POLITECNICO DI TORINO

Master's Degree in Computer Engineering

Master's Degree Thesis

Ontology-driven Threat Modeling for IoT Systems

Supervisors	Candidate
Prof. Alessandro SAVINO	Elena F. COMPIERCHIO
Dr. Nicolò MAUNERO	

December 2024

Abstract

The Internet of Things (IoT) is rapidly expanding, connecting billions of devices and offering new possibilities in many different contexts. However, it is also exposed to significant cybersecurity risks due to the unique characteristics of such interconnected and complex systems. Existing threat modeling approaches are often designed for traditional ICT environments and they struggle to address the complexity of IoT systems. This thesis proposes an ontology-driven framework to automate threat modeling for IoT systems, enabling more effective and efficient security assessments. This framework is built upon an ontology, developed using OWL 2 (Web Ontology Language) and Protégé. The ontology provides a formal representation of IoT systems, modelling their components, interactions, and potential threats. The ontology consists of three linked sub-ontologies. The IoT System sub-ontology is based on the ISO/IEC 30141 standard. This sub-ontology defines the physical and virtual components of an IoT infrastructure. It includes IoT devices, networks, data stores, services, and users. It aims to provide the foundation for understanding the system's architecture and security-related elements. The Data Flow sub-ontology models how information is exchanged within the IoT system. It describes communication paths, data exchanges, and trust boundaries, which are points where security risks might arise due to changes in privilege levels. The Threats sub-ontology, using the CAPEC and STRIDE frameworks, categorizes potential security threats specific to IoT. By mapping CAPEC attack patterns to STRIDE categories, the ontology connects detailed descriptions of specific attack techniques (from CAPEC) to broader categories of threats (from STRIDE). This mapping provides a more in-depth analysis of how different threats could impact the IoT system under analysis. To automate the process of identifying threats, the framework employs a set of inference rules expressed in SWRL (Semantic Web Rule Language). These rules examine the relationships and properties defined within the ontology to deduce potential threats based on the characteristics and interactions of system components. The use of inference rules enables automated reasoning, meaning that the framework can analyse the IoT system's ontology and identify potential threats without manual intervention. To evaluate the framework's effectiveness, it is applied to the HArMoNICS infrastructure, a digital replica of a

smart polygeneration microgrid. The case study demonstrates how the framework can automatically generate a detailed threat model, identifying relevant CAPEC entries and associated STRIDE categories linked to specific components and data flows in HArMoNICS. This evaluation underscores the framework's crucial role in real-world applications, demonstrating its capacity to significantly enhance the risk assessment process and drive the development of more effective mitigation strategies, ultimately strengthening IoT system security.

Acknowledgements

Desidero ringraziare il Professor Alessandro Savino,
relatore di questa tesi.

Un ringraziamento speciale va al Dottor Nicolò Maunero,
il cui contributo, incoraggiamento e disponibilità
sono stati fondamentali in ogni fase di questo lavoro
fino al suo completamento.

Infine, un sincero grazie a tutti coloro che, in modi diversi,
mi hanno motivata durante questo percorso,
offrendomi consigli e suggerimenti.

Table of Contents

List of Tables

List of Figures

Acronyms

AES Advanced Encryption Standard

AI Artificial Intelligence

AiTM Adversary in the Middle

API Application Programming Interface

CAPEC Common Attack Pattern Enumeration and Classification

CIA Confidentiality, Integrity, Availability

CM Conceptual Model

CVE Common Vulnerabilities and Exposures

CWE Common Weakness Enumeration

DTS Digital Twins Infrastructure

EPC Electronic Product Code

EDR Endpoint Detection and Response

HTTP Hypertext Transfer Protocol

ICT Information and Communication Technology

IEC International Electrotechnical Commission

IIoT Industrial Internet of Things

IoT Internet of Things

IoTSec Internet Of Things Security

IP Internet Protocol

ISO International Organization for Standardization

JSON JavaScript Object Notation

MFA Multi-Factor Authentication

MiTM Man in The Middle

NFC Near Filed Communication

NIST National Institute of Standards and Technology

NVD National Vulnerability Database

OS Operating System

OWL Web Ontology Language

OWASP Open Web Application Security Project

PAN Personal Area Network

PKI Public Key Infrastructure

RFID Radio Frequency Identification

RSA Rivest–Shamir–Adleman

SCD Sensing and Controlling Domain

SOfIoTS Secure Ontologies for Internet of Things Systems

SQL Structured Query Language

STRIDE Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service and Elevation of Privilege

SWRL Semantic Web Rule Language

TMT Threat Modeling Tool

Wi-Fi Wireless Fidelity

WSN Wireless Sensor Networks

XSS Cross-Site Scripting

XML Extensible Markup Language

Chapter 1

Introduction

Cybersecurity has become more critical than ever as the frequency and severity of attacks continue to increase [1]. In this scenario, cyber risk management plays an important role in protecting governments, organizations, companies and many other entities. Risk assessment is one of the key parts of the risk management process [2]. Risk assessment is the process of evaluating the risk related to cyberattacks and incidents. It can be divided into three main steps: (i) Threat Modeling, which has the objective of identifying threats targeting the system under analysis; (ii) Vulnerability Assessment, where weaknesses are examined; (iii) Penetration Testing, in which security experts try to exploit vulnerabilities that have been identified before. The output is the overall cyber risk level to which the system is exposed. The threat modeling step is the core of the process. Its effectiveness is crucial for the following two phases. However, threat modeling is typically a manual activity that requires experts to be performed, making it prone to errors. This is a critical aspect because missing a threat can compromise the security of the entire system. For this reason, approaches for threat modeling automation should be preferred. Automating the process can reduce the likelihood of errors and improve the overall efficiency.

Threat modeling can also be applied to IoT systems, but there are two critical aspects to consider. Firstly, it has been proven that traditional threat modeling approaches are inadequate when applied to IoT systems. The main reason is that many risk assessment methodologies have been developed before the introduction of IoT. Secondly, applying security to IoT is a complex task because of the peculiar characteristics of these systems. IoT systems are highly distributed, they are composed of many devices using different protocols to communicate over networks and their connections change very quickly. This complexity demonstrates the importance of automating the threat modeling process.

The Internet of Things is a paradigm in Information Technology, where "things" from everyday life are interconnected through the Internet. These devices are able to collect and share data, but also to perform actions without human intervention. One of the key concepts in IoT is that sensors and actuators are embedded in objects and linked through wired and wireless networks. The number of connected devices is expected to rapidly increase from 13.8 billion in 2022 to more than 32.1 billion in 2030 [3]. There are many contexts in which IoT can be implemented, from industries and healthcare to smart homes and cities.
The IoT creates a lot of opportunities, however, it also introduces many challenges. Most of these challenges are related to cybersecurity. IoT devices are often vulnerable to cyber threats and, for this reason, they should be protected. One key point that allows threat modeling automation is the standard representation of data in the threat modeling process.
However, the challenge for the IoT domain is the lack of a standardized architecture for IoT systems. Even though some models exist, none of them are considered a standard. This lack of standardization makes it difficult to implement security measures and to perform risk assessment. Without a faithful representation of the IoT system, security measures may be ineffective.

