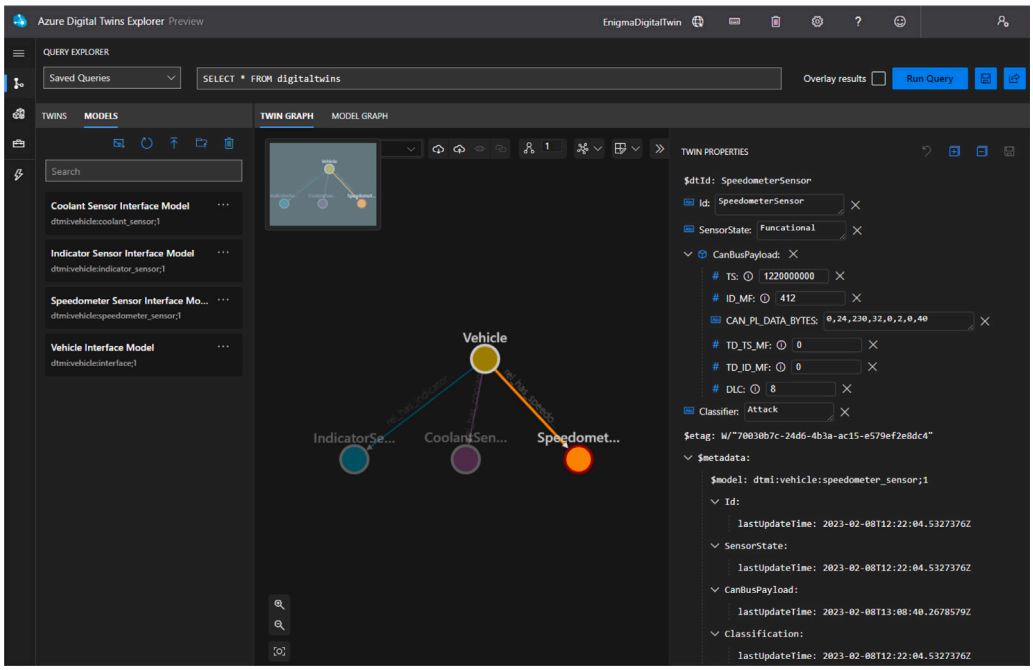


Fig. C.9. ADT explorer.

Appendix C. ADT explorer screenshot

See Fig. C.9.

References

Alcaraz, C., Lopez, J., 2022. Digital twin: A comprehensive survey of security threats. IEEE Commun. Surv. Tutor. 24 (3), 1475–1503. http://dx.doi.org/10.1109/COMST.2022.3171465.

Antonioli, D., Ghaeini, H.R., Adepu, S., Ochoa, M., Tippenhauer, N.O., 2017. Gamifying ICS security training and research: Design, implementation, and results of S3. In: Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy. CPS '17, Association for Computing Machinery, New York, NY, USA, pp. 93–102. http://dx.doi.org/10.1145/3140241.3140253.

Antonioli, D., Tippenhauer, N.O., 2015. MiniCPS: A toolkit for security research on CPS networks. In: Proceedings of the First ACM Workshop on Cyber-Physical Systems-Security and/Or PrivaCy. In: CPS-SPC '15, Association for Computing Machinery, New York, NY, USA, pp. 91–100. http://dx.doi.org/10.1145/2808705.2808715.

Barricelli, B.R., Casiraghi, E., Fogli, D., 2019. A survey on digital twin: Definitions, characteristics, applications, and design implications. IEEE Access 7, 167653–167671. http://dx.doi.org/10.1109/ACCESS.2019.2953499.

Bécue, A., Fourastier, Y., Praça, I., Savarit, A., Baron, C., Gradussofs, B., Pouille, E., Thomas, C., 2018. CyberFactory#1 — Securing the industry 4.0 with cyber-ranges and digital twins. In: 2018 14th IEEE International Workshop on Factory Communication Systems. pp. 1–4. http://dx.doi.org/10.1109/WFCS.2018.8402377.

