Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 2323228, 16 pages
https://doi.org/10.1155/2022/2323228

Research Article

Design of Threat Response Modeling Language for Attacker Profile Based on Probability Distribution

Shuqin Zhang,[1] **Shijie Wang**,[1] **Guangyao Bai**,[1] **Minzhi Zhang**,[1] **Peng Chen**,[1] **Chunxia Zhao**,[1] **Shuhan Li**,[1] **and Jiehan Zhou**[2]

[1]*School of Computer Science, Zhongyuan University of Technology, Zhengzhou 450007, China*
[2]*University of Oulu, Oulu, Finland*

Correspondence should be addressed to Shuqin Zhang; zhangsq@zut.edu.cn

Received 6 April 2022; Accepted 1 June 2022; Published 16 June 2022

Academic Editor: Zhiguo Qu

Threat modeling and simulation (TMS) was aimed at dynamically capturing the features of attacks, which is a challenging job in complex Industrial Internet of Things (IIoT) control systems due to the complicated relationships among attacks. Recently, Meta Attack Language (MAL) showed its powerful TMS capabilities for representing complex attacks. However, existing methods pay less attention to the impact of changes in threat profiles on the simulation of key attack techniques. This paper proposes a novel method called threat response modeling language (TRMLang) for threat modeling and simulation in complex IIoT attacks. TRMLang obtains attacker information through an automated analysis of cyber threat intelligence (CTI) to build dynamic attacker profiles. Furthermore, it merges attacker features and probabilistic attack graphs in the simulation to improve TMS performance. The experimental results demonstrate that TRMLang can represent and evaluate the security conditions of IIoT control systems with two attack cases by Lazarus Group on SEGRID smart grids.

1. Introduction

In recent years, the Industrial Internet of Things (IIoT) has grown rapidly and offers tremendous advantages in remote monitoring, information gathering, and workforce reduction [1]. Therefore, smart factories [2], smart grids [3], smart medicine [4], and other industrial scenarios have widely deployed IIoT systems. However, in complex IIoT networks, many serious attacks are difficult to identify dynamically due to decentralized infrastructures or the lack of well-developed security assessment mechanisms. Any part of anomalies leads to delay of the IIoT control network and system [5].

The main purpose of threat modeling for complex IIoT networks is to dynamically capture the features between attack processes, which are highly correlated with key attack techniques. These features can provide rich information for threat modeling and simulation (TMS). Threat modeling is based mainly on three types of models: first-principle models, data-driven models, and domain-specific language models [6]. First-principle models combine prior knowledge of industrial safety with a mathematical model. However, it is too difficult to obtain accurate mathematical model for complex industrial process because of strong uncertainty (including unknown of the attack process and unmappable of certain attack techniques). Compared to first-principle models, the main advantage of data-driven models is that less mechanical knowledge is involved. But the data-driven model solely offers the capability to model security-relevant properties, and analysis needs to be conducted manually [7].

The domain-specific language (DSL) model inherits the advantages of the data-driven model and greatly facilitates system security design and automated analysis, as it effectively separates security engineering from system engineering. This separation is convenient not only from the point of reuse but also because it allows the separation of services [8]. Thus, security experts can create modeling languages, perhaps by introducing modeling elements such as assets, firewalls, vulnerabilities, defenses, and attacker actions. Moreover, the

reusability of data-driven security designs is poor, and it is difficult to adapt to development in multiple scenarios, so we need to introduce a more appropriate approach to solve this problem.

With the extraordinary advances in security engineering technology, a large number of process data and asset status data can be obtained from attack process. Thus, domain-specific language modeling has become the most popular threat modeling method. The commonly used domain-specific language modeling methods are unified modelling language (UML) [9], object constraint language (OCL) [10], extensible markup language (XML) [11], etc. Currently, researchers have developed scenario-oriented attack languages [12–18], relating them to existing security concepts, attack tactics, techniques, and practices. Arshad et al. [12] constructed an attack-specific language to concisely provide information about attack techniques, which will streamline and automate the cyber range functions of threat and challenge execution. Briland and Bouquet [13] proposed a method that uses a domain-specific language to generate modified data, allowing simulated and tested attacks such as injecting forged data into the system. Kern et al. [14] propose a domain-specific language for industrial automation and control systems that complies with international security standards. However, since many attack processes exhibit strong dynamics, these hard-coded modeling approaches are not applicable, because they are inflexible and hard to change and reuse. To track the dynamic features of complex attack processes, some formal methods have been proposed, such as the Meta Attack Language (MAL) [15], which is used as a domain-specific language combined with object-oriented modeling, and it generates probabilistic attack graphs to simulate the intrusion process of the attacker, for example, EnterpriseLang [16] for the modeling of network attack of enterprise IT systems, PowerLang [17] for the modeling of threat of power-related IT and OT infrastructures, and CoreLang [18] for threat modeling of general IT infrastructures. The results show that MAL can analyze more dynamic features from probabilistic attack graphs. Most of these models can encode the attack and defense logic of key techniques in real attack processes, but they cannot obtain dynamic features which are more relevant to the corresponding attackers. Therefore, combining attacker features to build MAL-based models has become a new challenge for IIoT domain threat modeling.

In the area of cyber threat intelligence (CTI), with the introduction of threat intelligence sharing technologies, CTI has recently become an important source of information in IIoT security. Atluri and Horne [19] designed a machine learning-based CTI framework for industrial control systems. It supports collecting threat intelligence passively from network traffic from critical control systems and extracts indicators of compromise (IoC) from anomaly type features. Yinghai et al. [20] proposed an overall framework for defense of industrial control network security, which integrated fragmented multisource threat intelligence with an industrial network layout using a security knowledge graph. The CTI dataset was used to construct a cybersecurity knowledge graph (CSKG) based on the basis of analyzing specific industrial control scenarios for further security analysis of the industrial control system. Moustafa et al. [21] proposed a framework to protect the physical layer of intelligent energy. The framework is based on CTI, machine learning, and physical layer security technology to enhance the security of intelligent energy systems in different applications. Cabana et al. [22] used network traffic analysis tools to analyze dark network traffic and generate threat intelligence on scanning campaigns targeting ICSs in the form of campaign fragments, and they investigated the payloads of the identified campaigns using a custom deep packet inspection technique to dissect and analyze the threat intelligence. CTI and its automated analysis have shown great power in dealing with dynamic attacker signatures.

However, most of the current threat intelligence researches are focused on anomaly extraction, information collection, and intelligence generation. Research on embedding dynamic features of complex data into system modeling is still in the exploratory stage [23]. Therefore, it is difficult to perform domain modeling, threat analysis, and follow-up work. The MAL framework-based model proposed in this paper combines attacker features with CTI. This design not only provides a higher-level attack concept but also effectively captures the features of intelligence data and generates high-quality intelligence data [24].

The term TTP (tactics, techniques, and procedures) originated in military and antiterrorism operations [25] and has since come to refer to the conventional attack tactics, techniques, and processes employed by attackers during their attack campaigns. In the complex IIoT attack process, relying on a single organization cannot build a complete TTP feature of the attacker. Therefore, compared to traditional security engineering technology for threats analysis, CTI automated analysis is a "space-for-time" technology, which can use threat intelligence in other networks to capture attacker TTP features and reduce the time required for analysis. Thus, this paper introduces attacker information obtained from the CTI automated analysis to build dynamic attacker profiles. Then, we built on the MAL framework to propose a new formal method called threat response modeling language (TRMLang) for threat modeling and simulation in IIoT complex attack processes. TRMLang encodes the attack defense logic in the IIoT domain, enabling semiautomatic generation and computation of large-scale probabilistic attack graphs. After that, the attacker features and probabilistic attack graphs in attack processes are further merged to improve TMS performance. The main contributions of this paper are summarized as follows.

(i) Propose a dynamic attacker profile for the mapping of threat features in complex attack processes. The attacker profile introduces dynamic information of adversary to capture the impact of changes in threat features over scenarios on the being simulated key techniques

(ii) Developed a threat response modeling language for IIoT threat modeling and simulation. Furthermore, it merges dynamic attacker profiles and probabilistic attack graphs during attacks to improve TMS performance. The dynamic features of the attacker are fed into TRMLang through an assigning probability distribution method to construct a dynamic threat mapping between the attacker action and key techniques

(iii) The TRMLang model with the attacker profile is compared with a model based on a standard penetration test to demonstrate the effectiveness of the method in attack simulation

The rest of the paper is organized as follows. Section 2 describes threat response modeling language in detail. Section 3 designs a method for assigning probability distributions. Then, the effectiveness of the proposed method is verified through two case studies in Section 4. Finally, Section 5 concludes this paper.

2. TRMLang Modeling Based on IIoT

This section presents the definition of TRMLang and the structure of the metamodel. Then, it describes the procedure for TRMLang-based attack simulation.

2.1. Definition. The state change of IIoT assets caused by a cyberattack initiated by an attacker essentially changed their security settings. TRMLang treats assets as nodes and attack dependencies between nodes as edges. This means that it can be applied to model an action that interacts with the system or a class in the MAL framework. The IIoT control system threat modeling has some predefined concepts that provide an abstract representation of its TMS capabilities, such as the following.

Definition 1. TRMLang denotes the IIoT assets set as a five-tuple: $G = (S, T, D, E, R)$.

$S = \{s_0, s_1, \cdots, s_i, \cdots, s_n\}$ denotes the set of all asset nodes, where s_0 is the first asset node that the attacker has successfully compromised and s_n is the attacker's target asset node.