Starting from the challenges presented earlier, the objective of this thesis is to design and develop an ontology for IoT systems to support automated cybersecurity threat modeling. The ontology provides a formal representation of IoT system components, the relationships and interactions between them and the potential threats they may face. It represents the starting point for the threat modeling process and its automation.
The ontology is presented as three sub-ontologies: (i)The IoT System Sub-Ontology focuses on defining the physical and virtual components of a generic IoT environment according to ISO/IEC 30141 standard [4];(ii) The Data Flow Sub-Ontology captures the interactions, data exchanges and trust boundaries that exist between components of an IoT system; (iii) The Threats Sub-Ontology is designed to categorize security threats using existing knowledge-base such as CAPEC and STRIDE. Then, the inference rules for the automation of the threat modeling process are presented.

The chapters of this thesis are organized as follows: Chapter 2 provides background on IoT, cybersecurity concepts, and ontologies in general; Chapter 3 reviews the state of the art of Risk assessment approaches for ICT and IoT; Chapter 4 presents the proposed solution providing a description of the IoT ontology and the rules to support automated threat modeling; Chapter 5 provides a case study to validate the solution showing its application in real-world and discussing its limitations; Chapter 6 presents final considerations about the presented framework.

Chapter 2

Background

The development of an ontology for IoT systems to support automated cybersecurity threat modeling requires a deep understanding of existing concepts, methodologies, and tools.
This chapter provides the background to contextualize the solution presented in this thesis. We will first explore IoT and cybersecurity concepts separately. Then, we will focus on risk assessment and threat modeling. Next, we will examine the role of ontologies in cybersecurity, highlighting how they can be used to represent security-related information and assets. The chapter then introduces the frameworks utilized in this work: CAPEC (Common Attack Pattern Enumeration and Classification) and STRIDE, which provide standardized approaches for categorizing cyber threats and attacks.
By covering these topics, this chapter sets the foundation for the design and development of the ontology and threat modeling system presented in the following chapters.

2.1 Overview of IoT

The Internet of Things is a continuously evolving and permeating paradigm in IT (Information Technology). The phrase "Internet of Things" is also known as IoT and is coined from the two words. According to Nunberg [5], the "Internet" is a global system of interconnected computer networks that use the standard Internet protocol suite (TCP/IP) to serve billions of users worldwide. It is a network of networks that consists of millions of private, public, academic, business, and government networks that are linked by a wide array of electronic, wireless, and optical networking technologies. On the other hand, "Things" can be any object or person that can be distinguished in the real world. Everyday objects are not limited to electronic devices we use daily, but "things" that we do not normally

consider electronic at all. Some examples of "things" are food, clothing, furniture, materials, parts and equipment, merchandise, and specialized items [6].

There is more than one definition for the Internet of Things that is used by the community of people using it. The initial use of the expression has been attributed to Kevin Ashton, an expert in digital innovation. According to Madakam et al., [7], the best definition of the Internet of Things is the following: "An open and comprehensive network of intelligent objects that have the capacity to auto-organize, share information, data, and resources, reacting and acting in face of situations and changes in the environment". The IoT can also be considered as a global network that allows different types of communication: human-to-human, human-to-things, and things-to-things, which is anything in the world that has a unique identity [8].
One of the key concepts in IoT is that sensors and actuators are embedded in physical objects and linked through wired and wireless networks. The first appliance of IoT was a Coke machine at Carnegie Mellon University in the early 1980s, but the concept of IoT became very popular in 2003 through the Auto-ID center, and its related market analysts' publications [9].

According to Madakam et al. [7], there are some prerequisites for successfully implementing the Internet of Things (IoT):

1. Dynamic resource demand.
2. Real-time needs.
3. Exponential growth of demand.
4. Availability of applications.
5. Data protection and user privacy.
6. Efficient power consumption of applications.
7. Execution of the applications near to end users.
8. Access to an open and interoperable cloud system.

Other authors consider different prerequisites:

1. Hardware—composed of sensors, actuators, IP cameras, CCTV, and embedded communication hardware.
2. Middleware—on-demand storage and computing tools for data analytics with cloud and Big Data Analytics.

3. Presentation—easy to understand visualization and interpretation tools that can be designed for different applications.

As shown above, one of the main challenges with IoT is the lack of standardized, uniform architecture. Some models have been developed through the years, but there is no standard. In order to make the IoT system work, it should include sensors, actuators, networks, communications, and computing technologies, among others [10].

2.1.1 Common Technologies in IoT Systems

There are some technologies that are often included when talking about IoT, according to Madakam et al. [7]:

- **Radio Frequency Identification (RFID)** as explained by Doe et al. [11] is a system that transmits the identity of an object or person wirelessly using radio waves in the form of a serial number. The RFID technology plays an important role in IoT for solving identification issues of objects around us in a cost-effective manner [8]. The RFID technology is classified into three categories based on the power supply provision method in RFID tags: (i) Active RFID, (ii) Passive RFID, and (iii) Semi Passive RFID. Several components are important for this technology, such as tag, reader, antenna, access controller, software, and server. The main wireless applications of RFID are related to distribution, tracing, patient monitoring, military apps, and many others, as discussed by Moeinfar et al. [12].

- **Internet Protocol (IP)** is the most important network protocol used on the Internet, developed in the 1970s. There are two versions of this protocol that are both in use: IPV4 and IPV6. The main difference between the two versions is the way in which each of the two protocols defines an IP address. Because of its bigger diffusion, the generic IP address still refers to the addresses defined by IPv4. There are five classes of available IP ranges in IPv4: Class A, Class B, Class C, Class D, and Class E, while only A, B, and C are commonly used. The actual protocol provides for 4.3 billion IPv4 addresses while the IPv6 will significantly increase the availability to 85,000 trillion addresses [13].

- **Electronic Product Code (EPC)** is defined as a 64-bit or 98-bit code electronically recorded on an RFID tag and intended to design an improvement in the EPC barcode system. EPC codes can store information about the type of EPC, the unique serial number of the product, its specifications, manufacturer information, and many others. EPC was developed by the Auto-ID Centre at MIT in 1999.

It is composed of four components:

  - Object Naming Service (ONS).
  - EPC Discovery Service (EPCDS).
  - EPC Information Services (EPCIS).
  - EPC Security Services (EPCSS).

