

Impact Evaluation of Attacks on Data Flows in Systems With Heterogeneous Redundant Resources

Kosei Shimoda

Kyoto University

Kyoto, Japan

shimoda@inet.media.kyoto-u.ac.jp

Daisuke Kotani

Kyoto University

Kyoto, Japan

kotani@media.kyoto-u.ac.jp

Yasuo Okabe

Kyoto University

Kyoto, Japan

okabe@i.kyoto-u.ac.jp

Abstract—Cyber-physical and IoT systems gather diverse real-world data through systems created by heterogeneous hardware, software resources, and communication technologies, which introduce various threats and countermeasures. Although protecting data transmitted over the system is critical, previous work has not shown how to model these threats and countermeasures to evaluate attack impacts in each threat category. We propose a framework to evaluate the attack impact on data flows in systems with heterogeneous resources by representing the threats existing in each node and the countermeasures applied to data flows in their respective categories. Our framework determines whether a dataflow is compromised or not for each threat category. We define a hierarchical model of resources and vulnerabilities, represent threats and countermeasures by vectors, and develop a metric to quantify attack impacts. To validate the framework, we assess a simple IoT sensor system combining ZigBee and LoRa, demonstrating how it captures protocol and device differences. As a result, our framework flags only threats with realistic attack risks as compromised.

Index Terms—Heterogeneous Systems, Data Flow, Security Evaluation

I. INTRODUCTION

Cyber-physical and IoT systems collect a wide variety of real-world data, requiring the integration of heterogeneous hardware, software resources, and communication protocols [1]–[3]. For example, sensors measure different parameters such as temperature, acceleration, or water level in systems in many domains. Also, different communication methods can be used together to balance throughput and wide-area coverage.

Such heterogeneity is strongly related to the system's security. Resource heterogeneity can limit the damage caused by one vulnerability, thereby enhancing reliability [4]–[6], but it also introduces various threats. For instance, an OS-specific authentication vulnerability compromises only nodes running that OS, whereas other nodes remain unaffected. Likewise, encryption or authentication might protect data but be absent in others, creating uneven security across the same system.

Protecting data in these systems is critical since data from sensors often guides real-time decisions or critical processes. Compromises of data in integrity or availability prevent the system from understanding real-world environment correctly and timely, which may lead to inappropriate decisions made by the system. Similarly, breaches of confidentiality cause serious risks in domains handling sensitive data, such as healthcare.

To protect data, the systems usually apply various countermeasures according to threats, which are effects caused by attacks. Tampering, replay, and delay are examples of threats to data, and encryption and signatures are examples of countermeasures against specific threats. Threats to data are different according to the system's characteristics and the status of countermeasures. For example, confidentiality of data is not compromised if the data is encrypted, even if the attacker intrudes the devices that only transmit data to other devices without any modification. Thus, we need to evaluate the attack impact on data flows based on both attacks to each device in the system and the countermeasures applied to the data.

Existing approaches to heterogeneous system security tend to focus on attacker actions (e.g., cost, probability) [7]–[12] rather than the resulting impact on data flows. Likewise, generic threat modeling [13]–[16] does not specify how to incorporate heterogeneity inside the system to evaluate attack outcomes for each flow. Hence, we need to extend these models to evaluate how attacker's activities affect data transmitted over the system by incorporating data flows, threats to data, and countermeasures against threats into the model.

We propose a framework to evaluate the attack impact on data flows in systems with heterogeneous resources by representing the threats existing in each node and the countermeasures applied to data flows in their respective categories. It evaluates the presence or absence of compromise for each threat category based on the mapping between a threat and a countermeasure. We model a system's network using a hierarchical graph, define each data flow by its path and the countermeasures applied, and represent threats and countermeasures in the system using sets of categories and boolean vectors over the sets. Threats and countermeasures are modeled at node and data-flow levels. This enables us to capture how heterogeneous resources and vulnerabilities lead to distinct threats, and how different communication methods contribute to data-flow protection.

The remainder of this paper is as follows. Section II surveys related works. Section III details our proposed framework. Section IV shows the evaluation, and Section V discusses the results and implications. Section VI concludes the paper.

II. RELATED WORKS

Many studies have explored the relationship between heterogeneity and security properties in a system. Zhang et al. [4] surveyed heterogeneity-based security approaches in cyber-physical systems, noting that diverse resources can limit the impact of a single vulnerability. Meanwhile, some IoT systems inherently contain device-level and reliability-level differences [1], or involve multiple network architectures [2].

Research on heterogeneity and security typically proceeds in two directions: (1) introducing resource diversity to boost resilience against unknown exploits, and (2) evaluating the security effects of existing heterogeneity.

For instance, Hu et al. [17] arranged functionally equivalent but diverse components to resist zero-day vulnerabilities, and Prieto et al. [6] diversified homogeneous nodes to mitigate simultaneous exploits. Touhiduzzaman et al. [9] modeled the placement of security mechanisms in a power grid via a graph-coloring approach. Such methods focus on improving overall connectivity or robustness by introducing heterogeneity, but do not address how the heterogeneity impacts each data flow's security.

To evaluate effects by heterogeneity, other previous work uses attack graphs or host-connectivity graphs, placing vulnerabilities or nodes as vertices [7], [8], [10]–[12]. These studies often quantify attack cost, success probability, or reachability, yet do not show how attacks affect individual flows or how defenses vary across flows. Moreover, some metrics quantify overall diversity [5] or common attack surface [18], but do not directly evaluate per-flow impacts.