$T = \{t_i | t_i = \text{action}(s_i)\}$ is the set of attacker attack actions, action denotes the action taken to attack t_i, which may be the technique or vulnerability exploit used by the attacker; s_i indicates the attack target asset of t_i; the value of t_i is 1 or 0, indicating whether the attack occurred.

$D = \{d_i | d_i = \text{state}(s_i)\}$ is the set of defense states, and state represents the state of the defense action; the value of d_i is 1 or 0, indicating whether the defense is enabled.

$E = \{e_{i \longrightarrow j} | i = 1, 2, \cdots, N, j = 1, 2, \cdots, N\}$ is the set of directed edges between the state asset nodes, indicating the change caused by an attacker taking a single attack action, such as the elevation of privilege and the increase in vulnerability. This makes the previous node point to the next node. $E_d = S \times T$ means executing an attack when defense is not enabled; $E_t = T \times S$ means arriving at the next node after completing an attack.

$R = R_d \cup R_t$ denotes the weight function that depends on the distribution of attack actions over time, where R_d depends on E_d, which denotes the probability of defense enablement, called defense enablement probability, and R_t depends on E_t, which refers to the probability of an attacker reaching the next asset after a successful action, called attack success probability.

Definition 2. Attack lateral movement.

Probabilistic attack graphs are graphical models that represent the knowledge about assets in the IIoT network and their interactions, showing that the different paths an attacker can follow to reach a given goal. An attack path in a probabilistic attack graph is a finite sequence of states $\{s_1, s_2, \cdots, s_n, s_i \in S, i = 0, 1, \cdots, n\}$, $(s_i, s_{i+1}) \in E$, which is the process of state change caused by an attacker's successful compromise of an attack target. Thus, the probabilistic attack graph is made up of various insecure state nodes and a set of attack actions that lead to state transfer. Along each path, each successful lateral movement gives the attacker more privileges towards its goal. In this sense, probabilistic attack graphs provide an appropriate framework for model TRMLang, since they depict causal relationships between random variables in a compact way.

After each attack action is successfully compromised, the attacker can execute the next attack action, indicated by "$\longrightarrow$." The lateral movement of the attack action from node s_i to node s_j and the attacker initiates the attack $e_{i \longrightarrow j}$ on node s_j. The calculated success probability of the attack is as in

$$P\left(e_{i \longrightarrow j}\right) = P_0\left(t_j\right) \times D_p\left(e_{i \longrightarrow j}\right), \tag{1}$$

where $P_0(t_j)$ is the conditional probability of attack action t_j and $D_p(e_{i \longrightarrow j})$ is the probability distribution of defense enablement of node s_j.

Definition 3. Attack dependency.

Generally, probabilistic attack graphs calculate asset risks using prior probabilities. Thus, in order to calculate the asset's prior probability, we must first determine the asset's local conditional probability. For any $t_i, t_j \in T$ and $e_{i \longrightarrow j} \in E$, t_i is called the parent action of t_j, and t_j is called a child action of t_i. Similarly, we have the set of parent actions $Pa(t_j) \coloneqq \{t_i \in T | e_{i \longrightarrow j} \in E\}$. The local conditional probability of t_j with its parent attack action $Pa(t_j)$ has two dependencies: $\{\text{AND}, \text{OR}\}$ (denoted by "&" and "|"). A logical AND where *all* parent attacks should be successful to calculate the success probability of the attack action t_j can be expressed as

$$P\left(t_j | Pa\left(t_j\right)\right) = \begin{cases} 0, \exists t_i \in Pa\left(t_j\right) | t_i = 0, \\ \prod_{t_i=1} P\left(e_{i \leftrightarrow j}\right), \text{otherwise.} \end{cases} \tag{2}$$

A logical OR where *at least one* of the preconditions in $Pa(t_j)$ needs to be satisfied to calculate the success probability of the attack action t_j can be expressed as

$$P(t_j|Pa(t_j)) = \begin{cases} 0, \forall t_i \in Pa(t_j) | t_i = 0, \\ 1 - \prod_{t_i=1} [1 - P(e_{i \longrightarrow j})], \text{otherwise.} \end{cases} \quad (3)$$

Definition 4. Asset attack costs.

In real-world attack scenarios, most targeted assets require a large time cost for the attacker. That is, the cost of the compromise target can be expressed as the local time-to-compromise (TTC). The TTC value indicates the time to compromise an asset, and it can measure the security level of various assets in the IIoT network. In each attack action, the probability distribution of the associated local TTC is repeatedly randomly sampled. Then, the action speed is predicted to indicate the cost of compromising the target. From $\forall t_j \in T$, the local conditional probability formula further calculates the TTC prior probability distribution of t_j, as in

$$P_{ttc}(t_1, t_2, \cdots, t_j) = \prod_{i=1}^{j} P(t_j|Pa(t_j)). \quad (4)$$

2.2. Metamodel of Threat Response Based on the IIoT. This section presents TRMLang, which enables the modeling of IIoT networks from security requirement engineering point of view. The Tropos framework is a method for analyzing and modeling the security requirement analysis stage, and it plays an important guiding role in the practice of security requirement engineering [26]. In this paper, the MAL modeling method is combined with the Tropos framework for the first time to define the TRMLang metamodel, which inherits the concept of Tropos. The metamodel is effective in providing abstraction definition capabilities for modeling the synergistic relationships between physical and virtual components in controlled scenarios. This allows simulating the hardware, software, and modeling resources required for IIoT and construction of high-fidelity models that can be used to calculate the system's time to compromise under threat scenarios.

In order for simulation experiments to reliably capture the features of IIoT control environments, testing and experimental case studies need to be described and modeled, considering both the cyber and physical domains. In addition, the simulation setup must capture the threat modeling features of the attacker and attack logic. In terms of a potential attacker, the threat features are adversarial knowledge, resources, access to the system, and specificity. As for the attack logic, the features include attack frequency, target level, attacked assets, attack techniques, and premises. Therefore, for each modeling metric, we provide the appropriate background, relevant definitions, attack setup, and mathematical formulation. The current version of the TRMLang metamodel is shown in Figure 1, which extends and builds upon the Tropos framework.

Cyber asset is a digital entity that is part of the infrastructure. *Physical asset* represents the hardware aspects of the system. *Asset* classes can perform different tasks, forming various associations between concepts of the same class and other classes. Therefore, there are self-associations between assets.

Attackers represent malicious actors who threaten the security of the system by compromising assets. In TRMLang, the *attacker* entity defines the starting point of an attack. It can be connected to any attack action entity t_i. This connection represents the source of the attack path. These particular attack actions thus always have a TTC that evaluates to 0.

Mission represents a subtarget that cannot fail during the attack, assuming that the malicious actor's *attack* has been successful. Based on this concept, asset reallocation can be modeled together with other attack decisions. Thus, the *mission* is represented by an inheritance relationship.

Vulnerabilities represent flaws in the implementation or design of IIoT control systems: they constitute vulnerabilities in the rule set represented by other *assets*, associations, and relationships. In TRMLang, the attack action of vulnerabilities is modeled rather than the consequences of the vulnerability exploit. The fact that not all vulnerability exploits result in successful compromises is captured with the probabilities in the attack action relations. Moreover, the existence of a vulnerability may be uncertain. This uncertainty is represented as a probability distribution, which further influences the calculation of the TTC.

Incident stands for intentional unauthorized access to a system, service, or resource of an IIoT or the compromise of a system's security properties. This concept differentiates an incident and a threat clarifying that an incident is successful and has malicious intent. Aggregate one or more *attacks* as attempts to exploit a *vulnerability* and together they can constitute an *incident*. In some cases, they multiply and spread. In order for the TRMLang to able to show that an *incident* can associate, encapsulate, support, or generate another incident, the reflexive association is used. For instance, malware replicates itself in crucial locations on a system. In this case, the *incident* reflexive association will be used to connect the malware with its copies.

In TRMLang, the metamodel shows how attacks are modeled to interact with assets. Specifically, the metamodel shows how actions are related between asset objects. In any attack scenario, the sequence of attacks from one object to another is critical to modeling. This is also applicable to defense and tactical strategies.

