Journal of
Cybersecurity and Privacy

CITESCORE
5.3

Article

Enhancing Cybersecurity through Comprehensive Investigation of Data Flow-Based Attack Scenarios

Sara Abbaspour Asadollah, Shamoona Imtiaz, Alireza Dehlaghi-Ghadim, Mikael Sjödin and Marjan Sirjani

MDPI

https://doi.org/10.3390/jcp4040039

Journal of
Cybersecurity
and Privacy

Article

Enhancing Cybersecurity Through Comprehensive Investigation of Data Flow-Based Attack Scenarios

Sara Abbaspour Asadollah [1,*], Shamoona Imtiaz [1], Alireza Dehlaghi-Ghadim [2], Mikael Sjödin [1] and Marjan Sirjani [1]

[1] School of Innovation, Design and Engineering, Mälardalen University, 721 23 Västerås, Sweden; shamoona.imtiaz@mdu.se (S.I.); mikael.sjodin@mdu.se (M.S.); marjan.sirjani@mdu.se (M.S.)
[2] RISE Research Institute of Sweden, 722 12 Västerås, Sweden; alireza.dehlaghi.ghadim@ri.se
* Correspondence: sara.abbaspour@mdu.se

Abstract: Integration of the Internet of Things (IoT) in industrial settings necessitates robust cybersecurity measures to mitigate risks such as data leakage, vulnerability exploitation, and compromised information flows. Recent cyberattacks on critical industrial systems have highlighted the lack of threat analysis in software development processes. While existing threat modeling frameworks such as STRIDE enumerate potential security threats, they often lack detailed mapping of the sequences of threats that adversaries might exploit to apply cyberattacks. Our study proposes an enhanced approach to systematic threat modeling and data flow-based attack scenario analysis for integrating cybersecurity measures early in the development lifecycle. We enhance the STRIDE framework by extending it to include attack scenarios as sequences of threats exploited by adversaries. This extension allows us to illustrate various attack scenarios and demonstrate how these insights can aid system designers in strengthening their defenses. Our methodology prioritizes vulnerabilities based on their recurrence across various attack scenarios, offering actionable insights for enhancing system security. A case study in the automotive industry illustrates the practical application of our proposed methodology, demonstrating significant improvements in system security through proactive threat modeling and analysis of attack impacts. The results of our study provide actionable insights to improve system design and mitigate vulnerabilities.

Keywords: cybersecurity; cyberattack; attack scenario; threat modeling; STRIDE; attack impact analysis; cyber–physical system (CPS)

check for updates

Citation: Abbaspour Asadollah, S.; Imtiaz, S.; Dehlaghi-Ghadim, A.; Sjödin, M.; Sirjani, M. Enhancing Cybersecurity Through Comprehensive Investigation of Data Flow-Based Attack Scenarios. *J. Cybersecur. Priv.* **2024**, *4*, 823–852. https://doi.org/10.3390/jcp4040039

Received: 23 July 2024
Revised: 14 September 2024
Accepted: 25 September 2024
Published: 4 Octpber 2024

1. Introduction

Digital solutions are designed to ensure that desired information flows between different components of the system, as in the case of construction equipment management systems. These systems use various technologies, such as GPS, IoT, and wireless communication, to achieve the required functionality. However, due to continuous integration and essential information exchange, the risk of data leakage, vulnerability exploitation, and compromised information flows is continuously growing. Any weaknesses, when data either sit at rest or travel between different nodes, can provide passage to malicious actors. Interception and manipulation of data during transmission are not just theoretical concerns but are actively exploited by cybercriminals [1].

One way to counter these issues is the use of the STRIDE framework and the Microsoft Threat Modeling tool. This approach is widely used to enumerate potential security threats in systems by categorizing threats into six types, namely, Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege. Nevertheless, it does not provide a detailed mapping of attack paths, which are sequences of actions an adversary may take to exploit the threats reported by the Microsoft Threat Modeling tool.

One of the most important challenges today arises from the growing threat landscape of cybersecurity, primarily due to growing system complexity induced by integrating IoT technologies into industry [2]. These new vulnerabilities and attack paths make it increasingly necessary to develop comprehensive approaches for threat modeling. Although STRIDE serves as a valuable foundation for enumerating threats, it often lacks detailed mapping of the sequences of threats that could be exploited by attackers to execute cyber-attacks. This limitation can leave critical systems vulnerable to sophisticated multi-step attacks that exploit overlooked pathways [3]. On the other hand, the fact that STRIDE is based on individual components rather than looking at the holistic system interactions may cause incomplete threat assessments, particularly in complex IoT systems [4]. In addition, existing methods often do not take a systematic approach to quantifying potential impacts of identified threats, which may imply that security practitioners are often left with prioritization challenges related to potential impacts of identified threats [5].

This being the case, there is huge responsibility on system integrators to ensure the reliability and security of their IT infrastructure with respect to functionality, performance, vulnerabilities, threats, and possible attack surfaces. Lately, this challenge has been handled by physically securing digital assets; however this is no longer sufficient, as the triplet of humans, technology, and organizations has become too complex to restrict various unwanted activities. For instance, people who write programs do not consider a global view of the environment in which the application will be integrated. The interactions an application will make can introduce potential security gaps. In this scenario, extending STRIDE to map threats into attack paths and scenarios will allow designers, developers, system integrators, and security professionals to interrogate whether or not the things around them are eminent solutions. An attack scenario refers to a sequence of attack paths that an attacker might follow to exploit vulnerabilities and compromise a system.

Even perfect functionality may still lead to catastrophic events if proper threat modeling and risk analysis are not performed before deployment. In such situations, understanding various attack paths draws immediate attention to the most critical areas at the system architecture level, along with vulnerable entry points and weaknesses of existing defense lines. Sometimes weakness is a result of a very unlikely placement, while sometimes it is obvious from its origin and location, exposure to and from the world, and accessibility options. A very efficient function can suffer from security holes that might not have been imagined or considered during the security development life cycle. There are several ways to misalign the designed internal functionality. One approach is to exploit the data flow between different actors, as even a foolproof system design can still leak information about the system. Depending on the transmission channels, data encapsulation method, data protection technique, and data communication protocol, it is possible for data in transition to be more exposed to security challenges compared to the data at rest.

An important domain of cybersecurity research is attack scenario analysis, particularly in IoT systems and industrial environments [6,7]. This approach is represented by the mapping of potential attacker action sequences to provide a comprehensive view of system vulnerabilities. Recent studies have suggested that especially complex and highly interconnected environments can be secured with security measures designed upon a basic understanding of these attack paths [8]. While improvements are seen in threat modeling and the analysis of attack scenarios, a considerable gap exists with respect to the integration of these approaches with data flow analysis in IoT and industrial systems. To the best of our understanding, existing methodologies fail to capture the dynamic nature of data flows in these environment domains, leading to incomplete threat assessments. Our research aims to address this gap by developing a comprehensive methodology that combines STRIDE-based threat modeling with detailed data flow analysis and attack scenario mapping.

Therefore, we present a solution that aims to confront adversarial behavior against the flow of data, i.e., to identify the impact of threats and the depth of attacks that exist in a system design based on the information flow. The proposed solution is augmented with a verified actor-based system design and a threat list to discover attack scenarios automatically. At first, we systematically analyze the threat landscape using our customized threat enumeration technique for each entity in the data flow diagram, explained in more detail in Section 3.2.2. In general, instead of naive reasoning, we aim to quantify the probability of actions, i.e., from threat list to attack scenarios. We describe these contributions in further detail with our threat intelligence process. This study is motivated by the fact that interoperability increases the attack surface, while early attack impact analysis reduces attack probability and impact. In view of the identified limitations of current cybersecurity practices for IoT and industrial systems, the key problems to be addressed by this research are: (i) lack of full-fledged mapping of multistep attack scenarios in complex interlinked systems; (ii) inadequate integration of data flow analysis with threat modeling techniques; and (iii) lack of systematic methods for quantifying and prioritizing the impacts of identified threats. Within this context, the specific research goals are as follows: (a) to enhance the STRIDE framework by detailing attack scenarios in IoT and industrial systems; (b) to develop a methodology describing how data flow analysis can be introduced into the threat modeling process to provide an extended security assessment; (c) to develop a systematic approach for identifying and prioritizing threats according to their impact; and (d) to prove the applicability of the proposed procedure by implementing a case study in the automotive sector. It is important to note that the contributions made in this study relate to security by design, seeking to aid system designers and integrators when designing or updating system components. Because we focus on data leakage when data are in flow, the design time data flow diagram is the main focus of simulating security posture. The main contributions of this paper are outlined as follows:

- **Impact Enumeration:** We explore the various ways that attackers can manipulate systems, providing valuable insights that can enhance system security. These insights serves as important inputs for security and system designers who must make crucial decisions during the design phase and prior to the implementation phase. We then evaluate attack scenarios and reveal underlying attack paths that enable early prediction of attacks during the system development life cycle.
- **Attack Profile Generation:** Using the Microsoft Threat Modeling tool (STRIDE, TMT7), we generate a threat list from the data flow diagram of the system design. We then generate the attack profile in correspondence with the threat landscape of the system. The resulting profile enumerates the possible attacks with all possible sources and destinations. This profile can be used as a baseline for further security analyses, such as the threat impact of and depth of attack.
  In addition, we detect attack surfaces, provide an API (Application Programming Interface) for creating attack scenarios, and present an extensive collection of attack scenarios based on attackers' behavior. The generated attack profiles assist system designers in proactively anticipating potential attacks and enhancing overall system security.
- **Attack Impact Analysis:** We conduct an in-depth analysis to understand how system analysts and designers can use these attack scenarios to secure their systems during the design phase. System designers must make critical decisions at this stage prior to implementation, and as a result can greatly benefit from this analysis. By mapping all potential attack paths, we evaluate the impact on strengthening the system against cyberattacks that can be obtained by addressing each vulnerability. This approach offers system designers a prioritized list of vulnerabilities to address based on their contributions to various attack paths. While quantitative comparisons with existing methods are beyond the scope of this study, our approach offers several qualitative advantages. These include a more comprehensive analysis of attack scenarios through systematic examination of data flows, a structured and repeatable threat identification process, and improved prioritization of security efforts based on detailed impact analysis. Future work will focus on quantifying these benefits through empirical studies and providing statistical evidence to complement the qualitative benefits presented in this paper.

The remainder of this paper is organized as follows. In Section 2, we present an overview of the techniques and terminology employed in this paper. Next, we outline our research methodology and discuss the contributions of this paper in Section 3. In Section 4, we illustrate our proposed approach step-by-step through a case study. Section 5 illustrates the results of the case study, while Section 6 discusses the results obtained in our experiments. The related work in the field is surveyed in Section 7. Finally, we conclude the study and outline directions for future research in Section 8.

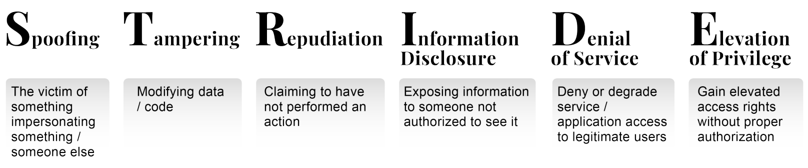
2. Background

The following subsections are dedicated to reviewing the background and terminology employed in this paper.

2.1. Threat Modeling

By definition, a threat is a potential harm that may result from a weakness present in the development of software applications [9]. When this weakness is exploited, a threat forms and an attack happens. Identifying possible threats during the design phase of software development using threat modeling can enhance system security [9], helping to recognize potential threats and different mitigation techniques. In this way, it is possible to identify critical design elements that need to be protected. Among the various security threat classification tools and methods, STRIDE is widely used for threat modeling and provides the industry-standard approach for identifying threat scenarios [10,11].

STRIDE is a framework for modeling threats developed by Microsoft (Redmond, WA, USA) [12] in 1999. The method provides threat identification in the design phase of any software or hardware, and as such provides understanding of possible attack scenarios. To apply the STRIDE framework, security experts first need to create Data Flow Diagrams (DFD) of the system, which show the communication patterns among the elements. Then, the method examines these diagrams to identify potential threats to the system. Based on the STRIDE model, threats are divided into six different categories, as shown in Figure 1: *Spoofing* , *Tampering*, *Repudiation*, *Information Disclosure*, *Denial of Service*, and *Elevation of Privileges* [13]. DFD diagrams can be investigated manually (brainstorming) or by using the Microsoft Threat Modeling tool. The methodology conducted in this paper applies the manual approach.