- **Barcode** is a different way of encoding numbers and letters by using a combination of bars and spaces of varying width. There are specific kinds of bar codes, such as Quick Response (QR) Codes, which are trademarks for a type of matrix bar code first designed for the automotive industry in Japan. Bar codes are optical machine-readable labels attached to items that record information related to the item. Recently, the QR Code system has become popular outside the automotive industry due to its fast readability and greater storage capacity compared to the standard. [7]. There are three types of bar codes :

  - Alpha Numeric
  - Numeric
  - Two dimensional

- **Wireless Fidelity (Wi-Fi)** is a networking technology that allows computers and other ICT devices to communicate over a wireless signal. The inventor of Wireless Fidelity is considered Vic Hayes. The first wireless products were brought on the market under the name WaveLAN with speeds of 1 Mbps to 2 Mbps. Today, there is nearly pervasive Wi-Fi that delivers high speed Wireless Local Area Network (WLAN) connectivity to millions of offices, homes, and public locations. The integration of Wi-Fi into notebooks, handhelds, and Consumer Electronics (CE) devices has accelerated the adoption of Wi-Fi to the point where it is a default in these devices, as Pahlavan et al. said[14]. Wi-Fi technology contains any type of WLAN product that can support any of the IEEE 802.11 standards.

- **Bluetooth** wireless technology is a short-range radio technology that eliminates the need for cabling between devices such as notebook PCs, handheld PCs, PDAs, cameras, and printers and has an effective range of 10 - 100 meters. Generally, communicate at less than 1 Mbps, and Bluetooth technologies use specifications of IEEE 802.15.1 standard. At first, in 1994, Ericson Mobile Communication company started a project named “Bluetooth”. It is used to create Personal Area Networks (PAN). A set of Bluetooth devices sharing a

common channel for communication is called Piconet. This Piconet is capable of 2 - 8 devices at a time for data sharing, and that data may be text, picture, video, and sound. The Bluetooth Special Interest Group comprises more than 1000 companies, including Intel, Cisco, HP, Aruba, Intel, Ericson, IBM, Motorola, and Toshiba. [7]

- **ZigBee** is one of the protocols developed for increasing the features of wireless sensor networks. ZigBee technology was created by the ZigBee Alliance, which was founded in 2001. Characteristics of ZigBee are low cost, low data rate, relatively short transmission range, scalability, reliability, and flexible protocol design. It is a low-power wireless network protocol based on the IEEE 802.15.4 standard [15].ZigBee has a range of around 100 meters and a bandwidth of 250 kbps. It is widely used in home automation, digital agriculture, industrial controls, medical monitoring, and power systems.

- **Near Filed Communication (NFC)** is a set of communication protocols that enables the communication between two electronic devices over a distance of 4 centimeters or less [16]. It helps consumers, making it simpler to make transactions, exchange digital content, and connect electronic devices with a touch. It was first developed by Philips and Sony. The data exchange rate nowadays is approximately 424 kbps.

- **Actuators** are components that produce force, torque, or displacement, usually in a controlled way, when an electrical, pneumatic, or hydraulic input is supplied to it in a system. In short terms, it converts energy into motion of a mechanical system. An Actuator can create different kinds of motion, such as linear motion, rotary motion, and oscillatory motion. Actuators cover short distances, typically up to 10 meters, and generally communicate at less than 1 Mbps. Actuators are typically used in manufacturing or industrial applications. There exist several types of actuators :

  – Electrical, e.g., motors and solenoids
  – Hydraulic that uses fluid to actuate motion
  – Pneumatic that take advantage of compressed air to actuate motion

  Each of them is used for a different scope. The most popular kind of actuator is the electric one which can be used in many situations.

- **Wireless Sensor Networks (WSN)** is a specific kind of network that has a distributed architecture, and it is composed of autonomous electronic devices (sensors). A sensor is a small device that is able to produce an output signal that takes a physical phenomenon as input. The objective of a WSN is to use a sensor to monitor physical or environmental conditions, such as temperature,

sound, vibration, pressure, motion, or pollutants, at different locations. A WSN is formed by hundreds or thousands of small devices that communicate with each other and pass data along from one to another. A wireless sensor network is an important element in the IoT paradigm. Due to the large amount, sensor nodes may not have global ID. WSN based on IoT has a very important role in many areas, such as military, homeland security, healthcare, precision agriculture monitoring, manufacturing, habitat monitoring, forest fire and flood detection, and so on [17]. Sensors can also be mounted to a patient's body to monitor the responses to the medication so that doctors can measure the effects of the medicines [18].

- **Artificial Intelligence (AI)** is the science of instilling intelligence in machines so that they are capable of doing tasks that typically require human intervention. AI-based systems are evolving rapidly in terms of application, adaptation, processing speed, and capabilities. Machines are increasingly becoming capable of taking on less-routine tasks[19]. AI-based systems are characterized by some common characteristics:
  - Embedded, which means that there are many devices connected to the network that interact with the system
  - Context-Aware, which means that there are devices that recognize the context and its change.
  - Personalized, which means that some devices can be used according to your needs
  - Adaptive, which means that a device can react after your action.
  - Anticipatory, which means that a device can predict your desires without any conscious meditation.

.

2.2 Overview of Cybersecurity

Cybersecurity has become a critical concern globally. Governments, organizations, and individuals recognize the need for a protective strategy against digital threats. Given its broad scope of application, there are several definitions of Cybersecurity. According to von Solms and van Niekerk [20], Cybersecurity is the collection of tools, policies, security, concepts, security safeguards, guidelines, risk management approaches, actions, training, best practices, assurance and technologies that can be used to protect the cyber environment, organization, and user's assets. Organization and user assets include connected computing devices, personnel, infrastructure,

applications, services, telecommunications systems, and the totality of transmitted and stored information in the cyber environment.
The most important concepts in Cybersecurity can be synthesized with the so-called *CIA Triad*, where CIA is the acronym for: Confidentiality, Integrity and Availability. According to this concept, the definition of Cybersecurity is similar to the definition of Information Security, but there are some important differences even if, sometimes, the two terms are considered interchangeable.
Information security deals with the protection of the actual technology-based systems on which information is commonly stored and/or transmitted. There are some international standards that define system security as all aspects relating to defining, achieving and maintaining the confidentiality, integrity, availability, non-repudiation, accountability, authenticity, and reliability of information resources.
The paper [20] focuses on the fact that the assets cybersecurity aims to protect include an additional dimension that extends beyond the usual boundaries of information security.
There are cyber security threats that are not considered as part of a formally defined range of threats to information security.
The following example will briefly present a scenario of interest, home automation. Advances in ICT and electronics have spurred significant growth in home automation applications. Some of these allow homeowners to integrate home security systems, hot water geysers, fridges, stoves, televisions, and other appliances with web-based management systems.
Unfortunately, the increased benefits of managing one's home via the web are accompanied by the increased risk that someone might gain unauthorized access to such systems and cause problems. This problem could vary between "pranks" such as turning off the hot water and crimes such as turning off the security system in order to break into the house. As previously said, in this case, it is possible to say that the victim's information is not necessarily harmed. Instead, other assets of the victim are the target of the cybercrime [21]. There are other scenarios such as cyberbullying, digital media, and cyber terrorism that involve humans as targets of attacks.
In the above scenarios, the compromising of information leads directly to an impact on the asset, in this case possibly a human, or society in general, as shown in Figure 2.1 from [20].