2.3. Attack Simulation Description. The proposed TRMLang is based on MAL. The MAL is a modeling and simulation language framework that combines probabilistic attack and defense graphs with object-oriented modeling, which in turn can be used to create domain-specific or scenario-specific languages and automate the security analysis of instance models within each domain. In this case, the MAL compiler uses a different backend to compile the MAL type of language into the corresponding files. This can then be used to create models graphically using their own proposed language and to simulate attacks on these models through

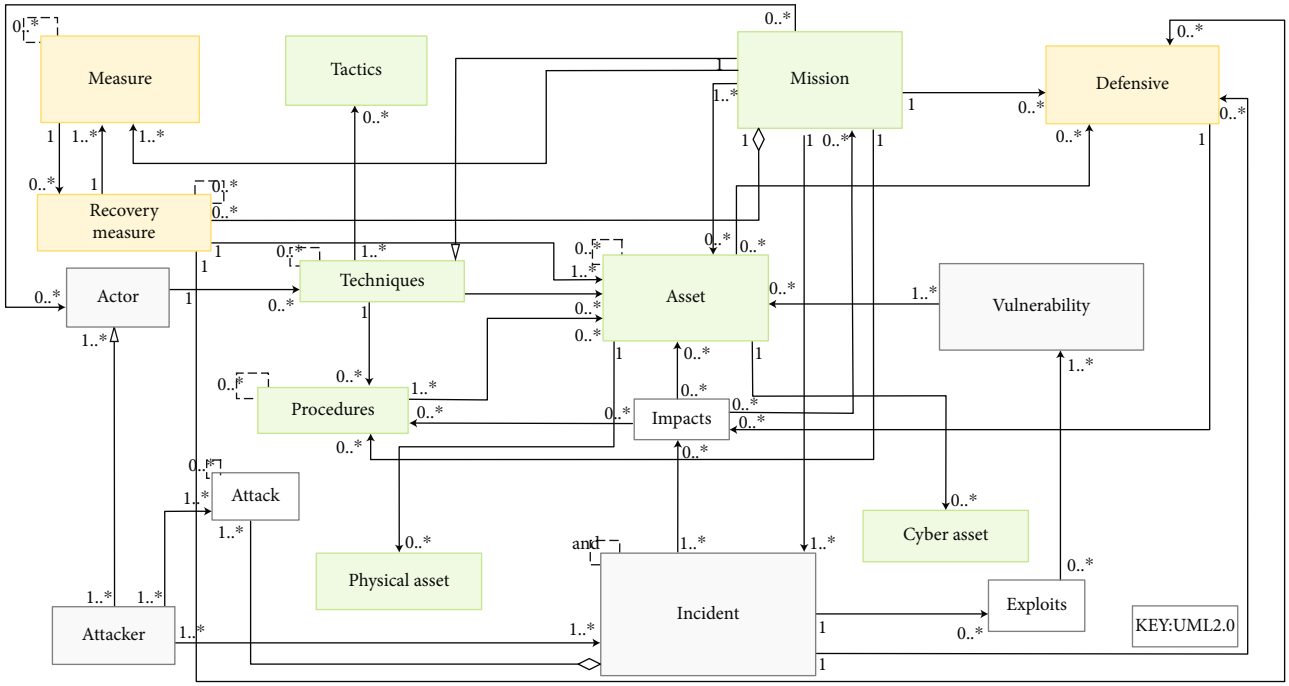

Figure 1: Schematic of the IIoT-based threat response metamodel. It shows the security concepts that can be used to instantiate TRMLang for further analysis, common understanding, and evaluation of complex attacks on control systems.

probabilistic attack graphs. Finally, we describe this correlation between attack and defense through an example of an attack simulation on a small control network.

The architecture of the small control network can be seen in Figure 2. In this scenario, the solid line represents the possible attack path that attackers can take to achieve their goals, and the dashed line represents the implemented defense. The attack simulation was developed based on premodeled models, scripts, and complementary self-developed code to deploy all phases of key techniques. The following sections of the scenario briefly provide the documentation according to TRMLang, while the full experiments are shown in Section 4.

The attacker uses t_i: *attemptPhishing* to send spear-phishing emails containing malicious links, typically to execute malicious code on victim systems. Phishing may also be conducted via third-party services, like social media platforms. Phishing may also involve social engineering techniques, such as posing as a trusted source. Users can also use defensive means, such as d_i: *userTraining*, to be aware of access or manipulation attempts by an adversary to reduce the risk of successful spear phishing, social engineering, and other techniques that involve user interaction.

The defense of the asset is denoted by "#." Bernoulli (0.4) means that the probability that the defense protected is enabled for an asset UserWorkstations is 0.4 (40%).

Attackers use the t_j: *drive-by compromise* technique to gain privileges from users while they are accessing the system normally. This attack usually targets specific organizations, industries, regions, etc., forming the so-called "watering hole" attack. There are two defensive measures to choose from, d_i: *data execution protect* (*DEP*) and d_j: *address space layout randomization* (*ASLR*). As presented above, classes containing attack actions constitute the core entities of a TRMLang specification. The description is as follows.

TRMLang is based on MAL syntax rules and generates an attack graph with probability distributions over the TTC of each attack action in a path by quantifying the dependencies of attacks and defenses.

As shown in Equation (5), TRMLang probability distribution requires the introduction of an attacker feature mapping algorithm. It can establish the attacker capability factor δ based on the attacker capability, which is used to quantify the attacker's sophistication attributes. This approach addresses a problem not considered in previous work, where the TTC distribution should change with different attacker attributes. Such a negative effect leads to a serious underestimation of the potential attack's impact. As such, this paper simulates the use of attacker sophistication attributes to adjust the TTC probability distribution over time. This allows fine-tuning of the well-trained model to adapt to different individuals [27]. In theory, this should make it easier to differentiate threat levels based on attackers. The details are described in detail in Section 3.

$$\forall_{\mathrm{DEP}\in\{0,1\},\mathrm{ASLR}\in\{0,1\}} \Longrightarrow \mathrm{exponential}(\mathrm{default}\times\delta). \quad (5)$$

The expression above means every entry in the dependency truth table of *WebServer*.. In *drive-by compromise*, theδshould be multiplied by it. Namely, the expression above would yield the dependency truth table, as in Table 1.

The two defenses, DEP and ASLR, will make the attack actions more difficult for the adversary. If both are set, then

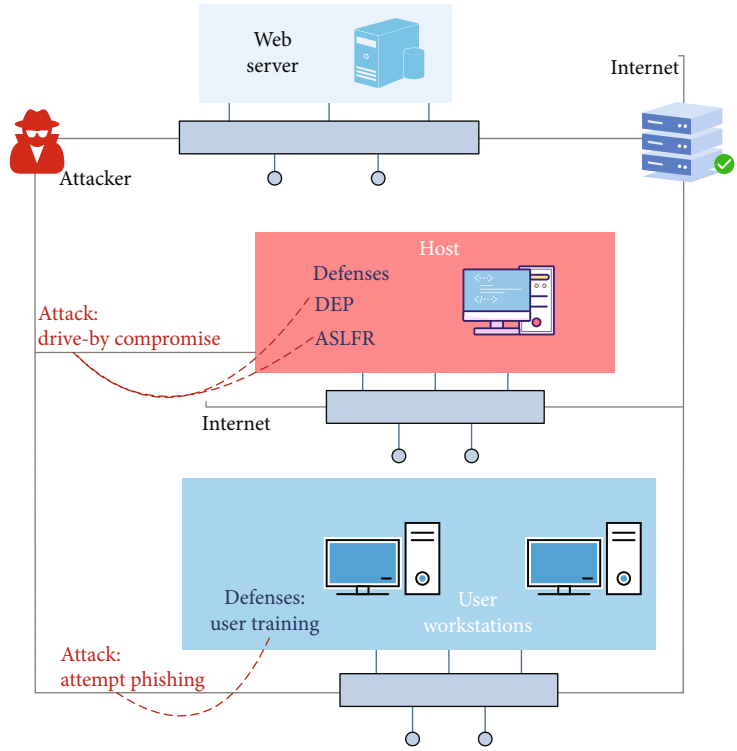

Figure 2: Schematic of attack and defense of small control network. It contains two examples of attack simulation, which can express the correlation between attack and defense.

```
asset UserWorkstations {
  | attemptPhishing
      ⟶ host. authenticate
  # userTraining [Bernoulli(0.4)]
      ⟶ attemptPhishing
}
```

Specification 1

```
asset WebServer extends Resource {
  & driveByCompromise
      ⟶ hosts.connect
}
asset Host extends Resource {
  | connect
      ⟶ access
  | authenticate
      ⟶ access
  # DEP
      ⟶ webserver.driveByCompromise
  # ASLR
      ⟶ webserver.driveByCompromise
}
```

Specification 2

the attacker cannot do anything, regardless of its capabilities. However, the attacker's ability factor δ is believed to have an effect when only one is set or not set.

3. Applying Attacker Profile for TRMLang Probability Distribution

3.1. ATT&CK Matrix. The MITRE's ATT&CK (adversarial tactics, techniques, and common knowledge) framework is the most widely known and utilized methodology for expressing the activity of a cyberattack or threat actor [28]. The MITRE ATT&CK framework analyzes the activities of cyberattacks and threat actors from the perspective of TTPs (tactics, techniques, and procedures) and composes and expresses them in the form of ATT&CK matrix. The framework assists in understanding the adversarial attack chain and enhances the security standpoint of IIoT and related control system assets.

There are three versions of this ATT&CK matrix: enterprise, mobile, and ICS. This study extracts six tactics from three versions of the ATT&CK matrix from the view point of IIoT networks. These tactics will map the attack actions in the TRMLang model to the attacker's techniques. The following six tactics are proved to be effective in threat modeling and attack simulation: *initial access*, *execution*, *privilege escalation*, *defense evasion*, *credential access*, and *lateral movement*. Each of the tactics presented can be deployed by different techniques. These tactics are chosen because they are useful in defining the attack action during the time between the initial attack and the compromised target. From a security engineering perspective, these six

Table 1: DEP and ASLR dependency truth table.

DEP	ASLR	TTC
0	0	exponential(default × δ)
0	1	exponential(default × δ)
1	0	exponential(default × δ)
1	1	exponential(default)

classifications of tactics are able to fully describe the threat elements of the system of interest. Moreover, as stated earlier, what we consider is the simulation of an attacker's movement in the control network. What the simulation does not take into consideration is what happens after the attacker reaches the target, such as the *impact* of a data breach. Therefore, the remaining tactics are difficult to accurately describe in ways of threat modeling.