Spoofing
The victim of something impersonating something / someone else
Tampering
Modifying data / code
Repudiation
Claiming to have not performed an action
Information Disclosure
Exposing information to someone not authorized to see it
Denial of Service
Deny or degrade service / application access to legitimate users
Elevation of Privilege
Gain elevated access rights without proper authorization

Figure 1. STRIDE acronym legend [13].

2.2. Attack Surfaces and Attack Scenarios

In cybersecurity, the attack surface commonly refers to the total number of points, interfaces, connections, and interactions with environments through which an attacker can potentially exploit vulnerabilities, launch malicious actions, or gain restricted data in the system. An attack scenario typically refers to paths by which cybercriminals can gain unauthorized access to a system or network.

An attack scenario refers to a sequence of attack paths that an attacker might follow to exploit vulnerabilities and compromise a system. Attack scenarios are essential for cybersecurity professionals, as they are crucial for designing adequate protections and play a vital role in overall cybersecurity strategy. The landscape of attack scenarios in cybersecurity is constantly evolving, with new scenarios emerging as technology advances and cybercriminals become more sophisticated. Cybercriminals may use a single attack path or a combination of paths to compromise a system or network. The difficulty of identifying a specific attack scenario depends on various factors, including the complexity of the system or network, the level of security measures in place, and the attacker's skills and resources. In certain cases the path may be relatively easy to find, such as when a system or network has known vulnerabilities that have not been sufficiently remediated. In other cases the path may be more challenging to identify, such as when an attacker uses advanced techniques or a combination of paths to avoid detection and bypass security measures. Finding attack scenarios is essential for cybersecurity professionals to develop appropriate security measures.

A common example of an attack scenario in a trusted smart system is unauthorized access by an attacker who manipulates the functionality of critical components. For instance, in a smart home system an attacker could potentially spoof sensor data to deceive the central control unit about home conditions in real time. This could result in false commands being issued, such as unlocking doors or disabling security systems, potentially compromising the safety and security of the residents. Another example is a false data injection attack on a trusted smart health monitoring system, where an attacker manipulates vital sign readings to provide inaccurate health information, leading to incorrect medical decisions. Such attack scenarios targeting the integrity of both the digital and physical aspects of trusted smart systems can have serious consequences, highlighting the need for robust security measures tailored to networked systems.

3. Proposed Approach

To decrease possibility of cyberattacks, architecture and requirements that are customized for the system must be considered. In addition, the risk of cyberattacks must be analyzed during the design phase to ensure that the system meets the security requirements. This proactive approach allows potential vulnerabilities and attacks to be identified and remedied, increasing the security of the system before it is deployed and executed.

This section presents a comprehensive approach to defending against cyberattacks in the design phase, focusing on tailored system architecture and analyzing the impact of attacks.

3.1. Research Design

Our study employs a mixed-method approach, combining qualitative threat modeling with quantitative impact analysis. The research design consists of two main steps: **(1) Threat Intelligence** and **(2) Attack Impact Analysis**. An overview of the proposed approach is shown in Figure 2. We systematically analyze the threat landscape, quantifying the probability of actions from threat lists to attack scenarios.

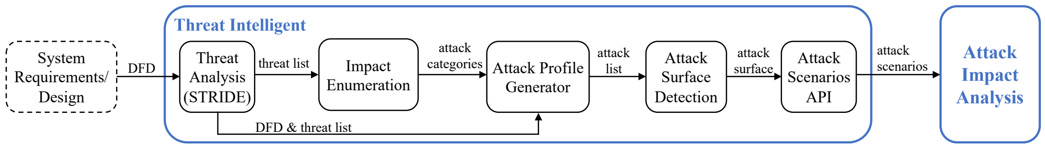


Figure 2. Our proposed approach for enhancing cybersecurity through comprehensive investigation of data flow-based attack scenarios.

In the first step, **Threat Intelligence**, the STRIDE framework and Microsoft Threat Modeling tool are employed to capture security properties by generating a threat list, introducing an impact enumeration, and generating an attack profile. The attack profile, acting as a baseline, identifies potential attack surfaces and an attack scenario for enhanced connectivity. This service is offered as an API, establishing a standardized connection or interface with other software in the pipeline. A detailed explanation of this step is provided in the following Section 3.2.

In the second step, **Attack Impact Analysis**, we take a closer look at the attack scenarios from the previous step, which entails assessment of the potential risks and vulnerabilities. By considering factors such as the severity, likelihood, and impact of these attacks, it is possible to gain insight into their potential impacts on the system. This assessment helps to prioritize the identified risks, allowing resources to be focused on eliminating the most critical vulnerabilities. Ultimately, this process strengthens the overall security posture of the system by supporting the development and implementation of proactive security measures.

3.2. Methodology

Our methodology extends the STRIDE framework and integrates data flow analysis to provide a comprehensive approach to cybersecurity in IoT and industrial systems. The key substeps in our approach are: (1) Threat Analysis using extended STRIDE, (2) Impact Enumeration, (3) Attack Profile Generation, (4) Attack Surface Detection, (5) Attack Scenario API Development, and (6) Quantitative Attack Impact Analysis

3.2.1. Threat Analysis Using Extended STRIDE

This is a crucial step in the proposed approach to identify attack scenarios, drawing on previous studies [13,14] and applying the STRIDE framework. It serves as the first step in our proposed process, and provides valuable insights and results that are utilized in the subsequent steps. We use the STRIDE framework, which relies on the Microsoft Threat Modeling tool and prior knowledge of the system design and requirements. We create a DFD using the tool, which automatically generates a list of potential threats. The outcomes of this step are important inputs for subsequent steps in our approach. These outcomes include the DFD, which shows a visual representation of the system's architecture, helping to identify potential attack surfaces. The other outcome is the threat list, which categorizes threats into Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service, and Elevation of Privileges. This threat list is important for creating the attack impact enumeration, while both the DFD and the threat list are essential in generating the attack profile.

3.2.2. Attack Profile Generator

In our proposed approach, the creation of an attack profile is a fundamental step for understanding and mitigating potential threats to a system. An attack profile encapsulates the characteristics, origins, and potential impact of each attack scenario that an adversary might employ to exploit vulnerabilities within the system. By systematically identifying these possible threats, we can prepare valuable inputs for the next steps. This section outlines our methodology for generating the attack profile.

3.2.3. Impact Enumeration

The threat impact enumeration plays a crucial role in systematic attack elicitation, making it valuable for cyber-threat modeling studies and CPS security assessment. The list also helps cybersecurity researchers and professionals to understand and analyze the different types of attacks prevalent in the cybersecurity landscape. In this step, we classify potential cyberattacks based on their characteristics and attributes as detailed in the STRIDE threat list. The origin of each category is provided in Table 1. The output of this step illustrates the diverse categories of attacks; an example is presented below. By grouping attacks into categories, it becomes easier to develop effective countermeasures and defense strategies against them.

- **Impersonation:** A type of cyberattack in which an attacker gains unauthorized access to a system or network by impersonating a legitimate user or device. This type of attack involves stealing or spoofing users, exploiting vulnerabilities in software or hardware, or using social engineering approaches to lead the system into accepting them as a trusted entity.
- **Crash:** A type of cyberattack that aims to disrupt or disable a system by causing it to crash or become unavailable. This type of attack typically involves sending malformed input or other unexpected data to a system or application, leading to unpredictable behavior that can cause a crash.
- **Data flow sniffing:** A type of cyberattack where an adversary intercepts and monitors data flowing between different components or subsystems of the system to capture

and analyze sensitive information such as passwords, login credentials, or other confidential data transmitted over the network.

- **Cross-site request forgery (CSRF):** This cyberattack exploits a system's trust by tricking users into sending a malicious request from a legitimate user system. The system executes the request without verifying its authenticity, allowing the attacker to take control of the system. This can result in various malicious actions, such as unauthorized data access, system setting manipulation, or even physical damage to the system.
- **Changing program execution flows:** A type of cyberattack that modifies the regular sequence of instructions executed by a computer program. Through this attack, the attacker can manipulate the program's execution flow to cause unintended actions, typically including unauthorized access to resources, data modification or deletion, or execution of arbitrary code.
- **Data repudiation:** A type of cyberattack in which an attacker manipulates or alters system data to deny responsibility or accountability for the actions associated with those data. This type of attack involves unauthorized access to the system and alteration of system logs or data records, making it difficult or impossible to trace the attack source or determine the actual state of the system.
- **Information not intended for disclosure (data breach/privacy leakage):** A type of cyberattack in which an unauthorized individual gains access to a system or network and steals sensitive information such as personal data, financial records, or intellectual property. The aim of this attack is to steal or leak data for the purpose of financial gain, identity theft, espionage, or other illegal activities.
- **Data flow interruption:** A type of cyberattack in which an attacker intentionally disrupts the normal flow of data and either interferes with the transmission or reception of data between system components or disrupts the communication channels that facilitate the exchange of information.
- **Remote code execution:** A type of cyberattack where an attacker gains unauthorized access to a system, then injects and remotely executes malicious code or commands on the target system.
- **Unauthorized access/unauthorized intrusion:** A type of cyberattack where an attacker gains entry or acquires privileged access to an application, network, or device without proper authorization or permission.
- **Resource consumption:** A type of cyberattack where an attacker intentionally consumes or exhausts critical system resources such as computational power, memory, CPU processing power, network bandwidth, and storage capacity.
- **Prevent access to data store:** A type of cyberattack in which an attacker intentionally disrupts or denies access to data stores within a system or network. This attack prevents legitimate users or system components from accessing and retrieving stored data.
- **Incorrect data delivery:** A type of cyberattack where an attacker intentionally manipulates, substitutes, or falsifies data that are transmitted or delivered within the system.
- **Denial of data reception:** A type of cyberattack in which an adversary intentionally prevents or obstructs the intended reception of data by the CPS.
- **Sending data to the attacker's target:** A type of cyberattack in which an attacker deliberately reroutes or forwards data generated or processed by the system to a destination of their choosing. This destination, known as the attacker's target, is typically under their control or influence.
- **Writing data to the attacker's target:** A type of cyberattack in which an attacker alters the system's data in such a way that the modified data are written or directed towards a destination controlled by the attacker. This destination typically resides outside the CPS environment and is under the control of the adversary. Such an attack can compromise data integrity and potentially cause harm to the system and its users.

By categorizing the attacks into general and CPS-specific attacks, it is possible to better understand the unique threats and vulnerabilities faced by cyber–physical systems. The list of CPS-specific attacks, including those targeting industrial control systems, sensor networks, and physical processes, serves as a foundation for the subsequent steps of our proposed approach. In the next step, we leverage the list to generate attack profiles that capture the characteristics, origins, and potential impacts of each attack scenario.

By explicitly highlighting the CPS-specific attack categories within the list and utilizing these for subsequent attack profiling and scenario generation, it becomes possible to focus on the unique vulnerabilities and challenges faced by cyber–physical systems.

In this step, we generate the attack profile by investigating and analyzing the threat report produced by the Microsoft Threat Modeling tool. For the purposes of this paper, the term 'threat report' specifically refers to the report generated by the tool. The list generated by STRIDE ($tList$) contains information about only one of the two interacting components. A component can be an entity, process, or data store in the DFD. Each exchange of information is represented by a data flow, called an *Interaction* (Int). We implement an additional functionality to identify the right Source (S) and Destination (D) for each Int. This information is needed in order to identify the origin and direction of the threat, where it enters the system, and where it goes. One of the challenges is that each component of the system can be a source for several components, and can also be a destination for several; therefore, we need to identify all the origins and destinations of each exchange of information. It may also be the case that the same information exchange (Int) takes place between several sources and destinations. We refer to such an interaction as redundant information. Consequently, while any of the Source, Destination, and Interaction can be redundant, but there must be a unique combination of Source, Interaction, and Destination such that $S \xrightarrow{Int} D$.

Because each data flow can cause more than one threat, we go through each of the associated threat categories, identify effects caused by all threat scenarios, and map them to the classes introduced in regard to the attack impact (see Section 3.2.3). We start with the DFD, consisting of a set E of generic entities only of size l, where $e \in E$. Our proposed approach retrieves the working dataset $tList$ of size $n \times m$ such that each $t \in tList$. Each t contains m attributes such that $t = \{tId, tDes, tTitle, tCat, tIntr\}$. A threat t originates from an entity e_i during information exchange and endangers the destined entity e_j and/or itself, also shown in Figure 3. It is clear from the case in Figure 3 that the entity named 'ControllerProcess' has an information flow through 'driveController()' to itself and another data flow to entity 'DoorProcess' through 'openDoor()', closeDoor()' and 'lockDoor()'. This creates at least three points of information leakage. Depending on the contents of the in-flow information, corresponding threats will be present; among others, these are our point of interest. This process is used to calculate how and where each point of interest leads, helping to identify security holes.