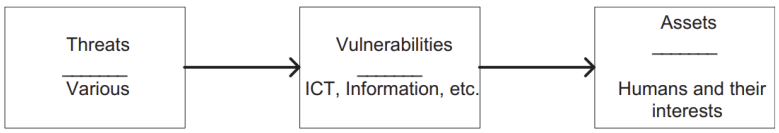

Figure 2.1: Cybersecurity Process

As demonstrated above, in cyber security the assets that need to be protected might be anywhere from the person him/herself to common household appliances, to the interests of society, including critical national infrastructure. In fact, such assets include absolutely anyone or anything that can be reached via cyberspace.

Finally, according to Von Solms et al. [20], Cybersecurity can be defined as the protection of cyberspace itself, the electronic information, the ICTs that support cyberspace, and the users of cyberspace in their personal, societal, and national capacity, including any of their interests, either tangible or intangible, that are vulnerable to attacks originating in cyberspace. Multiple approaches exist for strengthening system protection. The most used in the Cybersecurity context is Risk Assessment.

2.2.1 Risk Assessment

Risk Assessment is a process that consists of evaluating potential threats and vulnerabilities in an ICT infrastructure to determine the level of risk to which the system is exposed. Risks are first prioritized according to a combination of the identified severity and likelihood, then they should be mitigated based on the identified level of risk.
Generally, the risk assessment process is divided into three main steps: (i) threat modeling, (ii) vulnerability assessment, and (iii) penetration testing. Each of these stages focuses on different aspects of risk. The output of this evaluation process is a complete picture of the system's security status.

Let's define the three stages explaining their objectives:

- **Threat Modeling** is proposed as a solution for secure application development and system security evaluations. Its aim is to be more proactive and make it more difficult for attackers to accomplish their malicious intent. However, threat modeling is a domain that lacks common ground [22]. This stage is about identifying and categorizing potential threats that could target the system. There are several definitions of threat modeling according to the application domain. The objective of this step is to understand how an

attacker might compromise the system by analyzing attack vectors, potential adversaries, and the pathways they could exploit. Some advantages are common in every threat modeling methodology [23]: (1) when applied during the different stages of the system life cycle, from design to implementation, it allows threat ranking, prioritizing the most important ones and assuring resources are distributed effectively to develop and maintain adequate defenses; (2) applying threat modeling in an iterative way can assure proper mitigations be in place for newly discovered threats.

- **Vulnerability Assessment** can be defined as the process of identifying, quantifying, and prioritizing (or ranking) the vulnerabilities in all system components. Vulnerability assessment is an iterative process. A vulnerability can be a flaw or gap in the security that can be exploited by attackers. We can think of a vulnerability as an open door or a broken lock that allows someone to enter a building without permission. Vulnerabilities can be found in software, hardware, and network components. It involves evaluating configurations, outdated software, and known vulnerabilities. The objective is to build a comprehensive vulnerability list. There are tools such as Nessus, OpenVAS, or other scanning tools that are commonly used in this step to automate the detection of vulnerabilities.

- **Penetration Testing** is one strategy used to mitigate the risk of cyber-attack. Security experts attempt to compromise systems using the same tools and techniques as malicious attackers. The objective for the experts is to identify vulnerabilities before an attack occurs. Penetration testing is also known as ethical hacking, and it involves simulating real-world attacks to exploit the vulnerabilities identified during the previous stages of risk assessment. This phase helps to validate the work that has been done before. In penetration testing, it is possible to say if a vulnerability can be exploited or if it is a false positive and a security solution has already been applied. If the identified weaknesses can actually be exploited by an attacker, it is possible to understand the real impact of a successful exploit. Penetration testing exceeds theoretical analysis and focuses on practical attempts.

There are some traditional risk assessment frameworks, such as ISO/IEC 27005 and NIST SP 800-30, that provide guidelines for performing these analyses. These frameworks recommend an iterative cycle of risk assessment to adapt to changing threats and system conditions. The objective is to ensure that the security status remains robust over time.

2.3 Overview of Ontologies

In computer science and information systems, the term **Ontology** represents a formal representation of a set of concepts in a domain and the relationships between those concepts [24].
A generic ontology is composed of the following elements:

- **Classes** are categories of objects. A class is a collection of individuals (objects, things,...). A class can have subclasses. Classes and subclasses are structures as a hierarchy.
- **Individuals** are instances of a class. They are objects in the world. Individuals are related to other objects and to data values via properties.
- **Properties** represent a collection of relationships between individuals (and data) and explain how they are bounded together (has_father, has_ pet, service_number, ...) There are three different groups of properties:
  - Object Properties are relationships between two individuals
  - Data Properties are relationships between an individual and a data type (e.g., a string that provides additional information about the individual)
  - Annotation Properties are usually labels or comments used for meta-data

Ontologies help people in organizing information about a specific subject. They divide information into small parts such as things (entities), types of things (classes), properties (attributes), and examples (instances). This organized way makes it easier to share and analyze information.

As Noy and McGuinness explained [24], ontologies give everyone a common understanding of a subject that can be used by both people and computers. By creating a structure with relationships between different parts of information, ontologies help us reason about the information and manage it in a more complex way.
In the context of cybersecurity, ontologies represent a powerful tool for managing and reasoning about security-related information. Modern ICT infrastructures and the dynamic nature of cyber threats make it difficult to maintain an acceptable security level.
As described by De Rosa et al. 2022 [25], cybersecurity ontologies help in representing ICT infrastructure in a structured way. It is possible to use an ontology to represent assets, vulnerabilities, attack vectors, and mitigation strategies.
This knowledge base supports processes such as risk assessment, threat modeling, and other security operations. It is also possible to integrate into an ontology external security repositories such as CVE, CWE, and MITRE ATTaCK. Adding

this knowledge, we can perform automated reasoning, enabling security experts to identify dangerous dependencies and mitigate potential risks more efficiently.

In summary, using an ontology to describe an ICT infrastructure in a standard way is a valuable strategy. Moreover, this approach can be applied also to IoT infrastructures where characteristics of interconnected devices present new challenges and opportunities.

2.4 Used Frameworks

In the following sections will be presented the most important frameworks used to support the work of this thesis: CAPEC (Common Attack Pattern Enumeration and Classification) and STRIDE. Both frameworks play an important role in the development of an automated threat modeling framework for IoT systems. These frameworks provide a standard approach for identifying, categorizing and analyzing threats, which are essential for the threat modeling phase. These frameworks are widely recognized in the field of cybersecurity, but also relevant when integrated with ontological methods, as discussed in the previous section.