The complex nature of IIoT control systems, and consequently the attack and defense logic, urges the modeling of attack vectors on both the cyber and the physical domains of the system. Attackers are constantly improving, adapting, and modifying their attack patterns to avoid defense mechanisms. As a consequence, to support logic modeling, path prediction, and time to compromise calculations in TRMLang, we combined the six core tactics of MITRE ATT&CK matrix in the attacker profile for a more granular and explicit TTP analysis. Specifically, the proposed threat modeling approach extends the ATT&CK matrix methods to comprehensively characterize the logic of the attack in the MAL syntactic structure. As such, our model will help to implement a more accurate simulation safety assessment in experimental testing.

3.2. Attacker Profile. The research in this paper introduces a methodology that automatically maps dynamic attacker profiles. The goal of this study is to provide a new indicator for the threat profile of a specific attacker through CTI automated analysis. The proposed methodology is divided into two main parts, as shown in Figure 3. The first part of the methodology utilizes CTI and ATT&CK matrices to automate the analysis of the attacker's TTP features for quantified tactical and technical data. The second part considers mapping the attacker's sophistication attributes to the capability factor δ. These two components together form the overall structure of the attacker profile, which is used to map the adversary's ability range and skill features into the TRMLang metamodel.

The basic unit of the capability factor δ is the sophistication attributes (SA). Although the overall skill can transfer dynamic features through the state values of each attribute, it cannot reveal the relationship between the dynamic change in input and the output. For complex attack processes, most of the time, it should be in a stable reconnaissance state. However, it is difficult for the attack to run latently for a long time due to continuous changes in the information of the system network, the state of the logged-in devices, and the unknown disturbances in the system, leading to continuous changes of the IIoT control system state. Hence, the capability factor δ is proposed to characterize the regularity between input data and key techniques by introducing the configuration information of SA as dynamic features.

Saade and Conference [29] analyzed the features of attackers from an epistemological perspective. Based on this research, we summarized nine sophistication attributes of attackers. The first group consists of 6 attributes: *maintenance intrusion tools*, *combining complex components*, *active reconnaissance*, *intruded system familiarity*, *victim familiarity*, and *same organization*. The attribute sets of this group are $\{0, 1\}$. The second group includes two attributes: *intrusion tools developed* and *intrusion experience*; the sets are $\{0, 1, 2\}$. Finally, the third group has only one attribute: *level of resources*, which has the set $\{0, 1, 2, 3\}$. The higher the value, the better the attacker has resources. These thresholds are merely a suggestion and can be configured in many different ways. One such configuration is proposed below, where δ assumes a value in $\{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$, as in Equation (6). This means that, in the best case, the time required for the attacker to compromise the target will be doubled.

$$\delta = \text{delta}\left[\sum(SA)\right] = \begin{cases} 1.0, & [1, 3], \\ 0.9, & [4, 5], \\ 0.8, & [6, 7], \\ 0.7, & [8, 9], \\ 0.6, & [10, 11], \\ 0.5, & [12, 13]. \end{cases} \tag{6}$$

3.3. Method for Assigning Probability Distributions. The main purpose of threat modeling and simulation of complex attack processes is to dynamically capture the features between attack processes, which are highly correlated with key attack techniques. A common issue for MAL-based languages is the lack of quantitative analysis. We perform uncertainty calculations considering the actual impact of attacker profiles on TRMLang and leveraging both the threat modeling and IIoT scenario asset mapping. Therefore, TRMLang modeling must define probability distributions for most attacks and defenses to provide more realistic simulation results for their system model instances. The modeling method of TRMLang is shown in Figure 3.

In *preparation work*, the most important task is choosing the TTP and TTC of a specific attack action according to prior attacker profile and the key techniques mapping of the attack process. The TTC of an attack action is generally determined by the attack time of a specific attack process.

The MAL framework definition specifies that the time to execute an attack follows a certain probability distribution. As shown in Table 2, available distribution functions to represent the required time include *Bernoulli*, *Exponential*, *Gamma*, *LogNormal*, *Pareto*, and *TruncatedNormal* distributions. For example, if the time required to execute the attack action t_j compromise the target is $1/\lambda$ days, then we express it with an exponential distribution with parameter

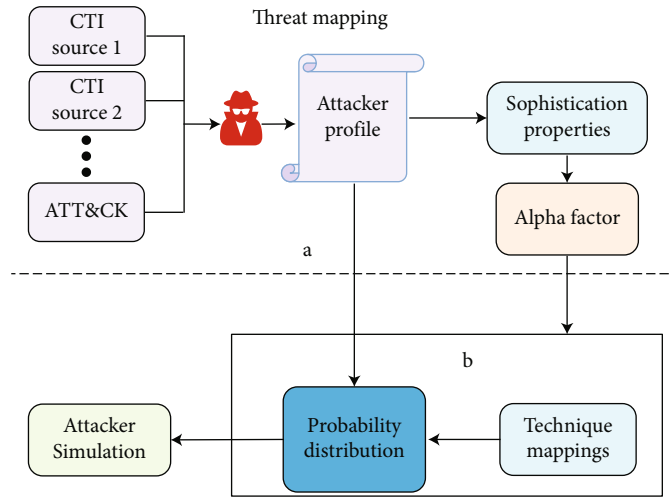

Figure 3: Graphical representation of the distribution probability distribution method. (a) The threat mapping phase involves automated CTI analysis, specifically the ATT&CK matrix, which has generated the attacker profile. (b) This phase accepts the mapped data for each key technique along with attacker's capability factor δ for assigning probability distributions, and it is used to construct code and scripts for attack simulation.

Table 2: Probability distributions supported by the MAL framework.

Distribution	Limits	Expected value
Bernoulli	$0 < = p < = 1$	$E[\text{Bernoulli}(p)] = p$
Exponential	$0 < \lambda$	$E[\text{Exponential}(\lambda)] = 1/\lambda$
Gamma	$0 < k, 0 < \theta$	$E[\text{Gamma}(k, \theta)] = k * \theta$
LogNormal	$0 < \theta$	$E[\text{LogNormal}(\mu, \theta)] = e\text{^}(\mu + \theta\text{^}2/2)$
Pareto	$0 < m, 0 < \theta$	$E[\text{Pareto}(m, \theta)] = \text{infinity if } m < 1, \text{otherwise } (m * \theta)/(\theta - 1)$
TruncatedNormal	$0 < \sigma$	$E[\text{TruncatedNormal}(\mu, \sigma)] = \mu$

λ, i.e., Exponential(λ). If a Bernoulli distribution is used in multiplication, e.g., Exponential(λ) $*$ Bernoulli(p), that means that the TTC of the attack action is Exponential(λ) with a probability of p and Infinity with a probability of $1 - p$.

In Table 2, p represents the probability, λ represents the rate, k and θ represent the shape and scale, m represents the minimum, μ represents the mean, and σ represents the standard deviation.

In order to better accommodate the dynamic features of attackers, specific improvement parameters η for key techniques are proposed to characterize the regularity between different attackers and key techniques by introducing attacker-specific discrepancy information as attacker features. η takes values in the range $[0, 1]$. The specific improvement parameters η do not scale with the capability factor δ, which is used to make slight improvements to the technique. Thus, the attacker who uses the technique gains some advantage compared to those who do not.

Distributions can also be combined with addition, subtraction, multiplication, division, and exponentiation, as in

$$\xi \in [\text{Add, Subtract, Divide, Multiply, Exponent}]. \quad (7)$$

The probability distribution function was modified by δ and η according to Table 2, as in

$$\begin{aligned}
\text{Bernoulli} &: \text{Bernoulli}(p * (1 + (\delta\xi\eta)))], \\
\text{Exponential} &: \text{Exponential}(\lambda * (\delta\xi\eta)), \\
\text{Gamma} &: \text{Gamma}(k, \theta * (\delta\xi\eta)), \\
\text{LogNormal} &: \text{LogNormal}(\mu, \theta * (\delta\xi\eta)), \\
\text{Pareto} &: \text{Pareto}(m, \theta * (\delta\xi\eta)), \\
\text{TruncatedNormal} &: \text{TruncatedNormal}(\mu, \sigma * (\delta\xi\eta)).
\end{aligned} \quad (8)$$

Then, the probability distribution of TRMLang needs to train a TTC-Global (TTCG) generation network for the instance model based on the dynamic attacker profile in order to link the key techniques to the attack action. The algorithm assumes that all local TTCs are independent of each other. The CTI automated analysis and ATT&CK presence of attacks are gathered to calculate the conditional

probability between TTC and attack action, i.e., $p(ttc_i|t_i) \in (0, 1)$ These probabilities are based on dynamic data that forms a dependence truth table $T(ttci)$ in the Technique-TTC mapping. The truth table may need to be normalized to eliminate null values. The normalization table is used to calculate the normalized likelihood or normalized conditional probability $\omega(ttc_i|t_i)$ to show the support of each key technique t_i to their calculated ttc_i shown in Equation (9).

$$\omega(ttc_i|t_i) = \frac{p(ttc_i|t_i)}{\sum_{t_i \in T_{ttc_i}} p(ttc_i|t_i)}, \tag{9}$$

where $\omega(ttc_i|t_i)$ is the normalized conditional probability, $p(ttc_i|t_i)$ is the conditional probability between TTC and the attack action, ttc_i is the calculated TTC, t_i is an attack action that relies on the truth table, $ttc_i|t_i$ refers to attack action t_i occurs under the condition that $ttci$ occurs, and T_{ttci} is the generated dependent truth table.