Let $aList$ be the attack list derived from $tList$, where $a \in aList$, as presented in Algorithm 1 line 1. To create the attack profile, in addition to $\{tId, tDes, tTitle, tCat, tIntr\}$, further information is determined such that $a = \{aId, tId, tDes, tTitle, tCat, tIntr, aEft, aSrc, aDst\}$, where aId is the attack ID, tId is the corresponding threat ID, $tDes$ is the threat description, $tTitle$ is the threat title, $tCat$ is the threat category, $tIntr$ is the interaction which caused the associated threat, $aEft$ is the resulting effect, $aSrc$ is the source of the attack, and $aDst$ is the destination of the attack. We initialize $aList$ with $tList$; $aList$ then continues to traverse $tList$ to further compute $a[aEft]$, $a[aSrc]$, and $a[aDst]$. Here, $a[aEft]$ is the result of the threat originated by $aSrc$ to $aDst$. This can be understood using the DFD diagram , example 1, which shows that 'FleetController' can cause a 'Crash' to 'RemoteController'. To determine these attributes, each threat t is evaluated based on the corresponding $t[tCat]$ to compute the impact $a[aEft]$ and identify the origin $a[aSrc]$ and destination $a[aDst]$.

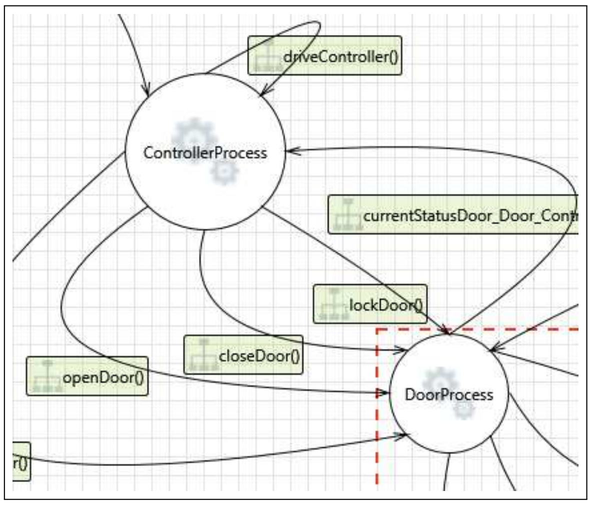


Figure 3. Example of a threat originating from a Source (S) to a Destination (D) through an Interaction (Int).

Algorithm 1 Identify the effect of each threat and map it to the corresponding attack impact classes

```
Require: load $dfd$
Require: generate $tList$
1: Initialize $aList \leftarrow tList$                              ▷ populate attack list
2: repeat
3:     for each $a \in aList$ do
4:         $c \leftarrow a[tCat]$                                   ▷ get threat category
5:         if c == 'Spoofing' then
6:             if a[tDes] endswith 'destination process' then
7:                 $Effect \leftarrow False$        ▷ data flow is outside the trust boundary
8:                 Impact goes to Information Disclosure
9:             else
10:                $Effect \leftarrow extractfrom(a[tDes])$
11:            end if
12:        else if c == 'Tampering' then
13:            if a[tDes] endswith 'data store' then
14:                $Effect \leftarrow' Corruption'$
15:            else
16:                if a[tDes] contains (denial of service) | (elevation of privilege) | (information disclosure) then
17:                    insertRow(aList,3) ▷ threat path for each category
18:                    $Effect \leftarrow False$
19:                    Impact goes to [Denial Of Service | Elevation Of Privilege | Information Disclosure]
20:                end if
21:            end if
22:        else if c == ['Denial Of Service' | 'Elevation Of Privilege' | 'Information Disclosure'] then
23:            $Effect \leftarrow extractfrom(a[tDes])$
24:        else if c == 'Repudiation' then
25:            $Effect \leftarrow extractfrom(a[tTitle])$
26:        end if
27:        $a[aEft] \leftarrow Effect$
28:    end for
29: until $EoF$
```

First, we evaluate those threats which can impact other categories, such as spoofing and tampering. For spoofing, threats may originate due to interactions across the trust boundaries, as shown in Algorithm 1, line 5, which implies the fact of threat propagation to other threat categories at line 8. The concept of the trust boundary is explained in the DFD diagram, Section 4, where 'SiteOperator' and SiteServer' are within the trust boundaries but 'SiteServer' and 'FleetController' are outside of it. In the case of tampering, threats related to data stores can cause data corruption, as shown in line 14, but do not lead to additional attacks. However, tampering with a process or entity can lead to other threats, such as *Denial of Service*, *Elevation of Privilege* and *Information Disclosure*. Therefore, such threats are examined in the related categories, as shown in line 19. Certain categories do not impact the destination, only the entity itself; thus, such categories are less critical in terms of depth of attack. When the correct category has been identified, the impact is determined from $a[tDes]$ or $a[tTitle]$ only. Considering all provided details, all possible effects are determined for each $t[tCat]$, as shown in Table 1. Different effects are introduced and listed in Section 3.2.3 as part of the attack impact list. We categorize these based on their characteristics and attributes.

Table 1. Category-wise list of possible effects for each threat.

Threat Category	Effect
Spoofing	Information not intended for disclosure (data breach/privacy leakage) Unauthorized access/unauthorized intrusion Data flow sniffing Incorrect data delivered Data written to the attacker's target Data sent to the attacker's target
Tampering	Changing the flow of program execution Crashes Cross-site request forgery (CSRF) Data flow sniffing Impersonate Information not intended for disclosure Interrupting data flows Preventing access to data stores Remote code execution Resource consumption
Repudiation	Data repudiation, Denial of data reception
Information Disclosure	Information not intended for disclosure Data flow sniffing
Denial Of Service	Interrupting data flows Crashes Preventing access to data stores Resource consumption
Elevation Of Privilege	Changing the flow of program execution Cross-site request forgery (CSRF) Impersonation Remote code execution

The rest of the categories in the threat report, namely, *Denial of Service*, *Elevation of Privilege*, and *Information Disclosure*, do not affect other categories, meaning that their effect is only related to the threats in which they are generated. Their effect can be extracted from the description of the corresponding threat, as shown in line 23; however, in the case of *Repudiation* the effect is extracted from the title of the threat, as shown in line 25. When the correct effect has been extracted, it can be added to the attack list. This process continues iteratively until all of the threats in the threat list have been evaluated and the attack list *aList* is populated with the corresponding effect of each threat.

The process continues until the source and destination of each *Interaction* have been identified. Because there can be redundant *Interactions*, traversing the *tList* sequentially for sources and destinations can be ambiguous. In the case of redundancies, the last source and destination always overwrite the previous events. Therefore, we have devised a two-step procedure involving the DFD and the *tList*. First, we evaluate the graphical representation of the system and translate each combination of source, interaction, and destination into tabular form. In this way, each data flow is translated into a scenario regardless of redundancy. We then group all *Interactions* to compute how many times each is repeated, as shown in Algorithm 2, line 4. This information is maintained in *sdList*, which becomes the reference point for the sources and destinations for each *Interaction*. With *sdList* in hand, we proceed to identify the source and destination using the description of each threat by going through the *sdList* category-by-category, starting with *Spoofing* and moving on to *Denial of Service*, *Elevation of Privilege*, *Tampering*, *Repudiation*, and *Information Disclosure*. In the case of a redundant interaction, the source and destination are identified from the description and *sdList* is used for a unique interaction, as shown in Algorithm 2, lines 7–20.

Algorithm 2 Identify the source and destination of each threat

```
Require: $aList$
Require: $dfd$
 1: Create $sdList$                                ▷ Source, destination list for interactions
 2: $sdList = \{Intr, Src, Dst, IFlag\}$
 3: $a[\text{IFlag}] \leftarrow \text{groupby}(a[\text{Int}])$     ▷ Count redundancy of Interactions
 4: repeat
 5:     for each $a \in aList$ do
 6:         if $a[IFlag] \geq 2$ then
 7:             $a[aSrc] \leftarrow extractfrom(a[tDes])$
 8:             $a[aDst] \leftarrow extractfrom(a[tDes])$
 9:         else
10:             repeat
11:                 for each $sd \in sdList$ do
12:                     if a[tInt] == sd[Intr] then
13:                         $a[aSrc] \leftarrow extractfrom(sd[Src])$
14:                         $a[aDst] \leftarrow extractfrom(a[Dst])$
15:                         break
16:                     end if
17:                 end for
18:             until $EoF$
19:         end if
20:     end for
21: until $EoF$
```

At this point, the attack profile in *aList* is ready to be used in calculating the paths and scenarios. The longer the path (steps), the greater the depth.

3.2.4. Attack Surface Detection

Given that an attack surface encompasses all potential entry points exploitable for malicious activities in a system, our approach leverages the *aList* to comprehensively identify these entry points. By aggregating all sources $a[aSrc]$ within *aList*, we obtain a holistic view of the entire attack surface inherent in the designed system. This enables a detailed analysis of system vulnerabilities associated with each threat. For instance, examination of *aList* reveals that the *FleetController* component is susceptible to various threats, including *Elevation* *of Privilege*, *Denial of Service*, *Spoofing*, and *Information Disclosure*. Consequently, the system is rendered vulnerable to potential adversities such as *Impersonation*, *Crashes*, *Unauthorized Access*, and *Data Flow Sniffing*, illustrating the multifaceted nature of the identified threats.

3.2.5. Attack Scenario API

The Attack Scenario API is created based on the attack profile, and displays all potential attack scenarios from each attack surface. This provides valuable information on the potential depth of impact and the number of actors within the system that could be affected in the event of a successful attack.

The idea is to take the sources as a starting point and proceed with their corresponding destinations. The destination in the next step becomes the source in the next step; thus, its destination is the source for the next step, and so forth. After the *aList* has been computed, we propose an automated approach to compute each path. Each path is composed of one or more components in succession. A threat generated from a source is a provision to reach the other node directly connected to it. Here, the scenario is a combination of multiple attack paths, where each path is a combination of three components, i.e., source, interaction, and destination.

The prerequisite for using the API is Python (Python 3 or later), which is usually installed in many operating systems, and can be installed easily. The API requires three arguments: the threat list, DFD, and our customized tool for parsing the graphical DFD, such that:

```
python STRIDEThreatAttack.py inputFile SourceDestinationFilename outputFile
```

Here, *inputFile* and *SourceDestinationFilename* should be provided with the extension and full path if they are not in the current folder. In addition, *outputFile* should be provided without extension but with the full path if it is not present in the current folder. The source code of the API can be viewed at https://t.ly/Rdfvr (accessed on 30 September 2024).

3.2.6. Attack Impact Analysis

While the core contributions of the proposed approach focus on the impact list, attack profiles, and attack scenario API, we also recognize the importance of analyzing the potential impacts and consequences of the identified attack scenarios. This involves evaluating the severity, likelihood, and potential impact of the attack scenarios generated by our approach. This analysis provides valuable insights into the most critical vulnerabilities and risks within a cyber–physical system.

By conducting an in-depth attack impact analysis, our proposed approach enables system designers and security professionals to gain a comprehensive understanding of the potential risks and threats facing cyber–physical systems. This includes evaluating the potential consequences and damage that could result from successful execution of each attack scenario, considering factors such as the impact on system availability, data integrity, and safety. In addition, we assess the probability of each attack scenario's occurrence based on factors such as the complexity of the attack, the attacker's required capabilities, and the existing security controls.

By combining the severity and likelihood assessments, it is possible to prioritize the identified attack scenarios based on their overall risk level, allowing mitigation efforts to focus on the most critical vulnerabilities. This risk prioritization process represents a crucial step in translating the generated attack profiles and scenarios into actionable security improvements for the target cyber–physical system. We believe that this analysis, while not a direct contribution of the proposed approach, is a crucial step in enhancing the overall resilience and robustness of systems against identified attack scenarios by informing the development and implementation of proactive security measures.