Threat modeling frameworks [13]–[15], [19] offer structured steps to identify and categorize potential threats (e.g., Spoofing, Tampering, DoS). However, they do not clarify which system elements must be modeled or at what level of granularity. As a result, it remains unclear how heterogeneous resources and vulnerabilities lead to distinct threats or protection of data.

Extensive mappings between threats and defenses, like MITRE ATT&CK [20] and ENISA guidelines [21], enumerate broad tactics, techniques, and mitigations. However, they do not provide a method to represent what threats exist in an actual system or how they are mitigated by countermeasures.

III. ATTACK IMPACT EVALUATION FRAMEWORK

We propose a framework for evaluating the attack impact on data transmitted over the system consisting of heterogeneous components. Figure 1 shows an overview of the proposed framework. The framework consists of two parts: the model of the system and the algorithm for evaluating the attack impact.

The framework models the system as a hierarchical, two-layered graph. The lower layer represents components and vulnerabilities of each node, and provides the threats of each node when data is handled at the node. The upper layer represents the data flows between nodes and the countermeasures applied to them. By representing resources and vulnerabilities per node at the lower layer, we can explicitly express the heterogeneity of the system. Actual evaluation at the lower layer is assumed

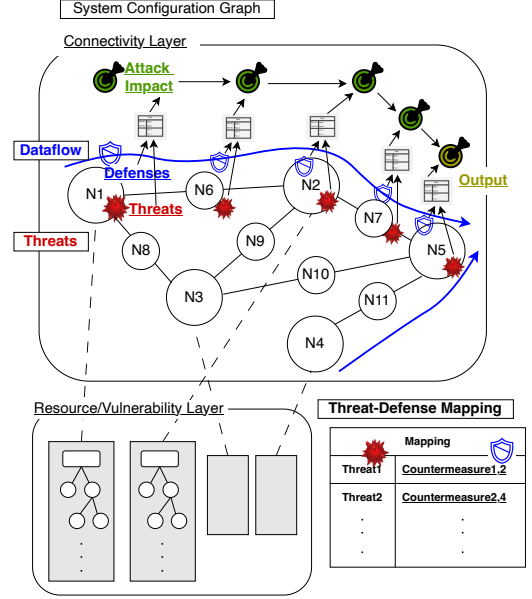


Fig. 1. Overview of the Proposed Framework

to apply existing node-level vulnerability analysis methods such as HARM [7].

In addition, we introduce virtual nodes at the lower layer to represent attacks on communication paths that occur without compromising any nodes, such as eavesdropping, replay, jamming, etc. This representation captures threats originating both inside and outside the system.

Threats and countermeasures are represented as vectors, and their elements correspond to each category. The presence or absence of the threats and countermeasures is represented by a Boolean value. The mapping between a threat and effective countermeasures is defined as a function that associates threats and effective countermeasures. Various works [13], [20]–[23] have categorized threats and countermeasures and presented their mappings. We assume that we use such mappings in practice.

To find the evaluation results, the proposed framework iteratively evaluates the threat faced at each node, the countermeasures applied to data at each node, and threat-countermeasure mapping functions, from the start of the data flow to the end. In this way, the proposed framework outputs the threat left on the data flows, given the vulnerabilities in the system and the countermeasures applied to the data.

A. Formal Model

We denote the model in the proposed framework by R , consisting of the System Configuration Graph G , the Threat-Countermeasure Mapping M , the Set of Data Flows D , and the Set of Threat Vectors A . The following sections describe each component.

$$R = (G, M, D, A)$$

System Configuration Graph: $G = (U, L, f)$

G is the two-layer hierarchical graph. The upper layer U represents the system's nodes and their connectivity, as well as virtual nodes for external attackers on communication paths. The lower layer L represents the resources and vulnerabilities of each node. f is the mapping between nodes in U and L .

The connectivity layer U is a graph composed of the vertex set V_U , which represents the nodes in the system and any potential attackers on each communication link, and the edge set E_U denoting connectivity among the nodes as a pair of vertices in V_U . Each node corresponds to the devices included in the system, such as sensors and routers, or the communication links if the node is virtual.

The resource/vulnerability layer L is a collection of graphs describing the resources and vulnerabilities of each node. Each graph depends on the method used to analyze the resources and vulnerabilities of the nodes, for example, EDG [24] and HARM [7]. We do not specify a particular approach for representing each node's resources and vulnerabilities, allowing the proposed model to use different vulnerability analysis methods.

f is a one-to-one mapping that indicates the correspondence between a node N_i in V_U and an element l_i in L .

Threat-Countermeasure Mapping: $M = (T, S, m)$

M comprises the set of threat categories T , the set of countermeasures categories S , and a mapping m between them. Each threat in T is denoted by t_j , and each countermeasure in S is denoted by s_l . Here, a countermeasure refers to a mechanism that safeguards security requirements such as data confidentiality, integrity, or availability in packets or signals used for communication. The mapping m is a one-to-many, table-based mapping that specifies which countermeasures in S are effective against each threat in T .