2.4.1 CAPEC

CAPEC was established by the U.S. Department of Homeland Security and initially released in 2007. The CAPEC List continues to be constantly updated by the community to form a standard mechanism for identifying, collecting, refining, and sharing attack patterns in the cybersecurity world.

The Common Attack Pattern Enumeration and Classification (CAPEC) provides a publicly available list of common attack patterns that help users understand how attackers exploit weaknesses in applications and other assets.
"Attack Patterns" are descriptions of the common approaches employed by attackers to exploit known weaknesses in assets of an ICT system. Attack patterns define the challenges that an adversary may face and how they can solve them. They derive from the concept of design patterns (used in software design) applied in a destructive context. These pattern are generated from the analysis of specific real-world exploit examples.
Each attack pattern captures knowledge about how specific parts of an attack are designed and executed. It also provides some information on ways to mitigate the attack's effectiveness. Attack patterns help those developing applications or managing cyber environments to better understand the specific elements of an attack and how to stop them from succeeding.

Some Well-Known Attack Patterns examples from the CAPEC list:

- HTTP Response Splitting (CAPEC-34)
- Session Fixation (CAPEC-61)
- Cross Site Request Forgery (CAPEC-62)
- SQL Injection (CAPEC-66)
- Cross-Site Scripting (CAPEC-63)
- Buffer Overflow (CAPEC-100)
- Clickjacking (CAPEC-103)
- Relative Path Traversal (CAPEC-139)
- XML Attribute Blowup (CAPEC-229)

There are several use cases in which CAPEC can be used. Among these use cases there is threat modeling [26].

2.4.2 STRIDE

STRIDE is a model for identifying security threats developed by Praerit Garg and Loren Kohnfelder at Microsoft [27].
This model is one of the most commonly used threat modeling methodologies as it provides crucial information to recognize threats and protect important system infrastructure, devices, and networks.
STRIDE is related to a Threat Modelling Tool (TMT) [28] to support analysts in the security assessment process. In particular, the tool tries to find issues in software design starting from Data Flow Diagrams; then, the tool extracts possible threats and proposes some mitigation for each of them, following the STRIDE model with its categories.

STRIDE model divides threats into six categories that are explained in Table 2.1 summarized from [27].

Table 2.1: STRIDE Threat Categories and Descriptions

Category	**Description**
Spoofing	Involves illegally accessing and then using another user's authentication information, such as username and password
Tampering	Involves the malicious modification of data. Examples include unauthorized changes made to persistent data, such as that held in a database, and the alteration of data as it flows between two computers over an open network, such as the Internet
Repudiation	Associated with users who deny performing an action without other parties having any way to prove otherwise—for example, a user performs an illegal operation in a system that lacks the ability to trace the prohibited operations. Non-Repudiation refers to the ability of a system to counter repudiation threats. For example, a user who purchases an item might have to sign for the item upon receipt. The vendor can then use the signed receipt as evidence that the user did receive the package
Information Disclosure	Involves the exposure of information to individuals who are not supposed to have access to it—for example, the ability of users to read a file that they were not granted access to, or the ability of an intruder to read data in transit between two computers
Denial of Service	Denial of Service (DoS) attacks deny service to valid users—for example, by making a Web server temporarily unavailable or unusable. You must protect against certain types of DoS threats simply to improve system availability and reliability
Elevation of Privilege	An unprivileged user gains privileged access and thereby has sufficient access to compromise or destroy the entire system. Elevation of privilege threats include those situations in which an attacker has effectively penetrated all system defenses and become part of the trusted system itself, a dangerous situation indeed

Chapter 3

State of the art

This thesis proposes an ontology-based framework to perform automated threat modeling for IoT systems; this chapter presents the current research and methodologies related to the thesis topics.
One of the most important aspects to understand strengths and limitations of existing approaches is reviewing the state of the art. This means to identify trends and understand what are the foundations upon which this thesis is built.
We will explore the current literature and the most relevant works. The aim is to understand in which area the contribution of this thesis could be more relevant.

In this chapter there are two main sections: (i) Risk Assessment Approaches in ICT and (ii) Risk Assessment Approaches in IoT. The first section is a review of risk assessment methodologies that use ontologies to model a generic ICT infrastructure. The second section focuses on the solutions related to risk assessment for IoT infrastructure. This review of the state of the art sets the basis for the following chapters.

3.1 Risk Assessment Approaches in ICT

Various methodologies have been developed for risk assessment in Information and Communication Technology (ICT) systems. Among the most relevant works are those that make use of ontological metamodels to model ICT systems and represent concepts related to cybersecurity. The use of ontologies for risk assessment allows to model complex relationships, enabling automatic reasoning to analyze threats and risks. By exploring these works, we can better understand how ontological models might be useful in more specialized domains, such as IoT.

The works presented by De Rosa et al. [25], [29] and Maunero et al. [30] provide a framework for cybersecurity governance of ICT systems focusing, respectively, on (i) the development of an ontology, (ii) threat modeling automation and (iii) risk assessment automation for ICT systems.
The aim of the paper by De Rosa et al. [25] is to provide an ontology that (i) supports a formal description of an ICT system, (ii) relates it to its potential vulnerabilities, possible attack vectors, and available mitigation, (iii) allows inferring a tight relationship between IT/OT assets and their vulnerabilities. Starting from the ICT system, the ontology is automatically populated with security information items obtained by external knowledge bases (e.g., CWE, CVE, MITRE ATTaCK) and then provides the user with the information to support operations such as Vulnerability Assessment and Penetration Testing.
The objective of this work is to provide a tool to support security operations that allows the analysis and management of related information:

- infrastructure composition in terms of assets, networks, functionality, and dependencies with external entities;
- information on system vulnerabilities;
- information on attacks and mitigation.

All these information items are organized in an ontology. The ontology is composed of three main parts, linked together by the relative relationships:

- ICT Ontology: It allows the description of the reference ICT infrastructure;
- Vulnerability Ontology: contains and organizes data on the vulnerability of the infrastructure. It is populated using external databases such as CVE, NVD, and CWE;
- Attack Ontology: contains and organizes data on possible attacks, how a vulnerability can be exploited, and possible mitigations. Populated using information coming from the CWE and MITRE ATTaCK databases [31].

The ontology has been described using the OWL 2 language and resorting to Protégé [32], a tool developed by Stanford University. The ICT Ontology describes the architecture of the infrastructure and the relationships with other entities and external services.
Figure 3.1 partially represents the structure of the ontology, highlighting the most important entities and relations. The vulnerability ontology is a trade-off between UCO and IoTSec ontologies, to allow a complete and easily integrated solution to represent vulnerabilities. The class Vulnerability is put in relation with the classes Asset and SecurityMechanism.