While our proposed approach provides a comprehensive and systematic approach to enumerating and analyzing potential threats in cyber–physical systems, it is important to consider how this approach can be applied in real-world settings. One key aspect is the integration of the approach with existing CPS architectures and security processes. The proposed approach depends on the presence of detailed system models and data flow diagrams, which may not always be readily accessible or easily obtainable in operational environments. In our future work, we aim to address this challenge by automating the process instead of relying on manual methods.

We also plan to provide guidance on how to incorporate the attack scenario analysis into the overall CPS security lifecycle, including during the design, deployment, and operational phases. This could involve integrating the proposed approach into existing risk assessment methodologies, incident response procedures, and security monitoring tools used in CPS environments.

To provide more tangible value and guidance for securing cyber–physical systems in real-world settings, the proposed approach should be validated and tested using more complex real-world CPS use cases and data to ensure its effectiveness and relevance in addressing the unique security challenges faced by these systems. This could involve collaborating with industry partners, critical infrastructure operators, and domain experts to gather relevant data and evaluate the approach's performance in realistic scenarios.

4. Case Study: A Remote Control System for Automated Guided Vehicles

In this section, we validate our proposed approach in the context of a research project [15]. The case study involves the use of several automated guided vehicles (AGVs) called Hxs. These Hxs are used to transport materials from a quarry site. A wheel loader and an excavator are used to load the materials onto the Hx, complementing the fleet of autonomous Hxs. The Fleet Control System manages the active Hx fleet, providing features such as traffic management and job assignment for each device. Hx functionality relies on wireless networks and precise commands, which are necessary in order to function. An Hx Remote Operator is able to fully control a single Hx using a remote control, especially when activating, deactivating, maintaining, or integrating a new Hx into production. Site Operators oversee the quarry from a control room where the Site Servers are located.

As illustrated in Figure 4, this case study demonstrates the process of taking control of a specific Hx, such as Hx-1. The Hx Remote Operator initiates the control request and transmits it to Fleet Control. Based on the circumstances, Fleet Control may either approve the request (Grant Control) or reject it (Reject Control). Simultaneously, while Fleet Control transmits this information to the Site Server, the *Info Status* message provides details about the active Hx. After accepting the Remote Control request, Fleet Control dispatches the task *Approve Checkout* to Hx-1. After completing this task, the Hx Remote Operator takes control of Hx-1. In addition, the Hx Remote Operator can return control of Hx-1 to Fleet Control. In this scenario, Fleet Control sends *Take Control* to Hx-1, which then listens for the commands issued by Fleet Control.

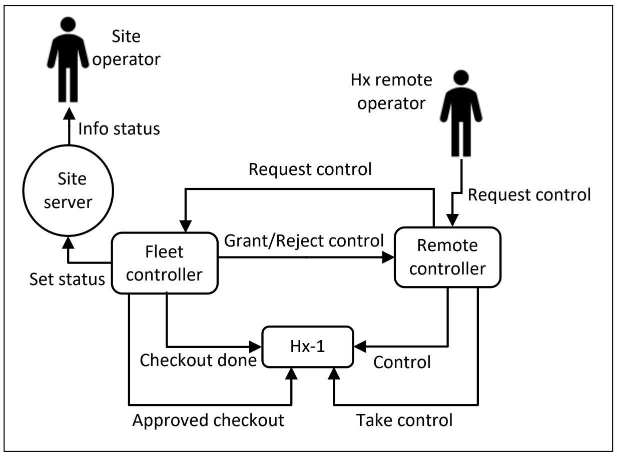


Figure 4. Control structure diagram of the remotely controlled Hx-1 [16].

To apply our proposed approach, we first review the system requirements and design, then create a DFD to anticipate potential attack types and conduct an attack impact analysis of the system. The DFD is depicted in Figure 5. This DFD is instantiated within the Microsoft Threat Modeling Tool, portraying the concepts elucidated in Figure 4 in the form of a data flow diagram. A detailed explanations of the DFD is not included in this paper.

We run the *Attack Profile Generator* to determine the origin, location, and direction of the threats along with potential attack types for this specific case study. The output of this execution is an attack list that serves as input for the next step, detection of attack surfaces, in which we activate the *Attack Surface Detection* component to identify all possible source and destination components as well as a catalog of potential attacks that could transpire between each component pair. The *Attack Scenario API* is then executed to pinpoint all potential attack scenarios that could be used by attackers targeting the system. The output of the case study can be viewed at https://t.ly/Rdfvr (accessed on 30 September 2024).

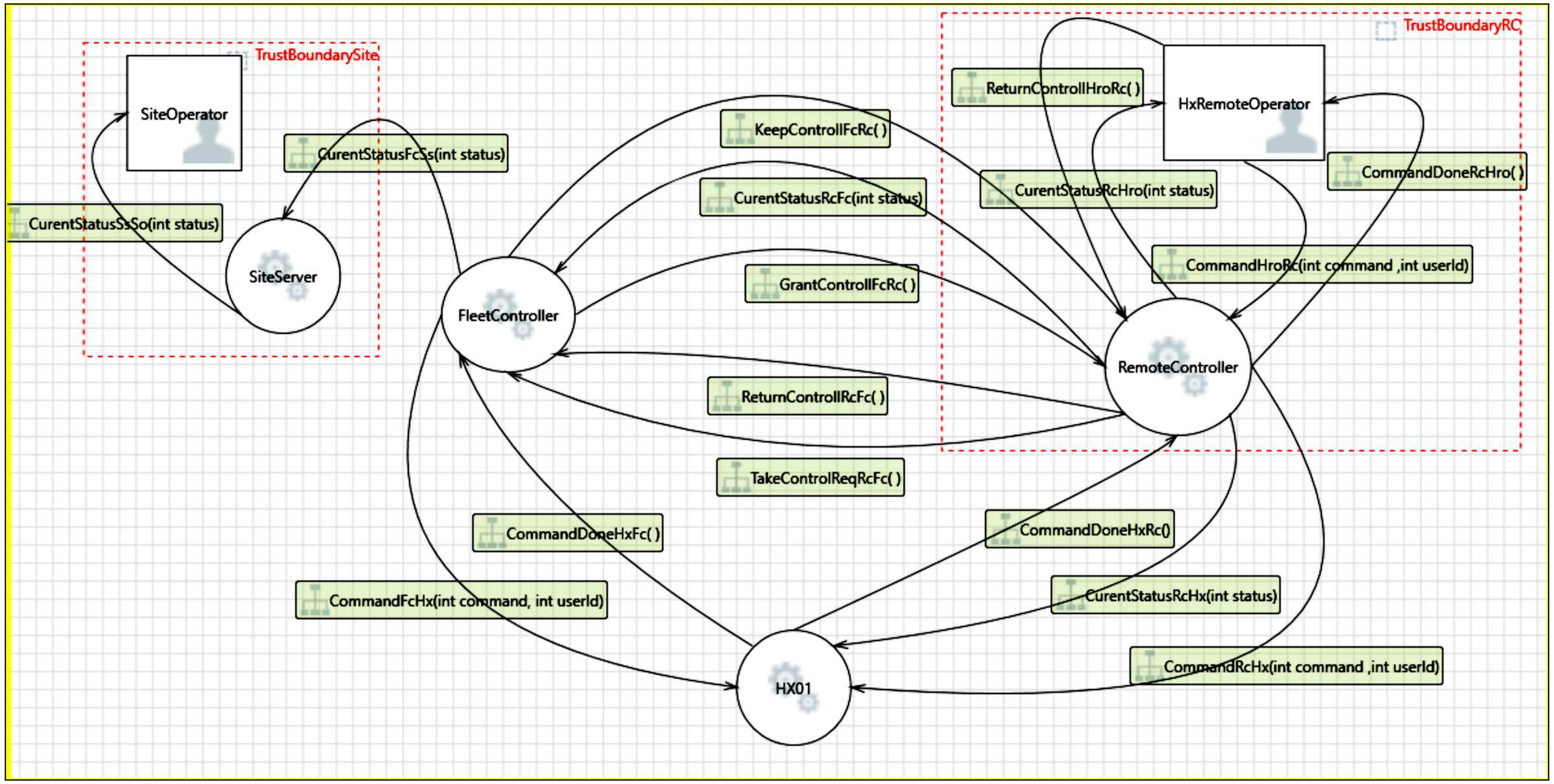


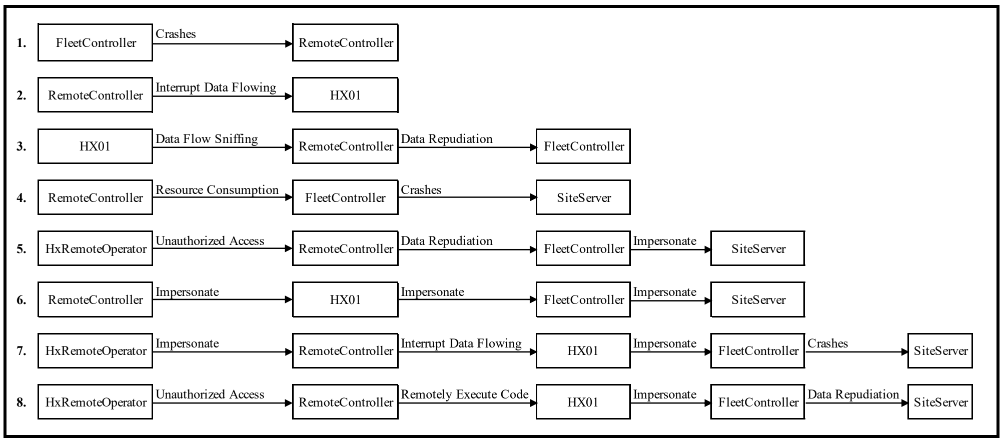
Figure 5. DFD diagram of the *remote control Hx-1* case study.

5. Results and Comparative Analysis

In this section, we present the results of implementing the proposed steps, as illustrated in Figure 2 and explained in Section 3, leveraging the case study from Section 4.

5.1. Key Findings

We managed to generate 3100 attack scenarios by running the *Attack Scenario API*, providing a non-comprehensive list of results prioritized for addressing fundamental inquiries and formulating definitive conclusions. After conducting a thorough analysis of the case study, we selected eight scenarios from the obtained pool of 3100 scenarios which illustrate the ability of our proposed approach to generate a wide range of scenarios. These scenarios are depicted in Figure 6. For instance, in scenario five the remote control Hx-1 system is vulnerable to three types of attacks. The first type is an *Unauthorized Access* attack that occurs during the data transfer between *HxRemoteOperator* and *RemoteController*. The second type is a *Data Repudiation* attack that may occur during the data exchange between *RemoteController* and *FleetController*. The third type is an *Impersonation* attack which might occur during the data transfer between *FleetController* and *SiteServer*. All obtained scenarios, results, and related data are accessible at https://t.ly/Rdfvr (accessed on 30 September 2024).



1. FleetController → Crashes → RemoteController
2. RemoteController → Interrupt Data Flowing → HX01
3. HX01 → Data Flow Sniffing → RemoteController → Data Repudiation → FleetController
4. RemoteController → Resource Consumption → FleetController → Crashes → SiteServer
5. HxRemoteOperator → Unauthorized Access → RemoteController → Data Repudiation → FleetController → Impersonate → SiteServer
6. RemoteController → Impersonate → HX01 → Impersonate → FleetController → Impersonate → SiteServer
7. HxRemoteOperator → Impersonate → RemoteController → Interrupt Data Flowing → HX01 → Impersonate → FleetController → Crashes → SiteServer
8. HxRemoteOperator → Unauthorized Access → RemoteController → Remotely Execute Code → HX01 → Impersonate → FleetController → Data Repudiation → SiteServer

Figure 6. Selected attack scenarios from the *remote control Hx-1* case study.

■ **Analysis of Attack Scenario Distribution:** Figure 7 illustrates the distribution of attack scenarios generated for the remote control Hx-1 case study. The analysis revealed a total of 3100 unique attack scenarios, representing various combinations of system components involved and different attack types. The graph categorizes these scenarios based on the number of components involved, ranging from two components (e.g., *FleetController* and *RemoteController*) to five components (e.g., *FleetController*, *RemoteController*, *HX01*,*HxRemoteOperator*, and *SiteServer*). Notably, the majority of the identified attack scenarios (71%) involve four components, highlighting the complexity and interconnectedness of the system. While scenarios involving fewer components (two or three) may seem more straightforward to identify manually, the prevalence of scenarios with four or more components underscores the value of our systematic approach. As systems become more intricate with numerous interacting components, the ability to comprehensively enumerate all potential attack scenarios becomes increasingly challenging without an automated process. Our analysis reveals that scenarios involving a higher number of components are not only more numerous but also more intricate, making them harder to anticipate and mitigate without a thorough understanding of the system's architecture and data flows. By systematically identifying and analyzing these complex multi-component attack scenarios, our approach provides system designers and security experts with valuable insights into the system's potential vulnerabilities and attack paths. It is important to note that while Figure 7 presents a high-level overview of the attack scenario distribution, our analysis delves deeper into the specific components involved as well as different attack types and their combinations. This granular analysis, which is not feasible to present in its entirety, enables us to draw more nuanced conclusions and provide targeted recommendations for enhancing the system's security posture.