An example of threats and countermeasures is shown in Table I, which is based on the mapping from Yoshinaga [8] and the railway safety standard IEC62280 [23] with an extension of adding "Eavesdropping" as a threat to confidentiality by us. In this example, $T = \{(1)\text{Replay}, (2)\text{Dropping}, (3)\text{Injection}, (4)\text{Reordering}, (5)\text{Tampering}, (6)\text{Delay}, (7)\text{Spoofing}, (8)\text{Eavesdropping}\}$. Similarly, $S = \{(1)\text{Sequence Numbering}, (2)\text{Timestamp}, (3)\text{Entity ID}, (4)\text{Feedback}, (5)\text{Device Authentication}, (6)\text{Integrity Code}, (7)\text{Encryption}, (8)\text{Timeout}\}$. The mapping m is defined for each element of T . For instance, $m((1)\text{Replay}) = \{1, 1, 0, 0, 0, 0, 0, 0\}$, meaning "Replay" is prevented by "Sequence Numbering" or "Timestamp." Further details of m is provided in Section III-B.

Each threat and countermeasure is defined by the administrators that evaluates the system. There are several guidelines or frameworks for threat modeling in domain-specific ways or general ways, and the use of these guidelines prevents administrators from missing important threats and countermeasures.

For simplicity, we assume the following for the threats and countermeasures.

TABLE I
EXAMPLE OF THREAT/COUNTERMEASURE CATEGORIES

Threat	Countermeasures
(1) Replay	1) Sequence Numbering, 2) Timestamp
(2) Dropping	1) Sequence Numbering
(3) Injection	1) Sequence Numbering, 3) Entity ID
(4) Reordering	4) Feedback, 5) Device Authentication
(5) Tampering	1) Sequence Numbering, 2) Timestamp
(6) Delay	6) Integrity Code, 7) Encryption
(7) Spoofing	2) Timestamp, 8) Timeout
(8) Eavesdropping	4) Feedback, 5) Device Authentication, 7) Encryption

- Each threat is treated as independent, so combining multiple threats does not alter how they map to countermeasures.
- If at least one countermeasure is applied against a threat, that threat is prevented from being realized.

Even with these simple assumptions, we can still capture foundational yet important scenarios in data flow security. For example, if encryption is applied to a flow, eavesdropping becomes significantly more difficult unless the attacker obtains the encryption key. If sequence numbers are in place, replayed packets can be discarded as long as those numbers remain unmodified.

Set of Data Flows: $D = \{d_i\}$

D is the set of data flows d_i defined on the connectivity layer U . Each data flow d_i is composed of a path p_i and a list c_i of countermeasures applied at nodes on p_i . c_i consists of countermeasure vectors $c_{i,k}$ that describe the status of countermeasures at node N_k . $c_{i,k}$ is a $|S|$ -dimensional vector. If the countermeasure s_l is applied to data flow d_i at node N_k , then $c_{i,k,l} = 1$; otherwise, $c_{i,k,l} = 0$. For instance, in the countermeasure categories shown in Table I, the countermeasure vector $c_{i,k} = \{0, 1, 0, 0, 0, 0, 1, 0\}$ indicates that, at node N_k , (2) Timestamp and (7) Encryption are applied to data flow d_i .

Here, "a countermeasure is applied" means that, for the data under evaluation, the corresponding mechanism is effectively functioning at node N_k . The countermeasures applied may differ between a node within the system and the communication channel. For example, even if data is encrypted when transmitted over the communication path, each node in the system may decrypt it at each hop to verify integrity, which should be treated as not encrypted on the nodes in the system.

Set of Threat Vectors: $A = \{a_k\}$

A represents the threats that arise at each node for the data under evaluation by an attacker. A is composed of threat vectors a_k at each node N_k . Each threat vector a_k is a $|T|$ -dimensional vector defined based on the set of threat categories T . We denote the element of a_k corresponding to threat t_j as $a_{k,j}$. If an attacker can realize a threat of category t_j at node N_k , then $a_{k,j} = 1$; otherwise, $a_{k,j} = 0$. As an example from the threat categories in Table I, the threat vector $a_k = \{0, 0, 0, 0, 1, 0, 1, 0\}$ indicates that threats (5) Tampering and (7) Spoofing exist at node N_k .

Each value of a_i is determined using information on resources and vulnerabilities in the lower layer L , for example, the node compromise status, such as the privileges the attacker has acquired, like file access, arbitrary code execution, etc. For example, if an attacker exploits a vulnerability that allows remote code execution and the vulnerability is not mitigated, there is a risk that all threat classes in Table I could be carried out. Vulnerabilities that cause restart of a node or process may result in data loss or a node's transmission being halted, raising the possibility of threats such as Dropping or Delay.

B. Metric

We define an Attack Impact Vector as a metric for evaluating whether or not a single data flow is at risk for each threat category. Similar to the threat vector A , h is a $|T|$ -dimensional vector whose elements correspond to each threat in T . For any data flow d_i , let h_i be the attack impact vector of d_i . We denote each element of h_i by $h_{i,j}$ for threat t_j , and $h_{i,j} = 1$ if the threat t_j is posed at d_i , otherwise $h_{i,j} = 0$.

Below is the procedure for calculating the Attack Impact Vector h_i .