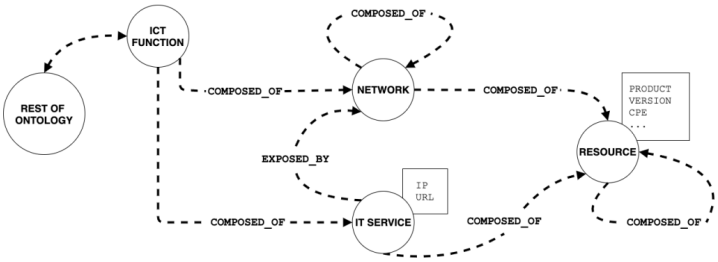

Figure 3.1: Partial Representation of the ICT Ontology

The attack ontology aims to provide information on how the vulnerability can be exploited by combining both technical information and cyber intelligence information describing how an attack can be carried out, the techniques used and so on, referring to the information provided by MITRE ATTaCK. In addition, the ontology also presents information on how a vulnerability can be mitigated.
Figure 3.2 from [25] provides a visual representation of both vulnerability and attack ontologies.

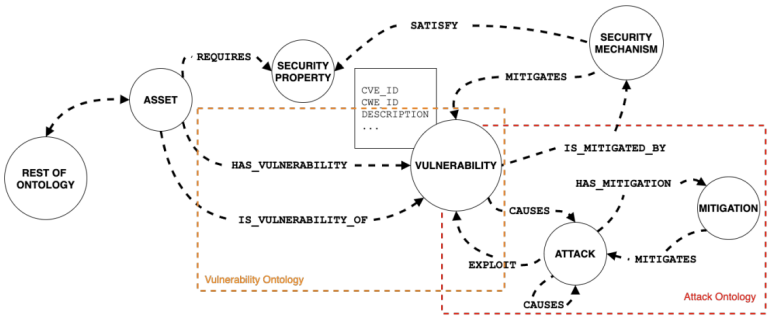

Figure 3.2: Vulnerability and Attack Ontology

The paper by De Rosa et al. [29] presents ThreMA, an ontology-driven threat modeling automation tool for ICT Infrastructures. This work starts from the paper, above mentioned, and provides a more detailed version of the initial ICT ontology. The focus is on automated threat modeling. This work provides a solution to the challenges related to threat modeling: (i) the need for a standard representation of models and data used in the process and (ii) the requirement for a well-defined inference rule set enabling reasoning process automation for threat identification.

The ThreMa metamodel is conceptually divided into three sub ontologies:

- ICT sub-ontology contains rules and vocabulary for modeling an ICT infrastructure;
- Data Flow sub-ontology is intended to represent the data flow diagram;
- Threat sub-ontology contains the characterization of threats.

These three parts are connected by means of relationships and the threat modeling logic is expressed using inference rules used by reasoners to map threats to infrastructure components.
The ThreMa architecture require two input: (i) the ICT infrastructure metamodel and (ii) the structural and behavioral descriptions of the target ICT infrastructure. Starting from these two inputs, ThreMA, by using the internal ontology reasoner, is able to automatically extract the threat model by applying the rules defined in the metamodel.
The output of the process is a security knowledge-base that contains all the information needed by a Security Architect to identify critical points in the infrastructure and to plan the implementation and adoption of mitigation.
Figure 3.3 presents the sequence diagram and use cases of the ThreMa architecture.

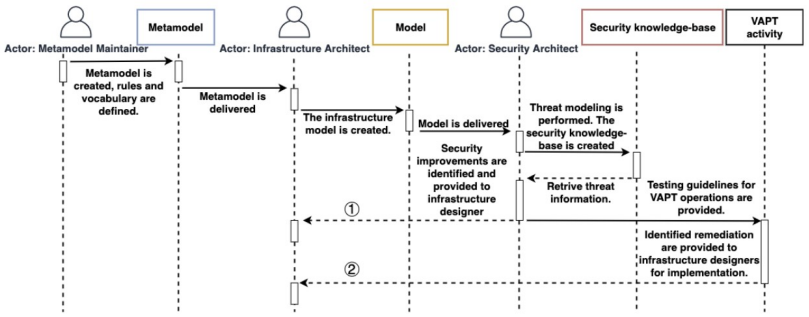

Figure 3.3: Sequence Diagram and Use Cases of ThreMa Architecture

The ThreMa logic is based on rules. Rules have been written using the Semantic Web Rule Language (SWRL) and represent the logic behind threat modelling automation. Automatic reasoners can apply the defined rules to the ICT infrastructure model to map components to corresponding threats.

The third paper by Maunero et al.[30] aims to automate risk assessment using an ontology-based approach. The solution proposed in this work builds on top of two previous works [25] [29]. From an architectural point of view, the solution

presented in this paper gets the following inputs: (i) the ontology for a generic ICT infrastructure, (ii) vulnerabilities information, (iii) structural and behavioral description of the target ICT infrastructure, and (iv) risk evaluation metrics for assessing the risk of identified threats.

The output is the following:

- Security knowledge-base for the target ICT infrastructure: the model containing the infrastructure component, architecture and the identified vulnerabilities and threats.
- Risk assessment: the risk evaluation of identified threats.

Starting from the threats identified in the analyzed infrastructure, assign each of these threats a risk value, for instance, an assessment of the probability and consequences of its occurrence. So, for each of the threats identified for infrastructure components (threat-component pairs) are extracted likelihood and impact from the CAPEC database, these values are represented by MITRE on a scale very low, low, medium, high, very high and are, hence, translated to numerical value in the range 1 to 5.

The work presented in this thesis builds on these three papers. The aim is to enhance and expand the previous works extending the approach to IoT environments.

3.2 Risk Assessment Approaches in IoT

Traditional approaches for risk assessment in ICT environments try to identify critical assets, the threats they face, the likelihood of a successful attack, and the problems that might result. The challenge with the Internet of Things is that risk assessment methodologies were created before its development. The complexity of IoT systems introduces issues that traditional risk assessment methods struggle to address.
In this section, we explore several approaches to perform risk assessment in IoT environments. By examining the most relevant works in this area, the objective is to understand what are the peculiar characteristics of IoT environments and what is the best way of assessing the risks.

The paper by Casola et al. [33] explores the development of an automated threat modeling approach specifically for edge computing systems. Edge computing brings processing and storage capabilities closer to the data sources, to reduce network latency, save bandwidth, and preserve data locality. However cloud-computing paradigm brings new cyber risks due to the combination of the security issues and

challenges of the cloud and Internet of Things (IoT) worlds.
A typical cloud system consists of three layers, namely (i) the cloud service layer, (ii) the edge layer, and (iii) the (IoT) device layer. Considering these layers the authors consider three main asset types: (i)Physical/Virtual Processing Nodes that include the processing nodes belonging to the different layers of an edge computing system, devoted to running application programs and services; (ii)Communication Channels among the nodes; (iii)Software Components such as modules and services that help implement the business logic of the system.
The authors considered also a data classification based on the element to which the data are related:(i) User-related data, (ii) Environmental data, (iii) Service data.
Starting from this information a developer should describe the system under analysis by identifying the involved assets and data, according to the system model. After this step, a comprehensive threat model of the system and a set of countermeasures to apply in terms of security controls will be provided.