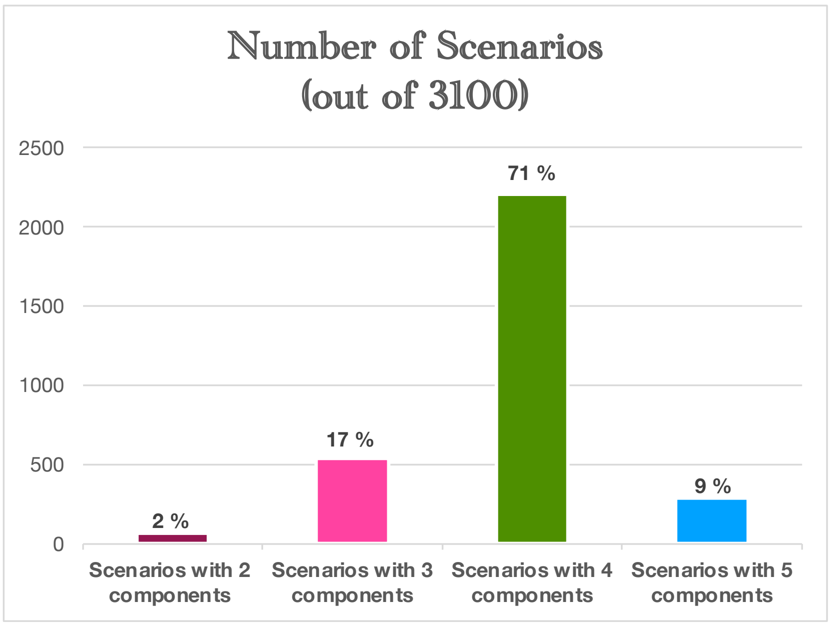


Figure 7. Frequency of attack scenarios in the *remote control Hx-1* case study.

■ **Analysis of Component Engagement in Attack Scenarios:** Figure 8 illustrates the distribution of component occurrences across all identified attack scenarios and their corresponding degrees of involvement in these scenarios. This analysis provides valuable insights into the relative importance and vulnerability of each system component within the context of potential attacks. Notably, the *RemoteController* component exhibits the highest degree of involvement, appearing in 26% of all attack scenarios. This observation highlights the critical importance of securing this component, as its vulnerability could serve as a focal point for attackers to compromise overall system security. By implementing robust security measures and safeguarding data transmission to and from the *RemoteController*, the system's resilience against attacks can be significantly enhanced. The analysis further reveals a prioritization of components based on their involvement in attack scenarios. Following *RemoteController*, *FleetController* emerges as the next most critical component, followed by *SiteServer*, *HX01*, and *HxRemoteOperator*, respectively. This prioritization can guide security professionals in allocating resources and implementing targeted security measures to protect the most vulnerable components. While deriving such insights may seem straightforward for systems with fewer components, the true value of this analysis becomes evident in larger and more complex systems. As the number of components and their interactions increase, manually identifying and prioritizing vulnerable components becomes increasingly challenging.

Our proposed approach enables the systematic extraction of this critical information, providing system security analysts with invaluable insights to enhance the overall security posture of the system. By visualizing the involvement and prioritization of different components, security professionals can develop a comprehensive understanding of the system's attack surfaces and potential vulnerabilities. This knowledge can then inform the development and implementation of targeted security measures, ensuring that the most critical components are adequately protected against potential attacks.

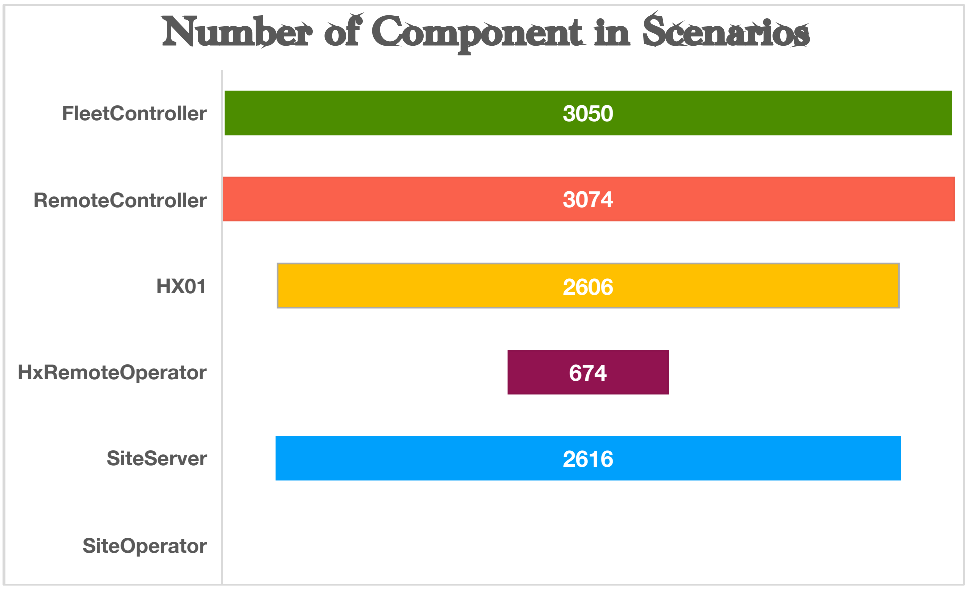


Figure 8. Component counts in scenarios for the *remote control Hx-1* case study.

■ **Analysis of Attack Scenario Distribution and Prioritization based on Attack Types:** We performed an analysis of the distribution of attack scenarios based on our classification of the different attack types, as explained in Section 3.2.3. This classification includes sixteen different types of attacks. Among the 3100 scenarios analyzed, a total of 8920 attacks were identified, with some scenarios having multiple attack types. Figure 9 provides a visualization of the categories of attack types and their corresponding frequencies obtained by aggregating the number of attack scenarios within each category. The chart shows that the *Impersonate* category accounts for the highest proportion of attack types at 17%. This finding highlights the importance of impersonation-based attacks and the need for robust authentication and access control mechanisms within the system. Security experts should prioritize the development of countermeasures and recovery strategies specifically tailored to mitigate these types of attacks, as they represent a significant threat to the integrity of the system. The *Unauthorized Access* category follows closely in second place, with 11% of the identified attack types. This category includes attacks aimed at gaining unauthorized access to system resources or data, highlighting the importance of implementing comprehensive access control policies and monitoring mechanisms to detect and prevent such attempts.

The pie chart also highlights other notable attack categories, such as *Interrupt data flowing* (7%), *Prevent access to data store* (7%), and *Remotely execute code* (7%). These categories represent different attack paths ranging from disruption of data flow to remote execution of malicious code, each presenting unique challenges and requiring tailored security measures. By exploring this visual representation, system security analysts can prioritize the development of recovery mechanisms and security controls based on the most critical attack types within the system. The chart serves as a valuable tool for identifying areas that require immediate attention and resource allocation, enabling a proactive approach to enhancing the system's overall security posture.

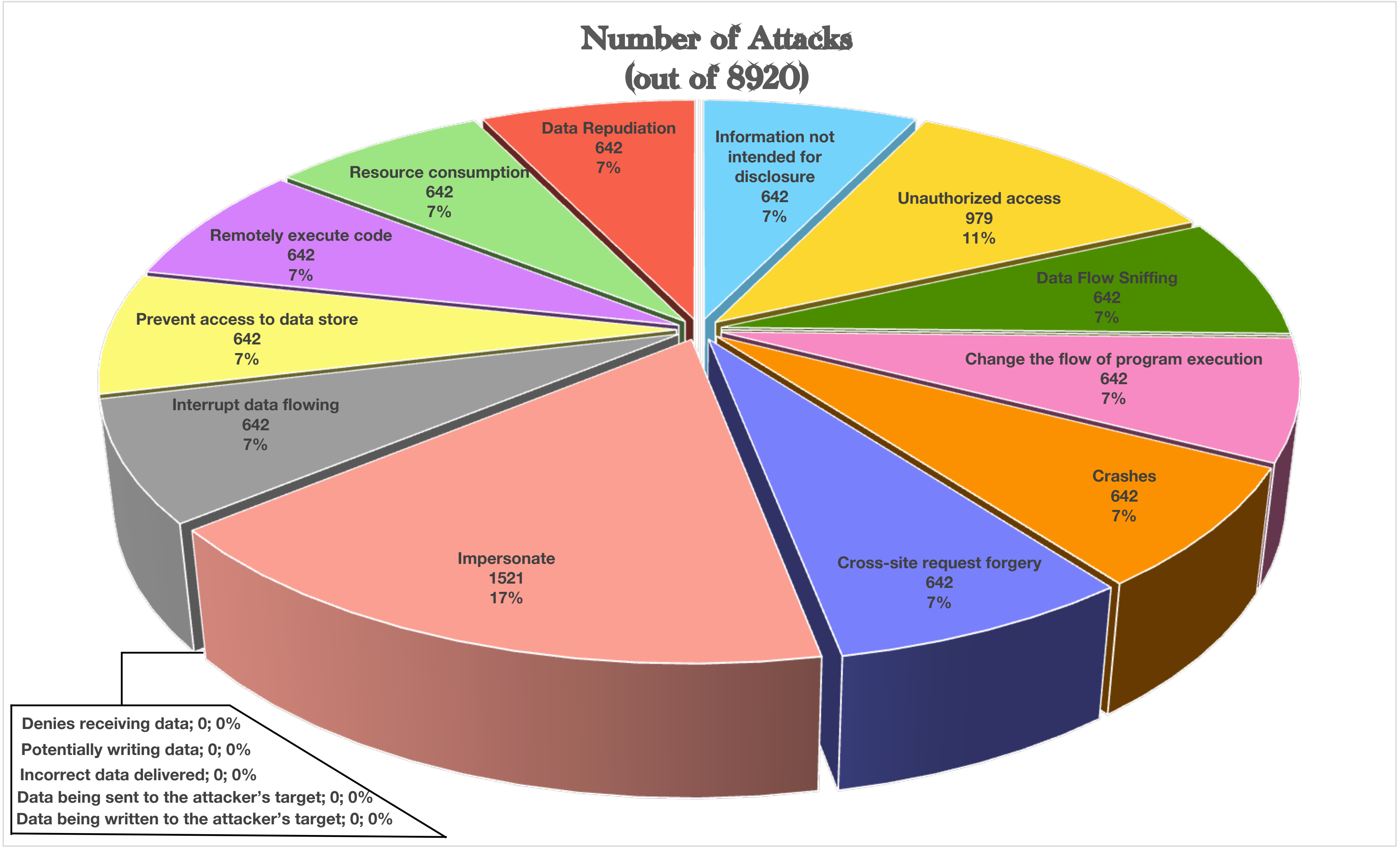


Figure 9. Variety of potential attack types targeting the *remote control Hx-1* case study.

■ **Prevalence and Distribution Analysis of Cyberattack Scenarios:** We analyzed the number of attack scenarios associated with each specific type of attack in the remote control Hx-1 case study. This can provides insight into the prevalence and distribution of the different attack types. The visualization in Figure 10 shows the distribution of scenarios with specific cyberattacks, illustrating the prevalence of the different attack types. The X-axis represents the number of scenarios, while the Y-axis indicates the types of attack. This chart provides a detailed overview of the attack landscape, enabling security professionals to effectively prioritize and allocate resources. The analysis shows that the *Impersonate* attack type is the most prevalent, accounting for 19% of the identified attack scenarios. This finding underlines the importance of implementing robust authentication and access control mechanisms in the system to mitigate the risks associated with impersonation attacks.