First, we define a function *isComp* that determines whether a data flow with the countermeasure vector $c_{i,k}$ is compromised under threat t_j . For each threat t_j , we define the vector $m(t_j)$ that lists the countermeasures for t_j using the threat-countermeasure mapping M . $m(t_j)$ is $|S|$ -dimensional. Let the l -th element of $m(t_j)$ be $m(t_j)_l$, $m(t_j)_l = 1$ if s_l is an effective countermeasure for t_j , otherwise $m(t_j)_l = 0$. For instance, in threat category (1)Replay in Table I, $m((1)Replay) = \{1, 1, 0, 0, 0, 0, 0, 0\}$, indicating that the effective countermeasures against (1)Replay are (1)Sequence Numbering or (2)Timestamp.

We define *isComp*($t_j, c_{i,k}, m$) using $m(t_j)$ as follows.

$$\text{isComp}(t_j, c_{i,k}, m) = \begin{cases} 0 & \exists s_l \in S, m(t_j)_l = 1 \ \& \ c_{i,k,l} = 1 \\ 1 & \text{else} \end{cases}$$

This means that, for a threat of category t_j , if any of the corresponding countermeasures in the mapping m is applied to data at node N_k , the data flow d_i will not be affected. This definition is based on the attack assumptions discussed in Section III-A. Conversely, if none of the corresponding countermeasures are applied, the data flow will be affected in the threat t_j .

In this study, we assume that once an attack impact has occurred in a data flow, it cannot be fully recovered. For example, from a confidentiality standpoint, if eavesdropping or key leakage occurs, we cannot ensure that the leaked information is fully removed. As for availability, if data is lost or transmitted with a long delay, it is typically impossible to fully compensate for the lost data or the failure to meet timing requirements. In practice, partial mitigation methods are available, such as removing suspicious data via anomaly detection at the application level or in network traffic. However, this study evaluates threats at the entire data flow level, and these mitigation strategies are beyond our scope.

We therefore iteratively evaluate whether a particular threat has been realized on d_i .

Below is the procedure for calculating the attack impact vector h_i for data flow d_i using *isComp*.

- 1) Initialize all elements of the attack impact vector h to zero.
- 2) Extract the path $p_i = \{N_{i,1}, N_{i,2}, \dots, N_{i, \text{length}(p_i)}\}$ from the data flow $d_i = (p_i, c_i)$.
- 3) For each node $N_{i,k}$ on the path p_i , perform the following steps in order:
 - a) Check the Threat Vector of $N_{i,k}$, $a_k = \{a_{k,1}, a_{k,2}, \dots, a_{k,|T|}\}$.
 - b) For each threat category $t_j \in T$ in a_k , if $a_{k,j} = 1$ and $h_{i,j} = 0$, then $h_{i,j} \leftarrow \text{isComp}(t_j, c_{i,k}, m)$
- 4) Evaluation on all nodes is complete. Use the attack impact vector h as the threats not prevented by the system.

IV. EVALUATION

We evaluate the effectiveness of our proposed framework by illustrating the process of assessing attack impact in multiple threat scenarios for a simple IoT system. First, we describe the system's design and configuration. Next, we present the threat scenarios. Then, we show how to apply the proposed framework to evaluate the attack impact under each scenario in the target system. Finally, we review the evaluation process to demonstrate the validity and usefulness of the proposed framework. Assuming that the data of particular importance is the environmental measurements transmitted from sensors to the gateway. We focus on the field storing measurement data within the protocol used when sending data from sensors to the gateway.

A. System Configuration

Figure 2 presents an overview of the assumed system. This system is for wide-area environmental measurements using two sensor networks, ZigBee and LoRa. Each network sends its collected data to a cloud-based monitoring application through a dedicated gateway. This system flexibly supports wide-area measurement by employing two communication protocols with differing coverage distances. In this evaluation, we focus on the ZigBee and LoRa networks.

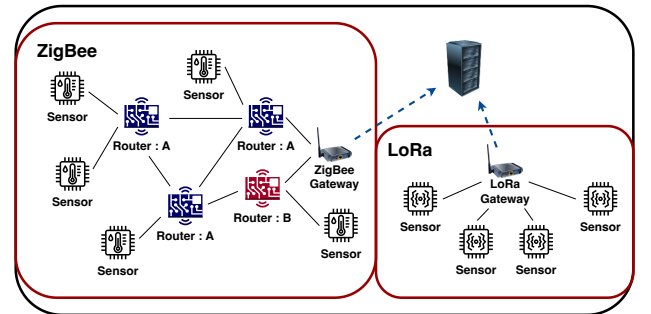


Fig. 2. System architecture of the assumed system

1) *Zigbee Network*: Sensors act as end devices and transmit measurement data to the ZigBee Gateway through a router. In this mesh network, the gateway acts as a coordinator and aggregates measurement data from all sensors.

The ZigBee protocol stack uses IEEE 802.15.4 for the physical and MAC layers and defines a ZigBee-specific network layer, application support sublayer (APS), and application layer. Sensor data is stored in the packet payload at the application layer.

The application-layer payload is protected by countermeasures at the MAC, network, and APS layers. Although both the APS and network layers offer optional security features, only the network-layer option is enabled.

The ZigBee network includes two types of routers: Router-A and Router-B. These two routers differ in their devices and OS. Router-A is not known to have any exploitable vulnerabilities. In contrast, Router-B contains multiple known vulnerabilities in its control modules and OS, which can be exploited to gain unauthorized administrative access and tamper with firmware.