The article by Nurse, Creese and De Roure (2017) [34] analyze new methodologies for assessing risks in the context of the Internet of Things (IoT). The authors analyze reasons why current risk assessment approaches are unsuitable for the IoT and highlight the need for new approaches.
The article explains that there are several core concepts in traditional risk assessment, such as assets, vulnerabilities, threats, attack, likelihood, and impact or cyber harm. Each of these concepts has its definition and properties. These approaches for risk assessment are, typically, asset-oriented or threat-oriented. In the first case, the assessment is centered on critical assets rather than ephemeral threats. On the other hand, the threat-oriented approach tends to focus on current threats.
However, there are some relevant IoT dynamics that must be considered in order to understand why traditional approaches may not be effective:

- Shortcomings of periodic assessment: traditional risk assessment approaches are based on periodic assessment and assume that systems will not significantly change in a short time period. These assumptions are not valid for the IoT. IoT has vast variability in system scale, dynamism, and coupling.

- Changing system boundaries, yet limited systems knowledge: existing risk assessments require reasonable domain knowledge (on assets, threats, attack probabilities, potential impacts, and so on). The same knowledge is extremely hard to get in IoT systems.

- The challenge of understanding the glue: traditional risk assessment focus on known assets; the problems of this focus in IoT is that the processes through which devices are bound, the connections that allow them to couple and operate, and the inner workings of the actors themselves are not considered.

- Failure to consider assets as an attack platform: In current risk assessment approaches, assets are considered as valuable things to the organization. The reality, especially as it relates to the IoT, is that assets (such as IoT devices) can be the basis for attacks.

The article, then, underline the need for new approaches including automated and continuous risk assessment as well as the development of new support tools to assist with simulation and modeling that can enhance predictive powers.

The paper by Mozzaquatro et al. [35] focuses on the development of an ontological framework designed for enhancing cybersecurity process in IoT environments. According to the authors, the heterogeneous connectivity of IoT systems increases the task for security experts since it involves security provisioning services to billions of smart objects.
Another challenge within the IoT ecosystem involves the lack of knowledge of the basic elements of cybersecurity: assets, threats, security mechanisms, vulnerabilities and security properties. Different IoT systems require distinct security mechanisms to avoid intrusions.
Mozzaquatro et al. claim that "If knowledge about known cybersecurity issues (e.g., vulnerabilities, known threats), and the corresponding prevention measures could be integrated in a comprehensive ontology that is accessible to run time monitoring and actuation tools, then security systems could be improved to automatically detect threats to the IoT network and dynamically propose or implement suitable protection services."
To verify their hypothesis the authors developed an ontology-based cybersecurity framework that present a new approach to improve the security of IoT systems focusing on company point of view. According to the author of this paper, the framework analyzes and classifies vulnerabilities in a knowledge base. Then, for each of them, provides security services that mitigate the specific threat. The objective is to improve security mechanisms around business processes and technology assets. The framework proposed in this paper is composed of three layers that deal with cybersecurity at design and run time. The third layer is called "integration layer".

The approach proposed by Casola et al. [36] aimed at supporting the security analysis of an IoT system by means of an almost completely automated process for threat modeling and risk assessment, which also helps identify the security controls to implement in order to mitigate existing security risks.
In this work, both the architectural components of an IoT system and its security properties are represented. This helps with identification of possible threats,

analysis and evaluation of security risks, and selection of countermeasures to mitigate those risks. Starting from the model of the IoT system, the threat model is automatically built using information stored in a security knowledge base that maps threats to assets, countermeasures, and other information. Then threats are associated with a risk level, computed according to the OWASP Risk Rating Methodology.
Suitable countermeasures are then mapped to security controls which are introduced in the original IoT System. This last step helps mitigate the existing risks. The proposed methodology uses a modeling approach aligned with ISO standards to build a semi-automated threat model for specific IoT deployments. Also, security countermeasures are specified in terms of security controls, defined according to the NIST Security Control Framework.
The adoption of standards allows the methodology to achieve a minimal security level even when applied by people that are not security experts in a company.
In the paper it is also presented the MicroBees case study which illustrates an home automation system with components that interact via radio using a custom protocol, coordinated by a gateway using cloud services. This case study highlights some challenging aspects in IoT systems such as the involvement of non-skilled technicians for installation and the system's operation within a home network.

The paper from Kandasamy et al. [37] provides a critical analysis of the cybersecurity risks associated with Internet of Things (IoT) systems. The paper focuses on the limitations of existing risk assessment frameworks. Then, proposes new risk assessment methodologies for IoT environments, in particular for high-risk sectors such as healthcare and financial technology.
The authors of the paper claim that common IoT vulnerabilities arise due to the following factors: (a) complex architecture, (b) inappropriate security configuration, (c) physical security, and (d) insecure firmware or software. Physical security is one of the main vulnerabilities that has been exploited in IoT devices.
The authors said that securing IoT systems involves solving many complex technology-related issues. Also, a recent IoT security research literature [38] discusses the existing authentication, access control methods, and trust management techniques and recommends that IoT threat modeling could be used for the IoT risk mitigation process.
Starting from this knowledge the authors analyze cyber risk assessment frameworks, risk vectors, and risk ranking. Then, based on the literature review and analysis, a scientific approach to computing the cyber risk for IoT systems has been designed by the authors as a part of this research, taking into consideration the IoT-specific impact factors.

The authors identified three main categories of IoT risk:

- Ethical IoT risk: "This refers to the unforeseen adverse effects of unethical actions using IoT devices."
- Security and privacy IoT risk: "This refers to the exploitation of vulnerabilities in the system to gain access to assets with intent to causing harm."
- Technical IoT risk: "This is due to hardware or software failure because of poor design, evaluation, etc."

The paper provides also a list of vulnerabilities targeting IoT devices like sensors, smart devices, and wearable devices. Some of these vulnerabilities are: (a) CIA (confidentiality, integrity, and availability) triad is compromised if the network services are not secure enough on the IoT devices; (b) device and its related components are compromised if the web, API, and cloud are not secured; (c) lack of firmware validation on a device can lead to CIA triad violation and non-compliance; (d) use of insecure OS platforms and the use of components from a compromised supply chain could allow the device to be compromised; and (e) lack of hardening of devices (hardening is the process of securing a system by reducing its surface of vulnerability) lead to vulnerabilities. Then, the paper discuss about pros and cons of popular Risk Assessment Process Framework like NIST, ISO/IEC, and OCTAVE.