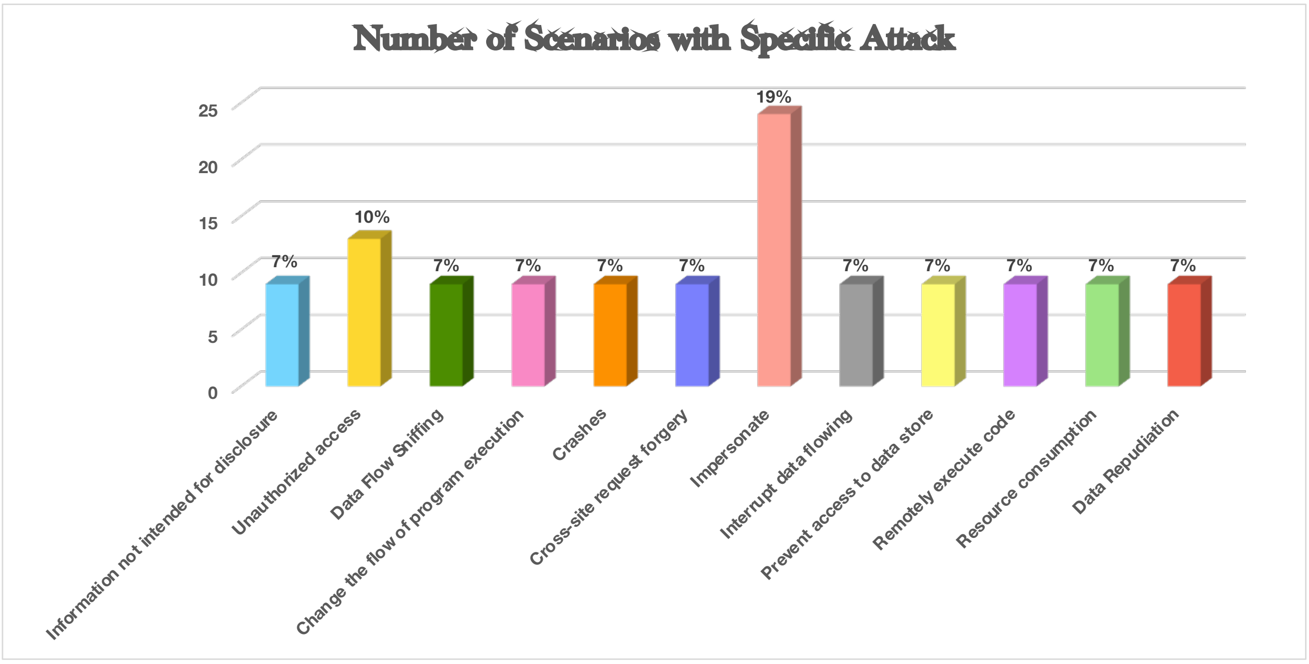


Figure 10. Distribution of attack scenarios by specific attack type in the *remote control Hx-1* case study.

Furthermore, the chart highlights several other notable attack types, each comprising 7% of the scenarios. These include *Information not intended for disclosure*, *Unauthorized Access*, *Data Flow Sniffing*, *Change the flow of program execution*, *Crashes*, *Cross-site request forgery*, *Interrupt data flowing*, *Prevent access to data store*, *Remotely execute code*, *Resource consumption*, and *Data Repudiation*. This diverse range of attack types underlines the complexity of the threat landscape and the need for a comprehensive security strategy that addresses different attack paths. By analyzing the distribution of attack scenarios by specific attack types, security professionals can gain valuable insight into the areas that require immediate attention and prioritization. For instance, the high prevalence of *Impersonation* attacks may result in the need to implement multi-factor authentication, secure communication protocols, and rigorous identity management practices. The chart serves as a valuable tool for risk assessment and resource allocation. Higher-frequency attack types may warrant greater investment in security controls, incident response planning, and employee training to improve the overall resilience of the system against these threats.

■ **Results Overview:** Our analysis of the obtained results addresses several critical issues regarding the security landscape of the system. The largest number of components observed in the attack scenarios is five: *FleetController*, *RemoteController*, *HX01*, *HxRemoteOperator*, and *SiteServer*. Further investigation shows that the system can be exposed to attacks through a variety of attack types, with 8920 cases in total.

For a deeper analysis, we investigate specific attack scenarios based on different parameters. For example, we find that there are 397 attack scenarios with four components that involve *HX01* as the initial component and *SiteServer* as the final component, all of which involve *Impersonation* attacks. This analysis can provide valuable insights when attack scenarios with different initial and final components are examined. For instance, in *Impersonation* attack scenarios with *HxRemoteOperator* as the initial component and *SiteServer* as the final component, a population of 288 attack scenarios is identified when five components are considered, while only one scenario appears when two components are considered. Similar patterns emerge when we extend this analysis to the *Unauthorized Access* attack type, with the size of the population varying depending on the number of components and configurations.

As mentioned earlier, our investigation involved a thorough analysis of 3100 scenarios in the provided case study, each of which has a different number of components. While configurations with different component counts provide valuable insights, scenarios with higher component counts, such as 4 or 5, reveal complicated interactions between components and potential attack paths. On the other hand, simpler scenarios consisting of 2 or 3 components are less comprehensive but easier to handle. More detailed results can be found in Tables 2 and 3, which provide a comprehensive overview of our analysis of the results.

Table 2 outlines the frequency and characteristic features of various attack scenarios. Each row corresponds to a scenario with a different number of components, providing insight into the complexity and diversity of these scenarios. This table provides a comprehensive overview of the distribution and characteristics of attack scenarios, enabling in-depth understanding of potential security vulnerabilities. In parallel, Table 3 provides a comprehensive representation of the distribution of attack types based on the complexity of the scenarios. The table illustrates the distribution of different attack types of scenarios with a component count of 2 to 5, providing valuable insights into the prevalence of certain attack types at the different complexity levels of the scenarios.

Table 2. Frequency and characteristics of attack scenarios.

Property / Scenarios	Number of Scenarios	Most Frequent Initial Component (#)	Most Frequent Final Component (#)	Most Frequent Attack type(s)
Scenarios with 2 components	64	FleetController (25)	RemoteController (26)	Impersonate
Scenarios with 3 components	540	HX01 (168)	HX01 (180)	Impersonate
Scenarios with 4 components	2208	HX01 (1728)	SiteServer (2160)	Impersonate
Scenarios with 5 components	288	HxRemoteOperator (288)	SiteServer (288)	Impersonate
Total	3100			

Table 3. Distribution of attack types based on scenario complexity.

Attack Types / Scenarios	Information Not Intended for Disclosure	Unauthorized Access	Data Flow Sniffing	Change the Flow of Program Execution	Crashes	Cross-Site Request Forgery	Impersonate	Interrupt Data Flowing	Prevent Access to Data Store	Remotely Execute Code	Resource Consump-tion	Data Repu-diation	Total Number of Attack
Scenarios with 2 components	5	6	5	5	5	5	8	5	5	5	5	5	64
Scenarios with 3 components	81	105	81	81	81	81	165	81	81	81	81	81	1080
Scenarios with 4 components	508	676	508	508	508	508	868	508	508	508	508	508	6624
Scenarios with 5 components	48	192	48	48	48	48	480	48	48	48	48	48	1152
Total	642	979	642	642	642	642	1521	642	642	642	642	642	8920

We have excluded the attacks with zero outcomes from the table
(Incorrect data delivered, Data being written to the attacker's target, Data being sent to the attacker's target, Denies receiving data, and Potentially writing data)

5.2. Comparison with Existing Methods

To contextualize our results within the existing literature, we compared our approach to several prominent cybersecurity methodologies, including both traditional and more recent approaches: *STRIDE*, *MITRE ATT&CK* [17], *PASTA (Process for Attack Simulation and Threat Analysis)* [18], *OCTAVE Allegro* [19], *TARA (Threat Assessment and Remediation Analysis)* [20], *FAIR (Factor Analysis of Information Risk)* [21], and *NIST Cybersecurity Framework (CSF)* [22].

Table 4 provides a comparative analysis of these methodologies alongside our approach.

Table 4. Comparative analysis of cybersecurity methodologies.

Aspect	Our Approach	STRIDE	MITRE ATT&CK	PASTA	OCTAVE Allegro	TARA	FAIR	NIST CSF
Threat Identification	Comprehensive	Limited	Comprehensive	Comprehensive	Comprehensive	Comprehensive	Partial	Comprehensive
Data Flow Integration	Yes	No	Partial	Yes	Partial	Partial	No	Partial
Attack Scenario Mapping	Detailed	Basic	Detailed	Detailed	Moderate	Detailed	No	Moderate
Impact Quantification	Yes	No	No	Yes	Yes	Yes	Yes	Partial
IoT/Industrial Focus	Yes	No	Partial	Adaptable	Adaptable	Yes	No	Adaptable
Threat Intelligence Integration	Yes	No	Yes	Yes	Partial	Yes	No	Yes
Risk-based Approach	Yes	No	No	Yes	Yes	Yes	Yes	Yes
Continuous Updates	Yes	No	Yes	No	No	Yes	No	Yes

Our approach has several advantages over existing methodologies. It provides comprehensive threat identification, similar to MITRE ATT&CK, PASTA, and TARA, surpassing the limitations of traditional STRIDE. A unique feature of our methodology is the full integration of data flow analysis, an aspect that is not fully addressed by most other approaches, including recent ones such as FAIR and NIST CSF.

In terms of attack scenario mapping, our approach provides detailed insights comparable to MITRE ATT&CK and TARA, offering more depth than traditional methods and some newer frameworks such as FAIR. Our methodology is specifically tailored for IoT and industrial systems, another aspect that is not fully addressed by most other approaches, with TARA being a notable exception. We incorporate threat intelligence, a feature shared with newer methods such as MITER ATT&CK, PASTA, and NIST CSF. Our approach combines a risk-based methodology with impact quantification, similar to PASTA, OCTAVE Allegro, and FAIR but with a specific focus on IoT and industrial systems.

Finally, similar to MITRE ATT&CK, TARA, and NIST CSF, our approach is designed to be continuously updated to ensure that it remains relevant in the face of evolving threats. These comprehensive features make our methodology a robust and adaptable solution to the unique challenges of IoT and industrial cybersecurity.

6. Discussion

Vulnerabilities in a system can arise from unexpected and seemingly unlikely sources, but can also arise from more obvious origins that expose the system to the external environment and potential access points. Even a highly efficient function may have security vulnerabilities that were not anticipated or considered during the security development cycle [23]. The coherence of internal functionality can also be compromised by various means. Exploiting the data flow between different actors is one such way, as even a well-designed system can inadvertently reveal information about its operation. In particular, data in transit can be more susceptible to security issues than data at rest. This depends on factors such as transmission channels, data encapsulation methods, data protection techniques, and data communication protocols.

The solution proposed in this paper focuses on assessing the impact of threats and the scope of attacks within a system design by comprehensively analyzing data flows. Our real-world case study demonstrates the benefits of threat modeling and attack impact analysis in enhancing system security. This methodology makes it possible to anticipate convincing attack scenarios and identify suitable defense strategies for future investigations.

Our investigation delves deeply into the intricate distribution of attack scenarios in a real system. We analyze the severity, length, and complexity of each scenario using our systematic approach. This approach involves several steps: *Threat Analysis (STRIDE)*, *Attack Taxonomy*, *Attack Profile Generation*, *Attack Surface Detection*, and the *Attack Scenarios API*. We have created a comprehensive collection of attack scenarios, with each scenario characterized by its source and target components and the impact that it creates. Through a systematic analysis, we have quantitatively assessed the path complexity. We have also identified the prevalence and hierarchies of different attack scenarios that reflect the differences in their composition.

Our investigation aims to identify the critical attack paths that pose a higher risk to the security of the system. A higher risk can be determined by the number of attack types in a path or the number of components involved in an attack scenario, which depends on the definition set by the system designer or security expert for a specific system in a particular company. This is achieved through a meticulous attack impact analysis process, wherein we examine various attack scenarios and layer different attack typologies. Following the attack impact analysis, the result of our framework is the generation of a quantitative metric calibrated to expert knowledge and contextual understanding that determines the severity of each scenario. This helps security experts to prioritize the most important attack possibilities and understand their implications.

Tangible Impact and Effectiveness of Our Proposed Methodology

Our methodology provides insightful answers to various questions in the Attack Impact Analysis step. These questions are crucial for understanding the security dynamics and potential vulnerabilities inherent in the system architecture. For instance, we aim to answer questions such as: Which attack type can be the most relevant one for this case study? What is the largest number of components observed in attack scenarios/attack paths? and, How many times can attackers target the system with different kinds of attack types? In the context of the case study presented in Section 4, these questions serve as a foundation for a detailed discussion on the richness of the attack scenario landscape, the prominence of certain threats, and the implications of attack complexity. The findings derived from these questions will be instrumental in shaping strategies for fortifying the system against potential cyberattacks. In the context of our case study presented in Section 4, these questions serve as a foundation for a detailed discussion on the richness of the attack scenario landscape. This study and its results serve as a preliminary outcome for future work in our ongoing research, the concept of which is presented in [24]. A list of potential questions related to the remote control Hx-1 case study can be found at https://t.ly/Rdfvr.