2) *LoRa Network*: Multiple sensors transmit data directly to a gateway, forming a star topology.

LoRa specifies the physical and MAC layers [25]. Data formats and upper-layer functions often rely on extended protocols such as LoRaWAN, or on user-specific implementations. We assume a vulnerable user-specific implementation for LoRa's upper layer. This implementation transmits a device ID along with sensor data, but the gateway does not verify it.

All sensors are identical devices sharing common hardware/software components, such as a wireless module and a control module.

B. Threat Scenarios

1) *Threats*: We consider three threat scenarios with different levels of attack capability: eavesdropping, replay attacks, and tampering. All three are typical attacks and vary in the attacker's level of infiltration into the system and whether the attack is passive or active.

Attack 1: Eavesdropping The attacker's capability is relatively limited, and the attack is passive from outside the system. The attacker attempts to eavesdrop on any ZigBee or LoRa wireless link using a radio receiver and does not know confidential parameters such as encryption keys.

Attack 2: Replay Attack The attacker does not have access to nodes within the system but performs an active attack from outside. The attacker captures traffic on any ZigBee or LoRa wireless link and replays it using their transmitter. By repeatedly sending legitimate data, they aim to inject unauthorized data into the system.

Attack 3: Tampering The attacker gains access to a node within the system and carries out an active attack. The attacker exploits a known vulnerability in a specific OS on Router-B, obtains administrator privileges, and modifies the firmware on Router-B. Then, the attacker attempts to modify the sensor data by running compromised firmware.

2) *Countermeasures*: We analyze the security features of the target system by focusing on the protocols employed in each network. We examine each layer of the protocol stack and describe the security functions and characteristics associated with it.

ZigBee Features at the MAC layer include assigning a sequence number, designating source/destination IDs, and incorporating an error-correcting code. These mechanisms provide reliable packet transmission between adjacent nodes.

The network-layer header includes source/destination IDs indicated by an 8-byte IEEE address or a 2-byte short address, and a 1-byte sequence number. With the network layer security option enabled, the AUX header is added to the network-layer payload. This header provides encryption, message authentication, an integrity code, and an additional sequence counter. All encryption relies on AES-128. The AUX header attaches a 4-byte integrity code. Encryption is performed hop by hop [26] using a network key shared by all nodes. Accordingly, the network-layer payload is decrypted and undergoes integrity verification at each hop. Therefore, the network-layer payload is handled in plaintext during verification at each node.

At the APS layer, the header includes a numeric endpoint identifier used to distinguish applications, as well as a 1-byte sequence number.

Additionally, device authentication is achieved by controlling network participation. A new node can send a join request only while the coordinator is in the Permit Join state, and the network key is shared only with nodes whose participation is approved.

LoRa In the LoRa network of this system, the firmware only handles data transmission and reception, as well as identification of each terminal. The transmitted data consists solely of sensor data and a sensor device ID. At the MAC layer, a cyclic redundancy check (CRC) is implemented for error correction [25]. However, it lacks additional features such as encryption or authentication, leaving it vulnerable to various attacks.

3) *Attack Impacts*: Below, we review the impacts of the attacks.

Attack 1: Eavesdropping In ZigBee, AES-128 encryption is employed at the network layer, so it is unlikely that an attacker could decrypt AES-128-encrypted packets in a practical time, without knowing the encryption key.

In LoRa, the attacker can obtain measurement data in plaintext from the received signals.

Attack 2: Replay Attack In ZigBee, the MAC and network layers each include a 1-byte sequence number in their headers, and the network-layer AUX header has a 4-bit sequence number. Thus, a replay attack without altering transmission timing or sequence numbers is highly unlikely to succeed, because the packet sequence number has a sufficiently large address space. Packets whose sequence numbers overlap or fall outside the reception window are discarded.

In LoRa, there are no effective countermeasures in any layer of the protocol stack, so a replay attack succeeds.

Attack 3: Tampering In Zigbee, only network-layer encryption is enabled, which decrypts packets hop by hop, so the attacker can arbitrarily modify the data in the packet on the node's firmware. The altered packet includes an integrity code based on the malicious data and is not detected by integrity checks.

In LoRa, there are no vulnerable nodes for attackers to exploit.

C. Evaluation of Data Flows with the Proposed Framework

Figure 3 shows a conceptual diagram of how the system is represented under the proposed model.

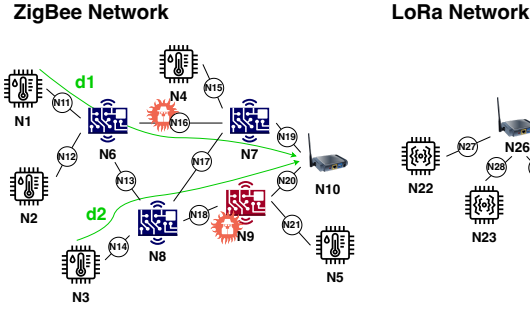


Fig. 3. Representation of the Assumed System in the Proposed Framework

First, we define the sensors, routers, and gateways that compose the system as nodes in the model. These correspond to N_1 through N_{10} and N_{22} through N_{26} in Figure 3. Next, for each communication link between nodes, a potential external attacker is inserted as a virtual node. Hereafter, we simply refer to these as virtual nodes. These correspond to N_{11} through N_{21} and N_{27} through N_{30} .