Using the above normalized conditional probability table, the normalized posterior probability $\mu(ttc_i|t_i)$ can be calculated by using Naive Bayes in Equation (10).

$$\mu(ttc_i|t_i) = \frac{\omega(ttc_i|t_i)p(t_i)}{\sum_{t_i \in T_{ttci}} \omega(ttc_i|t_i)p(t_i)}, \tag{10}$$

where $\mu(t_i|ttc_i)$ is the normalized posterior probability and $p(t_i)$ is the prior class probability.

The shortest paths between likely attacker entry points and the most critical assets represent the easiest way for a cyber attacker to compromise the IIoT control system. Assume that rational adversary would select the shortest path to reach an attack target. All calculated local TTCs are considered for this support and for each attack for different key techniques, so in order to extract TTCG, all these support values are combined to find the shortest compromise time with the maximum support value $S(t_i)$, as in

$$S(t_i) = \frac{\sum_{ttc_i \in TTCG_i,} \mu(t_i|ttc_i)}{\sum_{ttc_i \in TTC_{ti}} \mu(t_i|ttc_i)}, \tag{11}$$

where $S(t_i)$ is the attack support function, $TTCG_i$ is the shortest TTC computed due to the attack action $TTCG_i$, and TTC_{ti} is the set of local TTCs associated with the attack t_i.

Algorithm 1 presents the process of constructing a TTCG generation network for probabilistic attack graphs.

4. IIoT Case Study: Smart Grid Simulation

In this section, the proposed TRMLang model is simulated by the Lazarus Group penetration attack process to estimate the TTC distribution of the IIoT smart grid. Then, the effectiveness of the model is compared with the simulation based on standard penetration tests.

4.1. SEGRID Description. Figure 4 describes a scenario regarding central load balancing of SEGRID smart grid [30]. From an IIoT infrastructure point of view, this means that the traditional power grid infrastructure is extended with yet another type of substation, namely, the distributed energy resource (DER) generation that is monitored and controlled by the central supervisory control and data acquisition (SCADA) system. A traditional physical system viewpoint depicts networks (zones) operated by different stakeholders. These networks contain computer hosts and are connected to each other via firewalls. The SCADA zone is the most essential part of the structure. This is where commands from operators are delivered and then distributed to the actual substations. Looking to the left of the SCADA zone, we have the engineering zone. This is where the power system structure is defined. The office zone is where the staff not working with operating the process is located. Typically, an AMI zone contains systems for collecting readings of energy consumption from household and industrial. SEGRID is an extensive model consisting of several systems and suppliers, controlled by operators on a centralized system. This example model describes a real-world scenario well and adds complexity to our simulations when evaluating the result of our different attacker profiles, which makes it a good choice. Hence, the TRMLang model is applied to add the required tags to the affected assets.

As a basis for the security assessment, the experiment used a tool called securiCAD [31]. SecuriCAD makes its security risk assessments by performing probabilistic simulations of attack graphs on system architecture models. In short, securiCAD can use a domain-specific language that specifies IIoT assets, potential attack actions related to these assets, and defenses related to these assets. From model instances of the language, probabilistic attack maps are automatically generated and computed.

Test runs or samples are used to simulate different paths between runs, as some attack actions might succeed and sometimes fail, generating different TTC values. This makes sense as an attacker's skills, time, funds, and experience would vary between attacks. This difference affects the calculation of the TTC in the simulation. When the simulation is done, the probability results based on the cumulative distribution function are used for all the test sample data, and the average of all TTC values over time distribution is given in the end.

The overall result and the final security vulnerability assessment metric is that all attacks receive a probabilistic TTC distribution. A high TTC thus corresponds to a low risk since it takes a long time to compromise. Finally, in this experiment, in order to evaluate the effectiveness of the proposed method, two experiments were designed using the SEGRID smart grid project as the experimental background: Compromising Distribution System Operator (DSO) Office Computer and Compromising DSO Engineering Control Subnet.

4.2. Lazarus Group Attacker Profile Preparation. According to prior knowledge and mechanism of attack process mapping, 13 attack techniques out of 6 tactics that are highly related to the attack action of the Lazarus Group [32] are selected to be key techniques. All key techniques of the TRMLang model are presented in Table 3.

Data preparation: Choose the TTP and local TTC of a specific attack action according to the prior attacker profile and key techniques mapping of the attack process;
Input: Attacker Profile (AP) and local TTC set (T_{ttci});
Output: TRMLang model with integrated TTCG generation network.
Start:
Step 1: Get the attack action t_i of a specific attacker: $techniques \longleftarrow AP$, and calculate the capability factor δ based on the dynamic attributes of the attacker;
Step 2: Obtain the assets associations ($Asset.Connection$) and the defensive distribution ($Asset.Defenses$) in a defined scene;
Step 3:
for $S_i \in Asset.Connection$ **do**
Calculate the probability distribution $p(t_i)$ for attack action t_i:
$p(t_i) \longleftarrow new\_distribution(Asset.Defenses, \delta, t_i.\mu)$
end for
Step 4: Check the dependency table $p(ttc_i|t_i)$. If the technique t_i is related to the local TTC set, go to Step 5, otherwise go back to Step 1;
Step 5: Calculate the conditional probability $p(ttc_i|t_i) \in (0, 1)$ based on CTI automated analysis and ATT&CK collection of attack action t_i.
Step 6:
for $t_i \in T$ **do**
Calculate the normalized conditional probability $\omega(ttc_i|t_i)$;
Use the prior class probability $p(t_i)$ to calculate the normalized posterior probability $\mu(t_i|ttc_i)$.
end for
Step 7: Consider all T_{ttci} to calculate the maximal support of $TTCG_i$ for t_i and go to Step 8;
Step 8: If the technique t_i for a given ttc_i has the maximum posterior probability, then add technique ti to the TTCG generation network and exit. Otherwise, go back to Step 1.
End

Algorithm 1: TTCG generation network based on TRMLang.

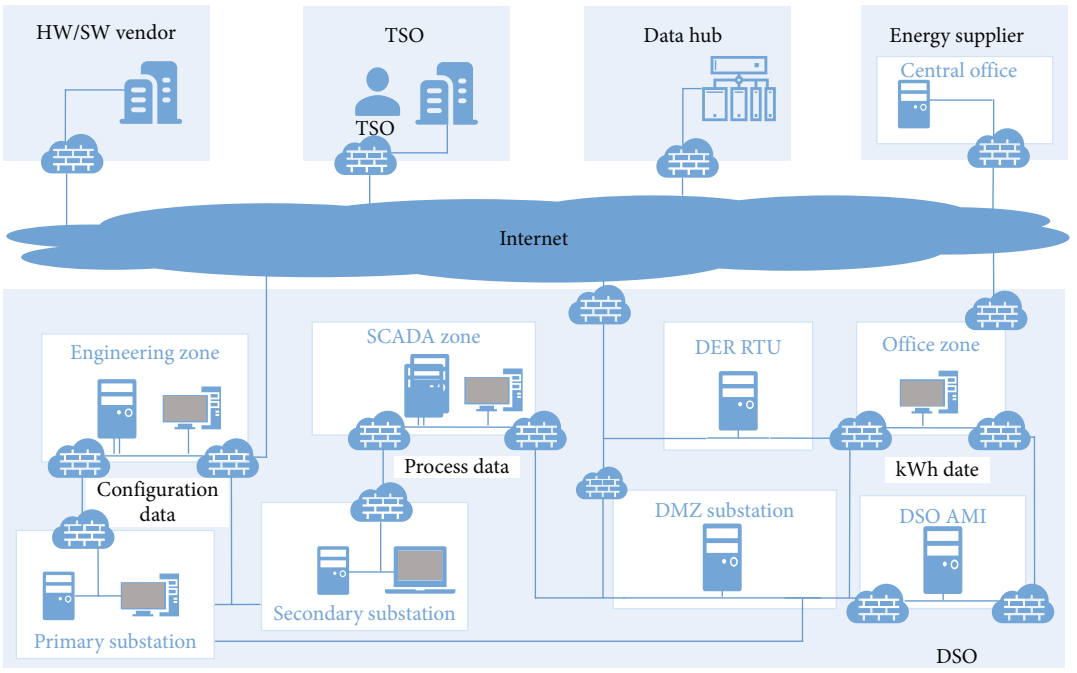

Figure 4: Diagram of the SEGRID smart grid network. In the environment in which the distribution control system operates, data for distributing control commands along with collecting status information are variously combined and passed through the infrastructure components.

Here, it can be analyzed some conclusions that the *initial access* attacks of Lazarus Group mostly send malicious code to the victim through a compromised legitimate website. The strategy is typically known as a "watering hole" attack, which is a type of drive-by compromise tactic. Also, there is an attack strategy of the same type, known as "phishing."

Table 3: In-scope key techniques for the Lazarus Group.

Tactic	Technique
Initial access	Drive-by compromise, phishing
Execution	Command and scripting interpreter, exploitation for client execution, and user execution
Privilege escalation	Account manipulation
Defense evasion	File deletion, modify registry, indicator removal on host, and software packing
Credential access	Input capture, password spraying
Lateral movement	Remote services

In this attack, spear phishing was used as the initial infection vector. Before launching the attack, the group studied publicly available information about the targeted organization and identified email addresses belonging to various departments of the company. Email addresses in those departments received phishing emails that either had a malicious Word document attached or a link to one hosted on a remote server. If the malicious code is executed, then the privilege escalation technique is used in the next step to help them gain an account with advanced privileges for further access to the environment. After gaining an initial foothold, the attackers gathered credentials and moved laterally, seeking crucial assets in the victim's environment. They shared tools and infrastructure among these campaigns to achieve their goals.