Corallo, A., Lazoi, M., Lezzi, M., Luperto, A., 2022. Cybersecurity awareness in the context of the Industrial Internet of Things: A systematic literature review. Comput. Ind. 137 (C), http://dx.doi.org/10.1016/j.compind.2022.103614.

Dedeoglu, V., Jurdak, R., Dorri, A., Lunardi, R.C., Michelin, R.A., Zorzo, A.F., Kanhere, S.S., 2020. Blockchain technologies for IoT. In: Kim, S., Deka, G.C. (Eds.), Advanced Applications of Blockchain Technology. Springer Singapore, Singapore, pp. 55–89. http://dx.doi.org/10.1007/978-981-13-8775-3_3.

Dietz, M., Pernul, G., 2020. Unleashing the digital twin's potential for ICS security. IEEE Secur. Priv. 18 (4), 20–27. http://dx.doi.org/10.1109/MSEC.2019.2961650.

Dunnett, K., Pal, S., Jadidi, Z., Jurdak, R., 2023. The role of cyber threat intelligence sharing in the metaverse. IEEE Internet Things Mag. 6 (1), 154–160. http://dx.doi.org/10.1109/IOTM.002.2200003.

Eckhart, M., Ekelhart, A., 2018a. A specification-based state replication approach for digital twins. In: Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy. Association for Computing Machinery, New York, NY, USA, pp. 36–47. http://dx.doi.org/10.1145/3264888.3264892.

Eckhart, M., Ekelhart, A., 2018b. Towards security-aware virtual environments for digital twins. In: Proceedings of the 4th ACM Workshop on Cyber-Physical System Security. Association for Computing Machinery, New York, NY, USA, pp. 61–72. http://dx.doi.org/10.1145/3198458.3198464.

Eckhart, M., Ekelhart, A., 2019. Digital twins for cyber-physical systems security: State of the art and outlook. In: Biffl, S., Eckhart, M., Lüder, A., Weippl, E. (Eds.), Security and Quality in Cyber-Physical Systems Engineering: With Forewords by Robert M. Lee and Tom Gilb. Springer International Publishing, Cham, pp. 383–412. http://dx.doi.org/10.1007/978-3-030-25312-7_14.

Eckhart, M., Ekelhart, A., Weippl, E., 2019. Enhancing cyber situational awareness for cyber-physical systems through digital twins. In: 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation. pp. 1222–1225. http://dx.doi.org/10.1109/ETFA.2019.8869197.

European Commission and Directorate-General for Research and Innovation, Breque, M., De Nul, L., Petridis, A., 2021. Industry 5.0 : towards a sustainable, human-centric and resilient european industry. Publications Office of the European Union, http://dx.doi.org/10.2777/308407.

Han, M.L., Kwak, B.I., Kim, H.K., 2018. Anomaly intrusion detection method for vehicular networks based on survival analysis. Veh. Commun. 14, 52–63. http://dx.doi.org/10.1016/j.vehcom.2018.09.004.

Kayan, H., Nunes, M., Rana, O., Burnap, P., Perera, C., 2022. Cybersecurity of industrial cyber-physical systems: A review. ACM Comput. Surv. 54 (11s), http://dx.doi.org/10.1145/3510410.

Langner, R., 2013. To Kill a Centrifuge: A Technical Analysis of What Stuxnet's Creators Tried to Achieve. Available at: https://www.langner.com/wp-content/uploads/2017/03/to-kill-a-centrifuge.pdf. *(Accessed 11 February 2021)*.

Lee, R.M., Assante, M., Conway, T., 2017. Crashoverride: Analysis of the Threat to Electric Grid Operations. Dragos Inc., [Online]. Available: https://www.dragos.com/wp-content/uploads/CrashOverride-01.pdf.

Lim, W.Y.B., Xiong, Z., Niyato, D., Cao, X., Miao, C., Sun, S., Yang, Q., 2022. Realizing the metaverse with edge intelligence: A match made in heaven. IEEE Wirel. Commun. 1–9. http://dx.doi.org/10.1109/MWC.018.2100716.