For simplicity, we omit the lower layer representation of each node.

We employ the Threat-Countermeasure Mapping M shown in Table I. The example in Table I is based on a mapping table created by Yoshinaga [8], drawing on the IEC62280 [23] safety standard for the railway sector.

We evaluated the attack impact on the four data flows depicted as arrows in Figure 3. The Set of Data Flows D represents these flows, namely $D = \{d_1, d_2, d_3, d_4\}$. Table II shows the node sequence for each data flow's path in the model. Table III shows the countermeasures applied to each data flow according to the type of nodes.

TABLE II
PATHS OF EACH DATA FLOW

D	$p_i (i=1,2,3,4)$
d_1	$\{N_1, N_{11}, N_6, N_{16}, N_7, N_{19}, N_{10}\}$
d_2	$\{N_3, N_{14}, N_8, N_{18}, N_9, N_{20}, N_{10}\}$
d_3	$\{N_{24}, N_{29}, N_{26}\}$
d_4	$\{N_{25}, N_{30}, N_{26}\}$

To determine the countermeasure vectors, we review the protocol-specific security features described in Section IV-B2.

TABLE III
COUNTERMEASURES OF EACH DATA FLOW AT EACH NODE

Zigbee Countermeasures	
Device (Nodes)	$c_{i,k} (i=1,2)$
Sensor (N_1, N_2, N_3, N_4, N_5)	$\{1, 0, 1, 0, 1, 0, 0, 0\}$
Router:A (N_6, N_7, N_8)	$\{1, 0, 1, 0, 1, 0, 0, 0\}$
Router:B (N_9)	$\{1, 0, 1, 0, 1, 0, 0, 0\}$
Gateway (N_{10})	$\{1, 0, 1, 0, 1, 0, 0, 0\}$
Virtual Nodes ($N_{11}-N_{21}$)	$\{1, 0, 1, 0, 1, 1, 1, 0\}$
LoRa Countermeasures	
Device (Nodes)	$c_{i,k} (i=3)$
Sensor ($N_{22}, N_{23}, N_{24}, N_{25}$)	$\{0, 0, 1, 0, 0, 1, 0, 0\}$
Gateway (N_{26})	$\{0, 0, 1, 0, 0, 1, 0, 0\}$
Virtual Nodes ($N_{27}, N_{28}, N_{29}, N_{30}$)	$\{0, 0, 1, 0, 0, 1, 0, 0\}$

First, we discuss the ZigBee network. The countermeasures of virtual nodes in the ZigBee network correspond to those carried out along the ZigBee communication links, which are summarized as follows.

- 1) **Sequence numbering**: Applied(1): Included in the MAC layer, network layer, APS layer, and the network layer's AUX header. Verified at each hop.
- 2) **Timestamp**: Not applied(0): Not present in any layer.
- 3) **Entity ID**: Applied(1): Included in the MAC layer's IEEE address or short address, the network layer's IEEE address or short address, and the APS layer's endpoint number. Referenced upon arrival at the next node in the system.
- 4) **Feedback**: Not applied(0): No mechanism to query the sending device about received data.
- 5) **Device Authentication**: Applied(1): New node participation is controlled by the coordinator, preventing unauthorized devices from joining.
- 6) **Integrity Code**: Applied(1): Included in the MAC layer and the AUX header of the network layer. Verified upon arrival at the next node in the system.
- 7) **Encryption**: Applied(1): The network-layer payload is encrypted, making decryption impossible without the encryption key.
- 8) **Timeout**: Not applied(0): In application layer data transfer, there is no mechanism for the receiving side to wait for transmitted data flows or time out.

As a result, the countermeasure vector at each virtual node is $a = \{1, 0, 1, 0, 1, 1, 1, 0\}$. In contrast, encryption and integrity codes are evaluated as not applied (0) in the actual nodes. This is because the network layer payload is decrypted at each hop, making the payload plaintext and thus unprotected. Hence, for those nodes, the countermeasure vector is $a = \{1, 0, 1, 0, 1, 0, 0, 0\}$.

For the LoRa network, the transmitted data consists only of the measurement value and an ID for device identification, and the MAC layer provides an error-correcting code. Consequently, we determined that only Entity ID and Integrity Code are applied, resulting in $a = \{0, 0, 1, 0, 0, 1, 0, 0\}$.

Next, each threat scenario was represented as the value of the threat vector for each node. Table IV shows the threat vector for each node in each attack scenario. Nodes not listed

are assumed to have a threat vector of all zeros. For Attack 1, we assign the eavesdropping threat category to all virtual nodes. This is based on the assumption that the attacker can attempt eavesdropping on any ZigBee or LoRa wireless link. Similarly, for Attack 2, every virtual node is vulnerable to the replay attack. In contrast, Attack 3 aims solely at Router B in the ZigBee network. Therefore, only node N_9 has this threat vector.