7. Related Work

The rise in cyberattacks highlights the importance of researching and implementing effective methods for securing computer systems. To achieve this, various security techniques such as encryption, cryptography hashes, deploying security patches, monitoring audit trails, and adopting reliable network protocols are utilized. These techniques are employed in data storage, communication, and industrial control systems to ensure their safety and security. The fields of threat detection and prevention [25], secure communication and data protection [26], intrusion detection and response [27], trust and accountability [28,29], resilience and fault tolerance [30], security in IoT and edge computing [31,32], security in cloud-based systems, and considering human factors and usability [33,34] have all witnessed continuous advancements and evolution.

Researchers are developing advanced algorithms and techniques to detect and prevent cyber threats in cyber–physical and intelligent systems [35], focusing on secure communication protocols, access control mechanisms, and authentication techniques to protect data within these systems. Intrusion detection systems are being improved to monitor and respond to unauthorized activities [36]. Trust and accountability mechanisms are being established to verify the integrity of the devices and software used [37]. Resilience and fault tolerance techniques aim to ensure that systems can recover from cyberattacks or failures. Security concerns specific to IoT, edge computing, and cloud-based systems are also being addressed [38]. Finally, human factors and usability must considered in order to develop user-friendly interfaces and enhance user awareness and training [39].

One effective approach in this domain involves the development of frameworks that facilitate the design, construction, and deployment of secure systems as well as the analysis and evaluation of existing systems. These frameworks emphasize the provision of methods, metrics, and tools that enable quantitative assessment of cyber threats. Furthermore, they support the development and implementation of cybersecurity programs aimed at mitigating vulnerabilities, while addressing the performance, reliability, and safety requirements specific to manufacturing systems.

In line with this research trajectory, several modeling, simulation, and machine learning techniques have emerged for analyzing CPS security. These approaches aim to enhance our understanding of CPS security and contribute to countermeasure development.

Nigam and Talcott [40] used Maude [41] to automate security analysis of the protocols in Industry 4.0 applications. This approach formalizes networked sets of devices and a symbolic intruder model in rewriting logic; in particular, attacks can be detected by changing the input and output behavior of the system and analyzing its effect on the system's behavior. Covert attacks and replay attacks were modeled and analyzed in this study; however, combinations of attacks were not considered.

A study conducted by Deng et al. [42] employed the STRIDE methodology to define privacy issues and establish a mapping between the system elements and identified privacy threats. The authors employed threat tree analysis and privacy-enhancing technologies (PETs) to identify privacy threats and align them with existing threats. This study introduced LINDDUN as a systematic threat modeling methodology, which adopts an approach similar to that of STRIDE for addressing threat modeling. LINDDUN stands for Linkability, Identifiability, Nonrepudiation, Detectability, Information Disclosure, Content Unawareness, and Policy and Consent Noncompliance, representing the privacy threats identified in their study. Using this framework for other use cases requires updating the privacy threats, as the privacy categories in this work are more conceptual. In addition, real-world case studies should be undertaken to evaluate the correctness of this framework.

With the aim of enhancing security during the design phase of software development, Kreitz et al. [43] devised an in-house tool in Java built upon the Oracle security tool. Their study concentrated on evaluating system security from various perspectives, including the integration of static code analysis into the development process, the software development life cycle, and the principles of security-by-design.

Numerous studies have employed the STRIDE framework to discern security threats across diverse domains and use cases. For instance, Olayemi [44] explored security concerns in smart homes and healthcare systems, leveraging STRIDE analysis to identify potential threats, offering potential countermeasures tailored to this use case. Additionally, the integration of intrusion detection and prevention systems within smart home networks was investigated to enhance the efficient identification and prevention of attacks and malicious activities. In another study, Xu et al. [45] contributed to the utilization of STRIDE threat modeling at various levels of abstraction within software systems. Their work focused on automating the generation of security tests through STRIDE threat modeling. Their proposed approach emphasizes the repeatability and reproducibility of security tests, facilitating their execution during software development processes that involve frequent changes. While the authors sought to mitigate the identified security threats using analysis and modeling results, their study revealed that only a limited number of mutants were successfully eliminated in the examined use cases. They attributed this outcome to the exclusive utilization of free threat scanning and security tools for system analysis.

Several methods have been developed to describe and analyze attack surfaces in cybersecurity contexts. Attack trees, first introduced by Ebrahimi et al. [46], provide a structured, hierarchical representation of potential attack paths. They provide a visual way to analyze security threats, but can become very complex for large systems. Attack graphs [47] also represent possible attack paths in a system, and are more flexible than attack trees; however, they can be difficult to create and analyze for complex systems. The DREAD model, which was also developed by Microsoft, helps to quantify, compare, and prioritize the risks of evaluated threats [48]. Data flow analysis has also been incorporated into security assessment in various ways [49]. Taint analysis tracks the flow of untrusted data through a program [50], while information flow control enforces security policies on data flows within a system [51].

Comparison with Related Work

Our proposed approach builds upon and extends the existing approaches discussed above, particularly the studies by Olayemi [44], Xu et al. [45], and Deng et al. [42]. In [44] the use of the STRIDE threat modeling approach was explored to identify potential security threats in smart homes and healthcare systems. While their work demonstrated the applicability of STRIDE in these domains, it did not provide the level of granularity and systematic analysis of attack scenarios that our approach offers. In contrast, our approach goes beyond simply identifying threats and provides a comprehensive impact list of CPS-specific attack types. This list serves as a foundation for generating detailed attack profiles that capture the source, destination, and depth of potential attacks. This level of granularity allows for a more thorough understanding of the attack surfaces and potential impact within cyber–physical systems. In [45], the authors also utilized the STRIDE threat modeling technique; however, their focus was on automating generating security tests within software systems. While their work aimed to improve the repeatability and reproducibility of security testing, it did not address the unique challenges of cyber–physical systems, such as the tight integration of cyber and physical components; our approach, on the other hand, is specifically tailored to the CPS domain, incorporating the STRIDE approach to generate a CPS-centric impact list. When combined with the systematic analysis of attack scenarios and their propagation through the system, this list provides a more comprehensive solution for identifying and mitigating threats in cyber–physical environments.

Our data flow-based attack scenario analysis differs from existing approaches in several important aspects. Unlike many existing methods that focus on either attack surfaces or data flows, our approach integrates both, providing a more holistic view of the system's vulnerabilities. We extend the STRIDE approach with a detailed mapping of attack scenarios, addressing a limitation of the original STRIDE approach. In terms of existing methodologies, attack trees and attack graphs provide valuable frameworks for understanding potential attack paths; however, they often lack the detailed integration of data flow analysis that our approach provides. While the STRIDE approach is useful for enumerating threats, it does not provide the necessary depth in mapping attack sequences, which can leave critical systems vulnerable to sophisticated multistep attacks. Our research aims to fill this gap by combining STRIDE-based threat modeling with detailed data flow analysis and attack scenario mapping.

Furthermore, the integration of an attack scenario API in our approach is a unique feature that sets it apart from related studies. This API generates and exposes potential attack chains, facilitating seamless integration with other security tools and processes. This standardized interface enhances the overall utility and applicability of our approach within the CPS security ecosystem. In contrast to the study published in [42], which focused on privacy threats and the LINDDUN framework, our approach specifically addresses security threats and attack scenarios in cyber–physical systems. While both approaches leverage the STRIDE methodology, our work is tailored to the unique challenges and vulnerabilities present in CPS contexts, providing a more targeted solution for this domain. In summary, our proposed approach represents a significant advancement in the field of attack scenario identification and analysis for cyber–physical systems.

8. Conclusions and Future Work

In this paper, we provide a comprehensive approach for the evolving landscape of industrial cybersecurity that emphasizes the early integration of cybersecurity measures. The increasing integration of the IoT requires careful monitoring of critical assets in cyber–physical systems. Recent incidents underscore the importance of robust threat modeling and risk assessment methods in the early stages of software development to prevent security vulnerabilities. The approach proposed here advocates incorporating cybersecurity measures from the early stages of the design process through systematic threat modeling and attack impact analysis. Our methodology serves as a foundation for understanding attacker behavior and facilitating early prediction of cyber-threats throughout the system's development life cycle.

Through a case study in the automotive industry, we have illustrated the adaptability and effectiveness of our proposed methodology and demonstrated its tangible application to strengthen system security. The results derived from our approach can be used to improve system design, eliminate vulnerabilities, and improve the overall security posture.

As this study presents a novel approach to improving cybersecurity through data flow-based attack scenario analysis, we recognize the need for quantitative validation of our method. Future work will focus on empirical studies to quantify the benefits of our approach. This will include the development of metrics for comparison with existing methods, such as the number of threats identified, false positive/negative rates, and the time efficiency of the analysis process. Such studies will provide statistical evidence to complement the qualitative benefits presented in this paper.

Our study encourages a paradigm shift in cybersecurity practices and challenges stakeholders to dispel misconceptions among developers and proactively address security concerns during software development. As cyber-threats become more frequent and sophisticated, our approach provides a valuable API for organizations and industrial entities to improve resilience against cyberattacks, increase preparedness, and gain a comprehensive understanding of potential system vulnerabilities. Our research approach not only facilitates the investigation and analysis of cyberattacks during the design phase but also enables us to propose efficient solutions for reducing or eliminating risks after the design phase is complete. The practical applications of our methodology are manifold, including its incorporation into cybersecurity frameworks for critical infrastructure protection and its use in cybersecurity education and training programs to improve threat awareness and mitigation strategies.

Last but not least, our proposed approach leverages the DFD as a crucial input to the STRIDE threat modeling process. We recognize the importance of automating or semi-automating the generation of the DFD as a way to enhance the scalability and applicability of the proposed approach for more complex and larger-scale cyber–physical systems. We plan to explore various methods for achieving this in our future work. These might include leveraging existing system modeling techniques such as model-driven engineering or architecture description languages to automatically extract the necessary information and construct the DFD. In addition, we aim to investigate the integration of our approach with common CPS design and engineering software tools to facilitate the seamless import of DFD data and reduce the amount of manual effort required. This would help ensure the reliability and consistency of the threat intelligent process, ultimately leading to more comprehensive and accurate attack impact analysis.

Finally, we intend to apply our approach to different industrial domains and explore new dimensions of cybersecurity. The continuous refinement and validation of our methodology in real-world applications and evolving threat landscapes is crucial. We plan to expand the application of our security-by-design approach to various systems, including industrial control systems and IoT devices, while conducting comparative studies with other intrusion detection methods. This comprehensive and iterative strategy positions our research at the forefront of advancing cybersecurity practices in today's dynamic digital landscape.

Author Contributions: Conceptualization, S.A.A.; methodology, S.A.A.; software, S.I. and A.D.-G.; validation, S.A.A.; formal analysis, S.A.A.; Investigation, S.A.A., S.I. and A.D.-G.; resources, S.A.A.; data curation, S.A.A., S.I., and A.D.-G.; writing—original draft preparation, S.A.A. and S.I.; writing—review and editing, S.A.A., S.I., A.D.-G., M.S. (Mikael Sjödin) and M.S. (Marjan Sirjani); visualization, S.A.A.; supervision, S.A.A.; project administration, S.A.A.; funding acquisition, M.S. (Mikael Sjödin) and M.S. (Marjan Sirjani). All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Swedish Foundation for Strategic Research through the Serendipity project and KKS SACSys Synergy project (Safe and Secure Adaptive Collaborative Systems, Grant No. 20190021)

Data Availability Statement: The datasets generated and source code implemented during and/or analyzed in the current study are available at the following link, which is also referenced in the paper: https://t.ly/Rdfvr (accessed on 30 September 2024).