Maddikunta, P.K.R., Pham, Q.-V., B, P., Deepa, N., Dev, K., Gadekallu, T.R., Ruby, R., Liyanage, M., 2022. Industry 5.0: A survey on enabling technologies and potential applications. J. Ind. Inf. Integr. 26, 100257. http://dx.doi.org/10.1016/j.jii.2021.100257.

Microsoft, 2022. What is azure digital twins? Available at: https://docs.microsoft.com/en-us/azure/digital-twins/overview. *(Accessed 21 June 2022)*.

Microsoft, 2023. Control access to IoT hub using shared access signatures. Available at: https://learn.microsoft.com/en-us/azure/iot-hub/iot-hub-dev-guide-sas. *(Accessed 02 January 2023)*.

Miller, S., Brubaker, N., Zafra, D.K., Caban, D., 2019. Triton actor TTP profile, custom attack tools, detections, and att&ck mapping. Fireeye Threat Res. Blog Available at: https://www.fireeye.com/blog/threat-research/2019/04/triton-actor-ttp-profile-custom-attack-tools-detections.html. *(Accessed 20 February 2023)*.

Minerva, R., Crespi, N., 2021. Digital twins: Properties, software frameworks, and application scenarios. IT Prof. 23 (1), 51–55. http://dx.doi.org/10.1109/MITP.2020.2982896.

Mink, M., Greifeneder, R., 2010. Evaluation of the offensive approach in information security education. In: Rannenberg, K., Varadharajan, V., Weber, C. (Eds.), Security and Privacy – Silver Linings in the Cloud. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 203–214.

Ometov, A., Molua, O.L., Komarov, M., Nurmi, J., 2022. A survey of security in cloud, edge, and fog computing. Sensors 22 (3), http://dx.doi.org/10.3390/s22030927.

Pavlov, V., 2022. Security Aspects of Digital Twins in Iot Platform. University of Twente, Available at: http://essay.utwente.nl/92695/. *(Accessed 11 February 2022)*.

Suhail, S., Hussain, R., Jurdak, R., Oracevic, A., Salah, K., Hong, C.S., Matulevičius, R., 2022a. Blockchain-based digital twins: Research trends, issues, and future challenges. ACM Comput. Surv. 54 (11s), http://dx.doi.org/10.1145/3517189.

Suhail, S., Malik, S.U.R., Jurdak, R., Hussain, R., Matulevičius, R., Svetinovic, D., 2022b. Towards situational aware cyber-physical systems: A security-enhancing use case of blockchain-based digital twins. Comput. Ind. 141, 103699. http://dx.doi.org/10.1016/j.compind.2022.103699.

Tao, F., Zhang, M., 2017. Digital twin shop-floor: a new shop-floor paradigm towards smart manufacturing. IEEE Access 5, 20418–20427. http://dx.doi.org/10.1109/ACCESS.2017.2756069.

Verma, M.E., Iannacone, M.D., Bridges, R.A., Hollifield, S.C., Kay, B., Combs, F.L., 2020. Road: The real ornl automotive dynamometer controller area network intrusion detection dataset (with a comprehensive can ids dataset survey & guide). arXiv:2012.14600. [Online].

Vielberth, M., Glas, M., Dietz, M., Karagiannis, S., Magkos, E., Pernul, G., 2021. A digital twin-based cyber range for SOC analysts. In: Data and Applications Security and Privacy XXXV: 35th Annual IFIP WG 11.3 Conference. Springer-Verlag, pp. 293–311. http://dx.doi.org/10.1007/978-3-030-81242-3_17.

Wang, Y., Su, Z., Zhang, N., Xing, R., Liu, D., Luan, T.H., Shen, X., 2023. A survey on metaverse: Fundamentals, security, and privacy. IEEE Commun. Surv. Tutor. 25 (1), 319–352. http://dx.doi.org/10.1109/COMST.2022.3202047.

Wood, L., Reiners, T., 2015. Gamification. pp. 3039–3047. http://dx.doi.org/10.4018/978-1-4666-5888-2.ch297.

Zhou, H., Li, M., Sun, Y., Tian, Z., Yun, L., 2022. Digital twin based cyber range for industrial internet of things. IEEE Consum. Electron. Mag. 1–11. http://dx.doi.org/10.1109/MCE.2022.3203202.