TABLE IV
THREAT VECTORS OF EACH ATTACK

Attack	a_k	Attack nodes (k)
1: Eavesdropping	$\{0, 0, 0, 0, 0, 0, 0, 1\}$	$\{N_{11} - N_{21}, N_{27} - N_{30}\}$
2: Replay Attack	$\{1, 0, 0, 0, 0, 0, 0, 0\}$	$\{N_{11} - N_{21}, N_{27} - N_{30}\}$
3: Tampering	$\{0, 0, 0, 0, 1, 0, 0, 0\}$	$\{N_9\}$

Finally, we address the attack impact evaluation for each data flow. For each attack, we used the threat vectors and countermeasure vectors described above. Table V shows the calculated values for d_1 through d_4 under each attack. We followed the calculation procedure presented in Section III-B to compute the attack impact vector.

TABLE V
ATTACK IMPACT OF EACH ATTACK

H	Attack 1	Attack 2	Attack 3
h1	$\{0, 0, 0, 0, 0, 0, 0, 0\}$	$\{0, 0, 0, 0, 0, 0, 0, 0\}$	$\{0, 0, 0, 0, 0, 0, 0, 0\}$
h2	$\{0, 0, 0, 0, 0, 0, 0, 0\}$	$\{0, 0, 0, 0, 0, 0, 0, 0\}$	$\{0, 0, 0, 1, 0, 0, 0, 0\}$
h3	$\{0, 0, 0, 0, 0, 0, 0, 1\}$	$\{1, 0, 0, 0, 0, 0, 0, 0\}$	$\{0, 0, 0, 0, 0, 0, 0, 0\}$
h4	$\{0, 0, 0, 0, 0, 0, 0, 1\}$	$\{1, 0, 0, 0, 0, 0, 0, 0\}$	$\{0, 0, 0, 0, 0, 0, 0, 0\}$

D. Validity of the Proposed Framework

In each attack, the attack impact vector is zero for data flows that are not targeted or have success probabilities that are negligible under the attacker's constraints. For example, the values for the threat "Eavesdropping" are zero for d_1 and d_2 , since encryption protects these flows. The values for threat "Replay" are zero for d_1 and d_2 , reflecting the low practical probability of success. The values for the threat "Tampering" are zero for d_1 , d_3 , and d_4 , as they lie outside the attack's scope. On the other hand, the evaluation results are 1 for those threat categories that can cause a substantial effect on the data flow's security requirements. For instance, the values for the threat "Eavesdropping" are 1 for d_3 and d_4 due to unencrypted communications. The values for threat "Replay" are 1 for d_3 and d_4 because the system accepts unauthorized data. The value for the threat "Tampering" is 1 for d_2 , where measurement values can be arbitrarily altered.

These results show that we succeeded in evaluating the attack impacts as "compromised" if the threat could severely affect the security requirements. Conversely, the attack impacts are evaluated to be "no compromise" when an attack is very difficult to realize in practice under the attacker's constraints or exerts no impact.

V. DISCUSSION

The proposed framework can be used for other protocols than our evaluation. By analyzing packet structures, the user can identify the countermeasures applied to the data flow.

The proposed model can be used with existing threat-countermeasure mappings. There are several domain-specific guidelines like IEC62280 [23], and we can use these guidelines if the domain of the system matches the guideline. If not, we can use these guidelines for reference, and they can be extended to a mapping that covers confidentiality, integrity, and availability. More general resources such as MITRE ATT&CK [20] or ENISA guidelines [21] may also be incorporated, covering diverse threats across multiple attack stages.

By representing each defense mechanism as a binary state, the framework easily accommodates dynamic changes in device resources or security requirements. These single-bit updates directly reflect how countermeasures interact with active threats along each data flow. However, there are limitations in the expressibility of the threat-countermeasure mapping. For example, the framework is designed to represent the presence or absence of countermeasures, but it does not account for the strength or effectiveness of these countermeasures.

Finally, the formal model and defined metrics facilitate computational exploration. They contribute to automatically finding optimal countermeasure placement or identifying unmitigated threats, leveraging threat and countermeasure vectors as key variables.

VI. CONCLUSION

We propose a framework to evaluate the attack impact on data flows in systems with heterogeneous resources by representing the threats existing in each node and the countermeasures applied to data flows in their respective categories. The proposed framework evaluates the presence or absence of compromise for each threat category. It comprises a formal model that represents the system's data flows and the threats from both inside and outside the system, and a metric for evaluating attack impact on that model.

We confirmed the validity of the proposed framework by applying it to a simple IoT sensor system. To the threats of eavesdropping, replay attacks, and tampering, the proposed framework evaluates data flows that are realistically impacted as "compromised," while treating others as "not compromised."

The proposed approach offers several potential advantages. The proposed approach can work well for computational exploration and automation, allows flexible expansion of threat and countermeasure categories by incorporating references like MITRE ATT&CK, and employs a single-bit representation of countermeasure presence to simplify dynamic defense modeling for real-time assessment.

Future works are expected to apply our method to real-world systems, extend the framework to capture temporal or quantitative information about each attack, and investigate the increase in computational complexity in systems with a larger number of nodes and data flows.

ACKNOWLEDGMENT

This work was supported in part by JST K Program JP-MJKP24K3.

We used ChatGPT to support English translation and to help summarize longer texts to meet page limitations. Grammarly was used for additional grammar checks. The authors performed final reviews and edits.