Specially, the Lazarus Group historical data from CTI automated analysis also reflects the regularity of changes in attacker features. Thus, the previous infiltration activities of the Lazarus Group also demonstrate its sophistication and capabilities, which can be used as an additional auxiliary characteristic variable.

According to the previously defined formula for calculating the sophistication attributes, the value of the capability factor for the Lazarus Group is $\text{delta}(13) = 0.5$.

The following experiments employ all the key techniques that have been listed in this section. They provide a practical guide on how to perform the attack, following an ordered approach of the implementation phases of the adversary emulation scenario.

4.3. Compromising Central Office Zone. The TRMLang model facilitates a probabilistic attack graph that is oriented to the attack flow, as shown in Figure 5. When examining security concerns, it is often advantageous to see how attacks move laterally. The flow of attack, or associations in assets, through an IIoT control systems can illuminate possible threat vectors and thus provide insight into appropriate areas for security controls. In Figure 5, the standard penetration test is performed by sending a malicious phishing email to gain initial access to the *DSO_OfficeComputer*. The central office zone is where the staff not working with operating the process is located. They may not be sufficiently security precautions aware and end up clicking on a malicious link in the mail or downloads malicious documents which appear legitimate. This action leads to the attacker gaining access to the employee's host. When this state has been reached, the central office zone can be compromised. In Figure 6, we can see that the standard penetration test attack performs successfully in the TTC distribution.

For penetration testing of the applied TRMLang model with the Lazarus Group attacker profile, the most effective attack remains malicious email. The attackers used malicious emails to infect computers in the central office zone by initiating attacks to gain employee access to the browser. The payload created by the initial spear phishing document is loaded as a backdoor installer running in memory. This installer is responsible for implanting the next stage loader type of malware and registering it for automatic execution for persistence. Later, we assess that the attacker successfully obtains login credentials from the host and starts using them for further malicious activities. After obtaining login credentials, this attacker begins to move laterally from the office host to the control subnet host. The TTC distribution curve applying the Lazarus Group attack profile is shown in Figure 7.

In the case where the same entry point is selected for both tests, the global TTC distribution remained more or less the same. In the attack actions related to malicious mail service, the TTC distribution with the Lazarus Group attacker profile has not been significantly improved, apart from the 90% success rate. However, the Lazarus Group attack path is simpler as the key technical features of the attacker were correctly mapped onto the probabilistic attack graph. This highlights the fact that just because Lazarus Group is able to get initial access through a drive-by compromise solution, it does not mean that it is always the best solution. This can be a much more discrete problem, because puddle attacks infecting legitimate websites may not alert intrusion detection systems and are more difficult to detect than email phishing. But the downside of the drive-by compromise solution is that one must wait until the target accesses the infected site, while one can expect a faster compromise response when this happens.

4.4. Compromising DSO Engineering Control Subnet. When attackers compromise less critical assets, such as DSO office computers, they want to use lateral movement techniques to penetrate the engineering control subnet of critical services or infrastructure, as shown in Figure 8. The DSO engineering control subnet is where the power system structure is defined in the SEGRID project. Compromising this network would allow an adversary to control the power system, which could be motivated by the Lazarus Group targeting the DSO engineering control subnet's essential foothold in

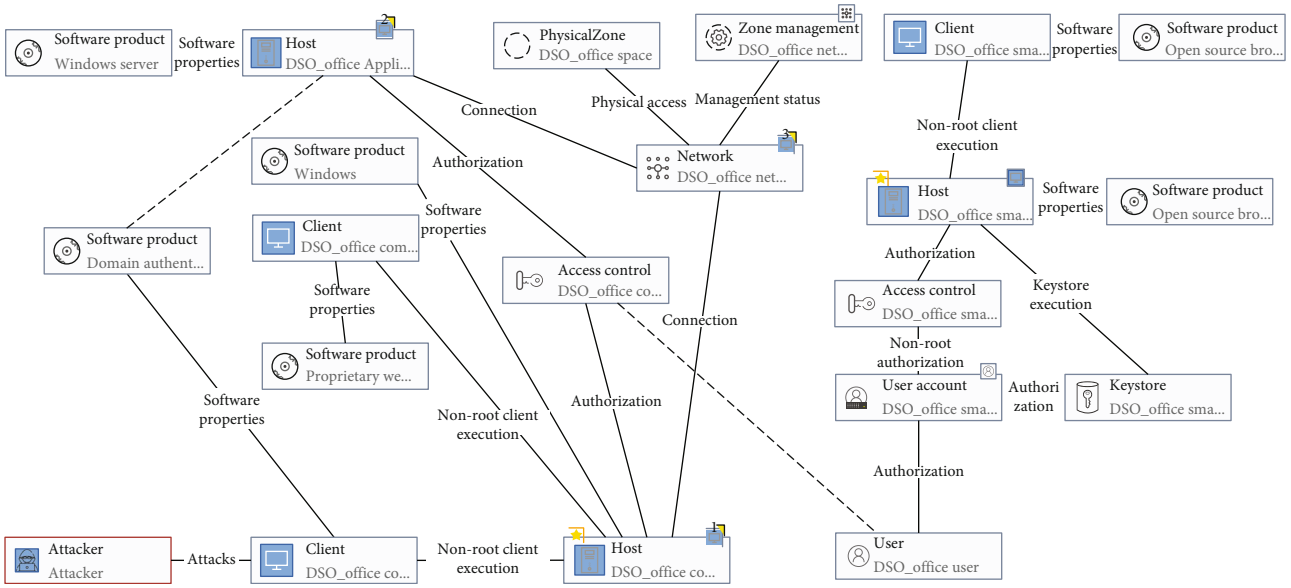

Figure 5: Compromising central office zone. This is where the attacker's attack is moved laterally and then compromised into the office host.

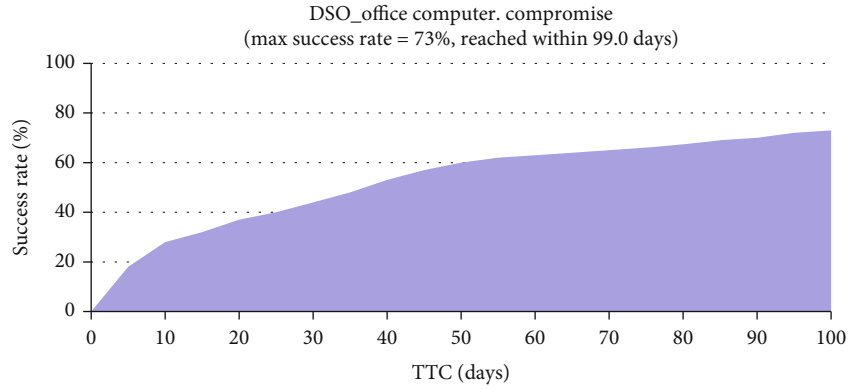

Figure 6: TTC distribution of standard penetration tests for compromising central office zone.

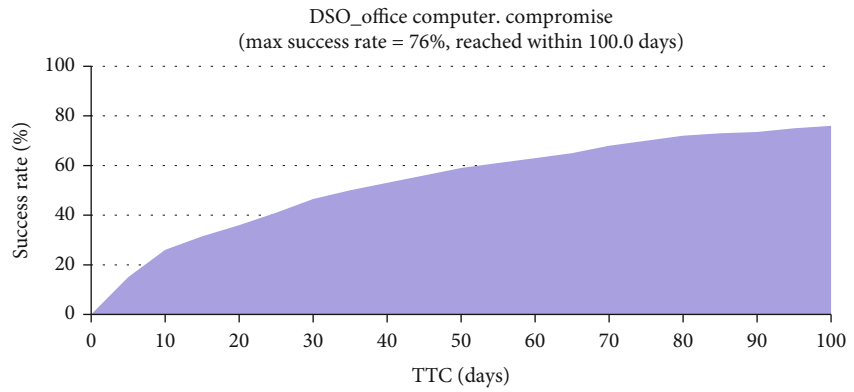

Figure 7: TTC distribution of the Lazarus Group penetration tests for compromising central office zone.

the system. The TTC distribution curve for a standard penetration test attack is shown in Figure 9.