Conflicts of Interest: The authors declare no conflicts of interest; the funders had no role in the design of the study, in the collection, analysis, or interpretation of data, in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

API	Application Programming Interfaces
AGVs	Automated Guided Vehicles
CSRF	Cross-Site Request Forgery
CPS	Cyber–Physical System
DFD	Data Flow Diagram
GDPR	General Data Protection Regulation
IT	Information Technology
IoT	Internet of Things
PETs	Privacy-Enhancing Technologies

References

1. Habib, M.Y.; Qureshi, H.A.; Khan, S.A.; Mansoor, Z.; Chishti, A.R. Cybersecurity and Smart Cities: Current Status and Future. In Proceedings of the 2023 IEEE International Conference on Emerging Trends in Engineering, Sciences and Technology (ICES&T), Bahawalpur, Pakistan, 9–11 January 2023; pp. 1–7.
2. Hora, A.; Kulkarni, P. Wearables and Cybersecurity: Navigating the Threat Landscape. In Proceedings of the 2024 2nd International Conference on Sustainable Computing and Smart Systems (ICSCSS), Virtual, 10–12 July 2024; pp. 563–567.
3. Davis, R.; Keskin, O.F. Cyber Threat Modeling for Water and Wastewater Systems: Contextualizing STRIDE and DREAD with the Current Cyber Threat Landscape. In Proceedings of the 2024 Systems and Information Engineering Design Symposium (SIEDS), Charlottesville, VA, USA, 3 May 2024; pp. 301–306.
4. Sahay, R.; Estay, D.A.S.; Meng, W.; Jensen, C.; Barfod, M. A Comparative Risk Analysis on CyberShip System with STPA-Sec, STRIDE and CORAS. *arXiv* **2022**, arXiv:2212.10830. [CrossRef]
5. Khalil, S.M.; Bahsi, H.; Korõtko, T. Threat modeling of industrial control systems: A systematic literature review. *Comput. Secur.* **2024**, *136*, 103543. [CrossRef]
6. Song, I.; Jeon, S.; Kim, D.; Lee, M.G.; Seo, J.T. GENICS: A Framework for Generating Attack Scenarios for Cybersecurity Exercises on Industrial Control Systems. *Appl. Sci.* **2024**, *14*, 768. [CrossRef]
7. Font, J.A.; Jarauta, J.; Gesteira, R.; Palacios, R.; López, G. Threat models for vulnerability analysis of IoT devices for Manipulation of Demand attacks. In Proceedings of the 2023 JNIC Cybersecurity Conference (JNIC), Vigo, Spain, 21–23 June 2023; pp. 1–8.
8. Sadlek, L.; Čeleda, P.; Tovarňák, D. Identification of Attack Paths Using Kill Chain and Attack Graphs. In Proceedings of the NOMS 2022—2022 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 25–29 April 2022; pp. 1–6.

9. Pfleeger, C.P.; Pfleeger, S.L. *Analyzing Computer Security: A Threat/Vulnerability/Countermeasure Approach*; Prentice Hall Professional: Saddle River, NJ, USA, 2012.
10. Swiderski, F.; Snyder, W. *Threat Modeling*; Microsoft Press: Redmond, WA, USA, 2004.
11. LeBlanc, D.; Howard, M. *Writing Secure Code*; Pearson Education: Saddle River, NJ, USA, 2002.
12. Potter, B. Microsoft SDL Threat Modelling Tool. *Netw. Secur.* **2009**, *2009*, 15–18. [CrossRef]
13. Shostack, A. Experiences Threat Modeling at Microsoft. *MODSEC@ MoDELS* **2008**, *2008*, 35.
14. Da Silva, M.; Puys, M.; Thevenon, P.H.; Mocanu, S.; Nkawa, N. Automated ICS template for STRIDE Microsoft Threat Modeling Tool. In Proceedings of the 18th International Conference on Availability, Reliability and Security, Benevento, Italy 29 August–1 September 2023; pp. 1–7.
15. Volvogroup. Electric Site Research Project. Available on: https://www.volvoce.com/global/en/this-is-volvo-ce/what-we-believe-in/innovation/electric-site/ (accessed on 14 June 2024).
16. Baumgart, S.; Fröberg, J.; Punnekkat, S. How to Analyze the Safety of Concepts for a System-of-Systems? In Proceedings of the 2021 IEEE International Symposium on Systems Engineering (ISSE), Vienna, Austria, 13 September–13 October 2021; pp. 1–8.
17. Alexander, O.; Belisle, M.; Steele, J. *MITRE ATT&CK for Industrial Control Systems: Design and Philosophy*; The MITRE Corporation: Bedford, MA, USA, 2020; Volume 29.
18. UcedaVelez, T.; Morana, M.M. *Risk Centric Threat Modeling: Process for Attack Simulation and Threat Analysis*; John Wiley & Sons: Hoboken, NJ, USA, 2015.
19. Caralli, R.A.; Stevens, J.F.; Young, L.R.; Wilson, W.R. *Introducing Octave Allegro: Improving the Information Security Risk Assessment Process*; SEI Administrative Agent: Hansom AFB, MA, USA, 2007.
20. Wynn, J.; Whitmore, J.; Upton, G.; Spriggs, L.; McKinnon, D.; McInnes, R.; Graubart, R.; Clausen, L. *Threat Assessment and Remediation Analysis (Tara)*; MITRE Corporation: Bedford, MA, USA, 2014.
21. Freund, J.; Jones, J. *Measuring and Managing Information Risk: A FAIR Approach*; Butterworth-Heinemann: Oxford, UK, 2014.
22. White, G.B.; Sjelin, N. The NIST cybersecurity framework. In *Research Anthology on Business Aspects of Cybersecurity*; IGI Global: Hershey, PA, USA, 2022; pp. 39–55.
23. Howard, M.; Lipner, S. *The Security Development Lifecycle*; Microsoft Press: Redmond, WA, USA, 2006; Volume 8.
24. Asadollah, S.A. Cyberattacks: Modeling, Analysis, and Mitigation. In Proceedings of the 2022 6th International Conference on Computer, Software and Modeling (ICCSM), Rome, Italy, 21–23 July 2022; pp. 80–84.
25. Hagan, M.; Sezer, S.; McLaughlin, K. Reactive and Proactive Threat Detection and Prevention for the Internet of Things. In Proceedings of the 2019 32nd IEEE International System-on-Chip Conference (SOCC), Singapore, 3–6 September 2019; pp. 195–196.
26. Rajba, S.; Wieclaw, L.; Nikolaienko, S.; Vasiliu, Y. Methods of data protection for quantum secure communication system. In Proceedings of the 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Bucharest, Romania, 21–23 September 2017; Volume 1, pp. 134–137.
27. Mitchell, R.; Chen, R. Effect of intrusion detection and response on reliability of cyber physical systems. *IEEE Trans. Reliab.* **2013**, *62*, 199–210. [CrossRef]
28. Huang, J.; Seck, M.D.; Gheorghe, A. Towards trustworthy smart cyber-physical-social systems in the era of internet of things. In Proceedings of the 2016 11th System of Systems Engineering Conference (SoSE), Kongsberg, Norway, 12–16 June 2016; pp. 1–6.
29. Arivarasi, A.; Ramesh, P. Review of source location security protection using trust authentication schema. In Proceedings of the 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 20–22 August 2020; pp. 215–222.
30. Bao, H.; Lu, R. A new differentially private data aggregation with fault tolerance for smart grid communications. *IEEE Internet Things J.* **2015**, *2*, 248–258. [CrossRef]
31. Roukounaki, A.; Efremidis, S.; Soldatos, J.; Neises, J.; Walloschke, T.; Kefalakis, N. Scalable and configurable end-to-end collection and analysis of IoT security data: Towards end-to-end security in IoT systems. In Proceedings of the 2019 Global IoT Summit (GIoTS), Aarhus, Denmark, 17–21 June 2019; pp. 1–6.
32. Grabovica, M.; Popić, S.; Pezer, D.; Knežević, V. Provided security measures of enabling technologies in Internet of Things (IoT): A survey. In Proceedings of the 2016 Zooming Innovation in Consumer Electronics International Conference (ZINC), Novi Sad, Serbia, 1–2 June 2016; pp. 28–31.
33. Grobler, M.; Gaire, R.; Nepal, S. User, usage and usability: Redefining human centric cyber security. *Front. Big Data* **2021**, *4*, 583723. [CrossRef]
34. Jirgl, M.; Bradac, Z.; Fiedler, P. Human-in-the-loop issue in context of the cyber-physical systems. *IFAC-PapersOnLine* **2018**, *51*, 225–230. [CrossRef]
35. Hassija, V.; Chamola, V.; Saxena, V.; Jain, D.; Goyal, P.; Sikdar, B. A survey on IoT security: Application areas, security threats, and solution architectures. *IEEE Access* **2019**, *7*, 82721–82743. [CrossRef]
36. Tidjon, L.N.; Frappier, M.; Mammar, A. Intrusion detection systems: A cross-domain overview. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 3639–3681. [CrossRef]
37. Cao, Q.H.; Khan, I.; Farahbakhsh, R.; Madhusudan, G.; Lee, G.M.; Crespi, N. A trust model for data sharing in smart cities. In Proceedings of the 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, 22–27 May 2016; pp. 1–7.

38. Treaster, M. A survey of fault-tolerance and fault-recovery techniques in parallel systems. *arXiv* **2005**, arXiv:cs/0501002.
39. Harte, R.; Glynn, L.; Rodríguez-Molinero, A.; Baker, P.M.; Scharf, T.; Quinlan, L.R.; ÓLaighin, G. A human-centered design methodology to enhance the usability, human factors, and user experience of connected health systems: A three-phase methodology. *JMIR Hum. Factors* **2017**, *4*, e5443. [CrossRef] [PubMed]
40. Nigam, V.; Talcott, C. Formal security verification of industry 4.0 applications. In Proceedings of the 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, 10–13 September 2019; pp. 1043–1050.
41. Clavel, M.; Durán, F.; Eker, S.; Lincoln, P.; Martí-Oliet, N.; Meseguer, J.; Talcott, C. *All About Maude-a High-Performance Logical Framework: How to Specify, Program, and Verify Systems in Rewriting Logic*; Springer: Berlin/Heidelberg, Germany, 2007; Volume 4350.
42. Deng, M.; Wuyts, K.; Scandariato, R.; Preneel, B.; Joosen, W. A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements. *Requir. Eng.* **2011**, *16*, 3–32. [CrossRef]
43. Kreitz, M. Security by design in software engineering. *ACM SIGSOFT Softw. Eng. Notes* **2019**, *44*, 23–23. [CrossRef]
44. Olayemi, O.; Antti, V.; Keijo, H.; Pekka, T. Security issues in smart homes and mobile health system: threat analysis, possible countermeasures and lessons learned. *Int. J. Inf. Technol. Secur.* **2017**, *9*, 31–52.
45. Xu, D.; Tu, M.; Sanford, M.; Thomas, L.; Woodraska, D.; Xu, W. Automated security test generation with formal threat models. *IEEE Trans. Dependable Secur. Comput.* **2012**, *9*, 526–540. [CrossRef]
46. Ebrahimi, M.; Striessnig, C.; Triginer, J.M.C.; Schmittner, C. Identification and Verification of Attack-Tree Threat Models in Connected Vehicles. *arXiv* **2022**, arXiv:2212.14435.
47. Konsta, A.M.; Lluch Lafuente, A.; Spiga, B.; Dragoni, N. Survey: Automatic generation of attack trees and attack graphs. *Comput. Secur.* **2024**, *137*, 103602. [CrossRef]
48. Das, P.; Asif, M.R.A.; Jahan, S.; Ahmed, K.; Bui, F.M.; Khondoker, R. STRIDE-Based Cybersecurity Threat Modeling, Risk Assessment and Treatment of an In-Vehicle Infotainment System. *Vehicles* **2024**, *6*, 1140–1163. [CrossRef]
49. Perata, J.P.; Betarte, G. A Security Analysis of a Referential Architecture of the FIWARE Platform. In Proceedings of the 2023 XLIX Latin American Computer Conference (CLEI), La Paz, Bolivia, 16–20 October 2023; pp. 1–9.
50. Yang, M.; Zhou, X.; Liu, D.; Zhou, L.; Tang, Y. Enhancing IoT Security: A Full-System Simulation Dynamic Taint Analysis Framework for Firmware. In Proceedings of the 2023 3rd International Conference on Electronic Information Engineering and Computer (EIECT), Changchun, China, 22–24 September 2023; pp. 381–388.
51. Zhang, Z.; Yang, Z.; Du, X.; Li, W.; Chen, X.; Sun, L. Tenant-Led Ciphertext Information Flow Control for Cloud Virtual Machines. *IEEE Access* **2021**, *9*, 15156–15169. [CrossRef]