REFERENCES

- [1] J. M. Voas, "Networks of 'Things'," National Institute of Standards and Technology (NIST), NIST Special Publication 800-183, Jul. 2016.
- [2] T. Qiu, N. Chen, K. Li, M. Atiquzzaman, and W. Zhao, "How can heterogeneous internet of things build our future: A survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2011–2027, 2018.
- [3] S. S. Mahadik, P. M. Pawar, and R. Muthalagu, "Heterogeneous IoT (HetIoT) security: techniques, challenges and open issues," *Multimed. Tools Appl.*, vol. 83, no. 12, pp. 35 371–35 412, Apr. 2024.
- [4] Q. Zhang, A. Z. Mohammed, Z. Wan, J.-H. Cho, and T. J. Moore, "Diversity-by-design for dependable and secure cyber-physical systems: A survey," *IEEE Trans. Netw. Serv. Manage.*, vol. 19, no. 1, pp. 706–728, Mar. 2022.
- [5] M. Zhang, L. Wang, S. Jajodia, A. Singhal, and M. Albanese, "Network diversity: A security metric for evaluating the resilience of networks against zero-day attacks," *IEEE Trans. Inf. Forensics Secur.*, vol. 11, no. 5, pp. 1071–1086, May 2016.
- [6] Y. Prieto, M. Figueroa, and J. E. Pezoa, "Maximizing network reliability to 0-day exploits through a heterogeneous node migration strategy," *IEEE Access*, vol. 9, pp. 97 747–97 759, 2021.
- [7] J. B. Hong and D. S. Kim, "Assessing the effectiveness of moving target defenses using security models," *IEEE Trans. Dependable Secure Comput.*, vol. 13, no. 2, pp. 163–177, 2016.
- [8] J. Yoshinaga, "Proposal of cyberattack resistance evaluation method by cpn for data-flow of railway signalling system using wireless-device in use," *IPSJ Journal*, vol. 65, no. 5, pp. 954–967, 2024.
- [9] M. Touhiduzzaman, A. Hahn, and A. K. Srivastava, "A diversity-based substation cyber defense strategy utilizing coloring games," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 5405–5415, Sep. 2019.
- [10] L. Wang, S. Jajodia, A. Singhal, P. Cheng, and S. Noel, "k-zero day safety: A network security metric for measuring the risk of unknown vulnerabilities," *IEEE Trans. Dependable Secure Comput.*, vol. 11, no. 1, pp. 30–44, 2014.
- [11] O. Duman, M. Zhang, L. Wang, M. Debbabi, R. F. Atallah, and B. Lebel, "Factor of security (FoS): Quantifying the security effectiveness of redundant smart grid subsystems," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 1018–1035, 2022.
- [12] G. George and S. M. Thampi, "A graph-based security framework for securing industrial IoT networks from vulnerability exploitations," *IEEE Access*, vol. 6, pp. 43 586–43 601, 2018.
- [13] M. J. C. Izar Tarandach, *Threat Modeling: A Practical Guide for Development Teams*. O'Reilly Media, 2020.
- [14] T. UcedaVelez, "Threat modeling w/ pasta: Risk centric threat modeling case studies," in *OWASP AppSecEU 2017*, May 2017.
- [15] —, "Real world threat modeling using the pasta methodology," in *OWASP AppSec EU 2012*, 2012.
- [16] A. Wolf, D. Simopoulos, L. D'Avino, and P. Schwaiger, "The pasta threat model implementation in the iot development life cycle," *INFOR-MATIK 2020*, pp. 1195–1204, 2021.
- [17] H. Hu, J. Wu, Z. Wang, and G. Cheng, "Mimic defense: a designed-in cybersecurity defense framework," *IET Inf. Secur.*, vol. 12, no. 3, pp. 226–237, May 2018.
- [18] L. Wang, Z. Zhang, W. Li, Z. Liu, and H. Liu, "An attack surface metric suitable for heterogeneous redundant system with the voting mechanism," *J. Phys. Conf. Ser.*, vol. 1168, no. 3, p. 032079, Feb. 2019.
- [19] N. Shevchenko, T. Chick, P. O'Riordan, T. P. Scanlon, and C. Woody, "Threat modeling: A summary of available methods," Jul. 2018.
- [20] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, and others, "MITRE ATT&CK: Design and philosophy," *Technical report*, 2018.
- [21] European Union Agency for Cybersecurity (ENISA), "Good practices for security of internet of things in the context of smart manufacturing," Nov. 2018, [accessed 18-January-2025].
- [22] MITRE, "ICS ATT&CK Matrix," <https://attack.mitre.org/matrices/ics/>, [Online; accessed 26-January-2025].
- [23] International Electrotechnical Commission (IEC), "IEC 62280: Railway applications – Communication, signalling and processing systems – Safety-related communication in transmission systems," 2014, IEC 62280:2014(E).
- [24] A. Longueira-Romero, R. Iglesias, J. L. Flores, and I. n. Garitano, "A novel model for vulnerability analysis through enhanced directed graphs and quantitative metrics," *Sensors*, vol. 22, no. 6, Mar. 2022.
- [25] A. Augustin, J. Yi, T. Clausen, and W. M. Townsley, "A study of LoRa: Long range & low power networks for the internet of things," *Sensors (Basel)*, vol. 16, no. 9, p. 1466, Sep. 2016.
- [26] S. Labs, "An1233: Zigbee security," <https://www.scribd.com/document/699037032/an1233-zigbee-security>, 2023, accessed: 2025-01-25.