The Lazarus Group begins by initiating the attack by infecting a web server with malicious code. Once the malicious code has gained entry into the system, the attack will typically evolve through the different stages of the kill chain. It carries out early reconnaissance, creates a state of persistence, seeks access to a user in the *DSO_*

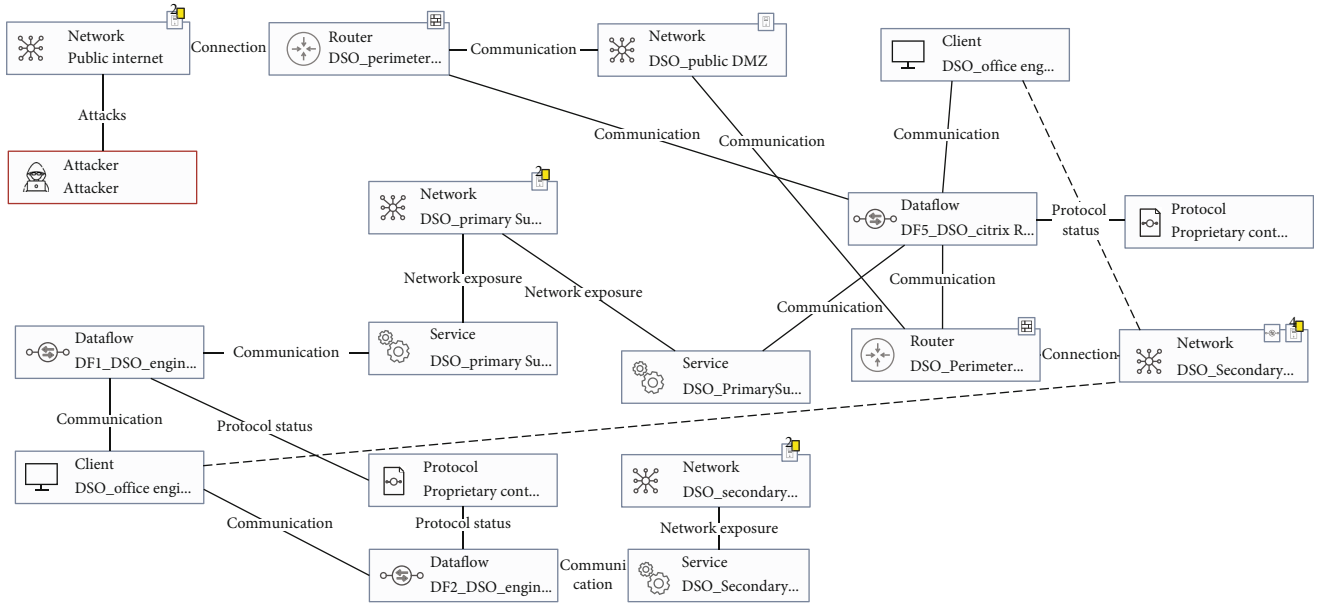

Figure 8: Compromising DSO engineering control subnet. This is where the attacker compromises the control subnet through attack paths.

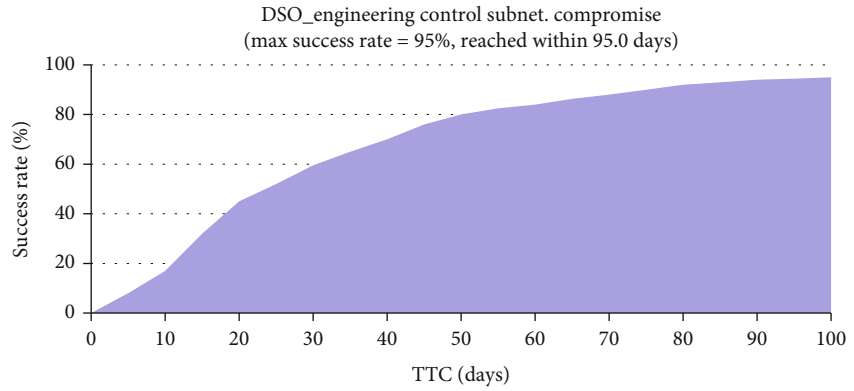

Figure 9: TTC distribution of standard penetration tests for compromising DSO engineering control subnet.

EngineeringControlSubnet through a DSO server, and then initiates a series of *lateral movements* or *exfiltration* attacks until it reaches its final goal of compromising the engineering control subnet (other impacts are possible as well of course). This accurately represents the dynamic behavior feature of the Lazarus Group, as shown in Figure 10, where the success rate of the TTC distribution grows slightly faster.

4.5. Experimental Method Comparison. Previous threat modeling work has involved domains that are primarily based on model-driven architecture modeling. Architecture modeling can help systems deal with increasingly complex cyber environments [33–36]. The importance of creating models to support security decisions has been previously demonstrated in some studies, including HinCTI [37], TVHARM [38], and PMCAP [39], also employed architectural modeling, in which attacks and defenses were coupled to the system architecture. However, these methods are "modelling only" models but do not provide any automatic methods to analyze and infer further conclusions from the model. Therefore, this type of work is not as relevant as it may seem. Consequently, this type of work is not as close to us as it may seem.

The current TMS models focus on different functional scopes. Some methods focus on the inference of attack intent and attack path [16–18, 40], and some methods focus on the uncertainty analysis of the attack [16, 41, 42]. There are also many studies that do not take into account the uncertainty of attack occurrence, thus failing to model real and valid attacks. The threat response modeling language proposed in this paper focuses on information on IIoT assets, uncertainty analysis, tactics, techniques, and attacker features, which provides a holistic view of the security situation and is more comprehensive than other methods. Table 4 compares TRMLang and other TMS models to show the scope of functionality involved in each model.

The experiments verified the effectiveness of the method in this paper. TRMLang extracts and models information

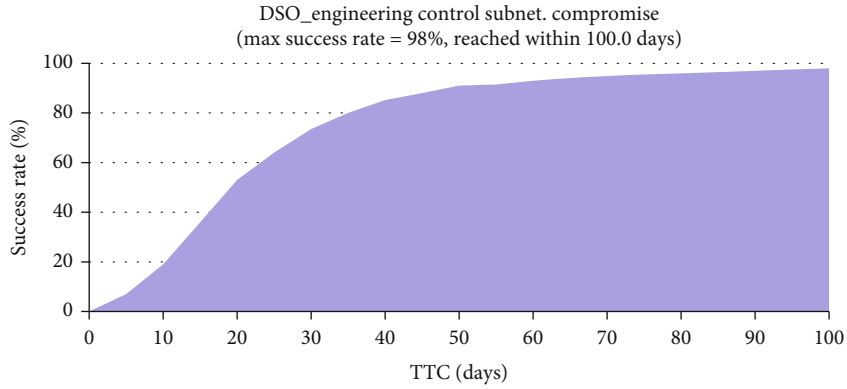

Figure 10: TTC distribution of the Lazarus Group penetration tests for compromising DSO engineering control subnet.

Table 4: Comparison of experimental methods of several TMS models.

Method	Security engineering	Attack intent inference	Attack path inference	Uncertainty analysis	Tactic, technique	Attacker features
P2CySeMoL [41]			✓	✓		
pwnPr3d [42]			✓	✓		
SecML [43]		✓			✓	
FMML^x [40]	✓	✓	✓			
BPATSM [44]	✓	✓				
PowerLang [17]	✓	✓	✓			
CoreLang [18]	✓	✓	✓			
EnterpriseLang [16, 45]	✓	✓	✓	✓		
IoTsecM [46]	✓				✓	
TRMLang	✓	✓	✓	✓	✓	✓

about IIoT target networks and attackers. By correlating the system asset model and attacker features, the state transfer and TTC probability distribution during the attack process are simulated. Then, the global TTC generation network is used to find out the minimal time of the attacker's pervasion and then infer the attack intention and path, which provides useful evidence and guidance for making a risk decision.

5. Conclusions

In this paper, in order to capture the impact of attacker dynamic features on key technologies, a novel attack simulation formal approach called TRMLang was proposed for threat modeling and simulation in the IIoT domain. The attacker profile of the dynamically constructed, along with the attack and defense logic of the key technologies, was integrated into the threat model to learn the actual dynamic characteristics of the industrial process. Then, the effectiveness of TRMLang was demonstrated by comparing it with a model based on standard penetration tests in two case studies on smart grids. Therefore, the use of this approach can track the attacker's attack actions based on highly vulnerable assets within IIoT control systems, efficiently calculating the time spent on the attack process.

For future work, first, the TRMLang model allows comparing the attacker's key techniques with those of other groups due to the complexity of the attacker's key techniques attack and defense logic in the actual IIoT process. Then, when the model conditions change, the modeled probabilistic attack graph may no longer adapt to the system. Therefore, it was important to explore the automated modeling method for TRMLang so that the model can be periodically updated. Finally, more knowledge of security engineering mechanisms should be combined to build an IIoT threat model.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Key Program Research Fund of Higher Education of Henan, China (18B520044 and 19A520048), and the Science and Technique Foundation of Henan, China (182102210526).

References

[1] L. Wang, Z. Ye, R. Zhang, J. Lin, F. Chen, and F. Tang, "The growth model of industrial internet platform in industrial 4.0," *Wireless Communications and Mobile Computing*, vol. 2022, Article ID 5145641, 9 pages, 2022.

[2] B. Yan, Y. Jeong-Bong, and L. Il-Kyoo, "Design and optimization of smart factory control system based on digital twin system model," *Mathematical Problems in Engineering*, vol. 2022, Article ID 2596946, 16 pages, 2021.

[3] M. K. Hasan, A. Alkhalifah, S. Islam et al., "Blockchain technology on smart grid, energy trading, and big data: security issues, challenges, and recommendations," *Wireless Communications and Mobile Computing*, vol. 2022, Article ID 9065768, 26 pages, 2022.

[4] S. Le, Y. Qiandi, S. Sudha, and W. Xuyang, "Fogmed: a fog-based framework for disease prognosis based medical sensor data streams," *Computers, Materials & Continua*, vol. 66, no. 1, pp. 603–619, 2020.

[5] M. Akbanov, V. G. Vassilakis, and M. D. Logothetis, "Ransomware detection and mitigation using software-defined networking: the case of WannaCry," *Computers & Electrical Engineering*, vol. 76, no. 1, pp. 111–121, 2019.

[6] J. Zhou, X. Wang, C. Yang, and W. Xiong, "A novel soft sensor modeling approach based on difference-LSTM for complex industrial process," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 5, pp. 2955–2964, 2022.

[7] S. Zhang, G. Bai, H. Li, P. Liu, M. Zhang, and S. Li, "Multi-source knowledge reasoning for data-driven IoT security," *Sensors*, vol. 21, no. 22, p. 7579, 2021.

[8] L. Sun, R. Zhou, D. Peng, A. Bouguettaya, and Y. Zhang, "Automatically building service-based systems with function relaxation," *IEEE Transactions on Cybernetics*, vol. 52, pp. 1–14, 2022.

[9] J. Liang and L. Jin, "Multi-perspective modeling of computer sales system based on unified modeling language," in *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, pp. 109–113, Chongqing, China, 2020.

[10] H. Grichi, O. Mosbahi, M. Khalgui, and Z. Li, "An extended object constraint language for adaptive discrete event systems with application to reconfigurable wireless sensor networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 10, pp. 3562–3576, 2020.

[11] W.-C. Hsu and I.-E. Liao, "UCIS-X: an updatable compact indexing scheme for efficient extensible markup language document updating and query evaluation," *IEEE Access*, vol. 8, no. 1, pp. 176375–176392, 2020.

[12] S. Arshad, M. Alam, S. Al-Kuwari, and M. H. A. Khan, "Attack specification language: domain specific language for dynamic rraining in cyber range," in *2021 IEEE Global Engineering Education Conference (EDUCON), p*, pp. 873–879, Vienna, Austria, 2021.

[13] M. Briland and F. Bouquet, "A language for modelling false data injection attacks in Internet of Things," in *2021 IEEE/ACM 3rd International Workshop on Software Engineering Research and Practices for the IoT (SERP4IoT)*, pp. 1–8, Madrid, Spain, 2021.

[14] M. Kern, E. Taspolatoglu, F. Scheytt et al., "An architecture-based modeling approach using data flows for zone concepts in industry 4.0," in *2020 IEEE International Symposium on Systems Engineering (ISSE)*, pp. 1–8, Vienna, Austria, 2020.

[15] P. Johnson, R. Lagerström, and M. Ekstedt, "A meta language for threat modeling and attack simulations," in *Proceedings of the 13th International Conference on Availability, Reliability and Security*, pp. 1–8, Hamburg, Germany, 2018.

[16] W. Xiong, E. Legrand, O. Åberg, and R. Lagerström, "Cyber security threat modeling based on the MITRE enterprise ATT&CK matrix," *Software and Systems Modeling*, vol. 21, no. 1, pp. 157–177, 2022.

[17] S. Hacks, S. Katsikeas, E. Ling, R. Lagerström, and M. Ekstedt, "powerLang: a probabilistic attack simulation language for the power domain," *Energy Informatics*, vol. 3, no. 1, pp. 1–17, 2020.

[18] S. Katsikeas, S. Hacks, P. Johnson et al., "An attack simulation language for the IT domain," in *Graphical Models for Security: 7th International Workshop*, pp. 67–86, Boston, MA, USA, 2020.

[19] V. Atluri and J. Horne, "A machine learning based threat intelligence framework for industrial control system network traffic indicators of compromise," in *SoutheastCon 2021*, pp. 1–5, Atlanta, GA, USA, 2021.

[20] Z. Yinghai, T. Yi, Y. Ming, X. Chuanyu, and L. Hai, "CTI view: APT threat intelligence analysis system," *Wireless Communications and Mobile Computing*, vol. 2022, Article ID 9875199, 15 pages, 2022.

[21] N. Moustafa, K. K. R. Choo, and A. M. Abu-Mahfouz, "Guest editorial: AI-enabled threat intelligence and hunting microservices for distributed industrial IoT system," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 1892–1895, 2021.

[22] O. Cabana, A. M. Youssef, M. Debbabi et al., "Threat intelligence generation using network telescope data for industrial control systems," *IEEE Transactions on Information Forensics and Security*, vol. 16, no. 1, pp. 3355–3370, 2021.

[23] L. Sun and J. Wu, "A scalable and transferable federated learning system for classifying healthcare sensor data," *IEEE Journal of Biomedical and Health Informatics*, vol. 26, p. 1, 2022.

[24] W. Yilin, S. Le, and S. Sudha, "CAB: classifying arrhythmias based on imbalanced sensor data," *KSII Transactions on Internet & Information Systems*, vol. 15, no. 7, pp. 2304–2320, 2021.

[25] U. Noor, Z. Anwar, U. Noor, Z. Anwar, and Z. Rashid, "An association rule mining-based framework for profiling regularities in tactics techniques and procedures of cyber threat actors," in *2018 International Conference on Smart Computing and Electronic Enterprise, Faculty of Computer and Information Technology Al-Madinah International University (ICSCEE)*, pp. 1–6, Shah Alam, Malaysia, 2018.

[26] Y. Lei, K. Ben, and Z. He, "A framework for self-adaptive software based on extended tropos goal model," in *2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics*, pp. 533–536, Hangzhou, China, 2015.

[27] Q. Yu and L. Sun, "LPClass: lightweight personalized sensor data classification in computational social systems," *IEEE Transactions on Computational Social Systems*, vol. 9, pp. 1–11, 2022.

[28] Y. Shin, K. Kim, J. J. Lee, and K. Lee, "ART: automated reclassification for threat actors based on ATT&CK matrix similarity," in *2021 World Automation Congress (WAC)*, pp. 15–20, Taipei, Taiwan, 2021.

[29] J. G. Saade and V. B. Conference, "Draw me like one of your french apts—expanding our descriptive palette for cyber threat actors," 2018, https://www.virusbulletin.com/conference/vb2021/.

[30] J. E. Y. Rossebø, R. Wolthuis, F. Fransen, G. Björkman, and N. Medeiros, "An enhanced risk-assessment methodology for smart grids," *Computer*, vol. 50, no. 4, pp. 62–71, 2017.

[31] M. Ekstedt, P. Johnson, R. Lagerström, D. Gorton, J. Nydren, and K. Shahzad, "Securi CAD by Foreseeti: a CAD tool for enterprise cyber security management," in *2015 IEEE 19th International Enterprise Distributed Object Computing Workshop*, pp. 152–155, Adelaide, Australia, 2015.

[32] US DEPARTMENT OF THE TREASURY, "Treasury sanctions north Korean state-sponsored malicious cyber groups," 2019, https://home.treasury.gov/news/press-releases/sm774.

[33] Z. Shuqin, Z. Minzhi, L. Hong, and B. Guangyao, "Threat analysis of IoT security knowledge graph based on confidence," *International Symposium on Emerging Technologies for Education*, vol. 13089, no. 1, pp. 254–264, 2021.

[34] M. Pelcat, A. Mercat, K. Desnos et al., "Reproducible evaluation of system efficiency with a model of architecture: from theory to practice," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 10, pp. 2050–2063, 2018.

[35] M. Efatmaneshnik, S. Shoval, and K. Joiner, "System test architecture evaluation: a probabilistic modeling approach," *IEEE Systems Journal*, vol. 13, no. 4, pp. 3651–3662, 2019.

[36] R. Heinrich, M. Strittmatter, and R. Reussner, "A layered reference architecture for metamodels to tailor quality modeling and analysis," *IEEE Transactions on Software Engineering*, vol. 47, no. 4, pp. 775–800, 2021.

[37] Y. Gao, X. Li, H. Peng, B. Fang, and P. S. Yu, "HinCTI: a cyber threat intelligence modeling and identification system based on heterogeneous information network," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 2, pp. 708–722, 2022.

[38] T. Eom, J. B. Hong, S. An, J. S. Park, and D. S. Kim, "A systematic approach to threat modeling and security analysis for software defined networking," *IEEE Access*, vol. 7, pp. 137432–137445, 2019.

[39] P. Jiaye and Z. Yi, "PMCAP: a threat model of process memory data on the windows operating system," *Security and Communication Networks*, vol. 2017, Article ID 4621587, 15 pages, 2017.

[40] S. de Kinderen and M. Kaczmarek-Heß, "Multi-level modeling as a language architecture for reference models: on the example of the smart grid domain," in *2019 IEEE 21st Conference on Business Informatics (CBI)*, pp. 174–183, Moscow, Russia, 2019.

[41] H. Holm, K. Shahzad, M. Buschle, and M. Ekstedt, "P^2CySeMoL: predictive, probabilistic cyber security modeling language," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 6, pp. 626–639, 2015.

[42] P. Johnson, A. Vernotte, M. Ekstedt, and R. Lagerström, "pwnPr3d: an attack-graph-driven probabilistic threat-modeling approach," in *2016 11th International Conference on Availability, Reliability and Security (ARES)*, pp. 278–283, Salzburg, Austria, 2016.

[43] C. Easttom, "SecML: a proposed modeling language for cybersecurity," in *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference*, pp. 1015–1021, New York, NY, USA, 2019.

[44] Z. Tian, W. Wu, S. Li, X. Li, Y. Sun, and Z. Chen, "A security model of SCADA system based on attack tree," in *2019 IEEE 3rd Conference on Energy Internet and Energy System Integration (EI2)*, pp. 2653–2658, Changsha, China, 2019.

[45] W. Xiong, S. Hacks, and R. Lagerström, "A method for assigning probability distributions in attack simulation languages," *Complex Systems Informatics and Modeling Quarterly*, vol. 26, no. 1, pp. 2955–2964, 2021.

[46] P. J. Escamilla-Ambrosio, D. A. Robles-Ramírez, T. Tryfonas, A. Rodriguez-Mota, G. Gallegos-Garcia, and M. Salinas-Rosales, "IoTsecM: a UML/SysML extension for internet of things security modeling," *IEEE Access*, vol. 9, no. 1, pp. 154112–154135, 2021.