The paper by Jarwar et al. (2022) [39] focuses on the Industrial Internet of Things (IIoT) which offers significant benefits for improving industrial operations. The authors underline that the key difference between IoT and IIoT is the service functionality requirements at the service layer.
This paper presents initial results from the PETRAS Secure Ontologies for Internet of Things Systems (SOfIoTS) project, focusing on a base security ontology for IIoT systems that supports security knowledge representation and analysis.
The authors identified some reason that makes the security of IIoT so challenging. Firstly, it is unsafe to perform security audits or apply security solutions on live industrial systems. Secondly, most security solutions require resources such as memory, processing power, which is limited in IoT/IIoT device technologies. Also, modeling for IIoT systems is even more difficult than IoT applications because IIoT consist of a large number of heterogeneous devices which are often distributed across a multiple and remote geographical location.
According to the authors, ontological methods are one of the recognized and acceptable approaches for structuring the knowledge of such complex environments. The paper presents the SOfIoTS project which aims to develop a base security ontology that provides sufficient metadata and allows the creation of subclasses and relationships to model the security of IIoT and/or their Digital Twins Infrastructure (DTS).

Chapter 4

Designed Solution

This chapter presents the design and implementation of an ontology for automated threat modeling in IoT environments. As we have seen in Chapter 3, traditional approaches for risk assessment are not suitable for the IoT environment. The unique characteristics of IoT systems make it necessary to develop specific approaches for such systems. By using ontologies, this thesis presents a framework that integrates several components for threat modeling in IoT.

The work presented in this chapter is built upon papers presented in Section 3.1 with the aim of extending their application to IoT systems.

4.1 Design of the Sub-Ontologies

An ontology-based approach for threat modeling is effective because ontologies accurately represent the different aspects of an IoT system. In this section, the design of the key sub-ontologies will be presented.

These sub-ontologies are designed to model the components of IoT systems, with their relationships, interactions and potential threats:

- The IoTSystem Sub-Ontology focuses on defining the physical and virtual components of a generic IoT environment according to the ISO/IEC 30141 standard [4];
- The Data Flow Sub-Ontology captures the interactions, data exchanges and trust boundaries that exist between components of an IoT system;
- The Threats Sub-Ontology is designed to categorize security threats using existing knowledge bases such as CAPEC and STRIDE.

Sub-ontologies will be presented individually for clarity. However, they represent three sub-parts of a single ontology. Figure 4.1 shows a simplified diagram of the ontology.

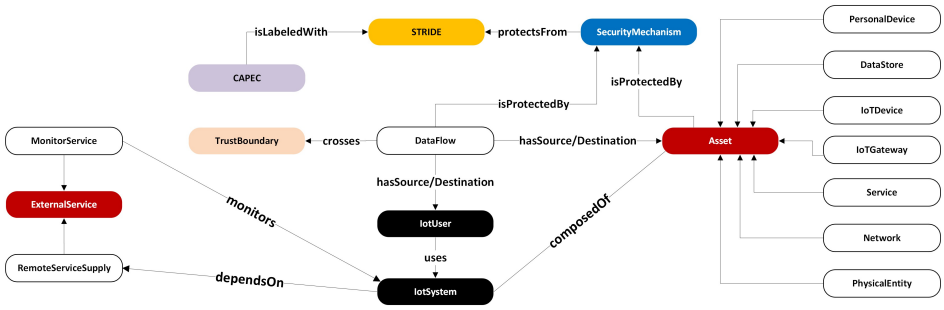

Figure 4.1: Simplified Ontology

4.1.1 IoT System Sub-Ontology

The IoT System Sub-Ontology is based on the IoT Conceptual Model defined by the ISO/IEC 30141 standard [4]. The Conceptual Model defined by the ISO/IEC 30141 standard is partially illustrated in Figure 4.2.
This international standard provides a standardized reference architecture for IoT systems. According to this standard, there are a number of possible application areas for IoT, such as smart city, smart grid, smart home, digital agriculture, smart manufacturing, intelligent transport system, e-health.
Moreover, IoT is an enabling technology that consists of many supporting technologies, for example, different types of communication networking technologies, information technologies, sensing and control technologies, software technologies, device/hardware technologies. The standard outlines the core concepts that characterize IoT environments, including the connection between Physical Entities ("things") with IT systems through networks, sensors that collect information about the physical world, while actuators can act upon Physical Entities. By aligning the IoTSystem Sub-Ontology with the standard architecture, the presented ontology captures the elements of IoT systems while remaining adaptable to various applications and contexts.
However, the ISO/IEC 30141 architecture does not model the security components that are needed for threat-modeling process. Security is a critical aspect in the context of IoT. IoT systems are increasingly exposed to cyber threats due to the highly interconnected nature of IoT devices.
To manage this lack, the security components have been modeled based on the work presented in the ThreMA ontology by De Rosa et al.[29].

This extension allows to add classes and relationships that capture security mechanisms and threats, which are important for supporting automated threat modeling. The level of abstraction chosen for the IoTSystem Sub-Ontology is high enough to adapt the general classes defined by ISO/IEC 30141, while still allowing for the detailed representation of IoT components, such as sensors, actuators, devices, gateways and networks.

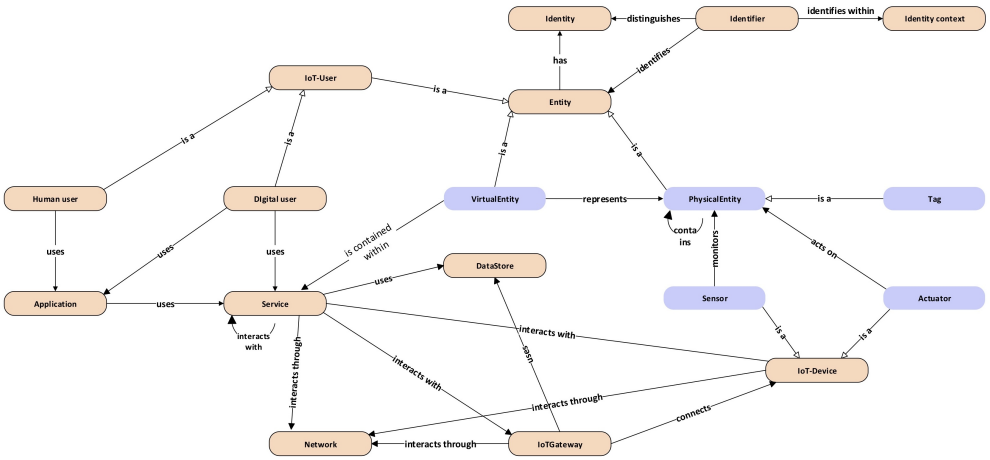

Figure 4.2: IoT CM from ISO/IEC 30141

In the following subsection, we present the structure of the IoTSystem Sub-Ontology, focusing on the key classes and relationships that define the core aspects of an IoT system.

Classes and relationships

In this section, the main classes and relationships in the IoTSystem Sub-Ontology will be presented. Key entities include IOTSystem, IOTUser, Asset and Service, each with specific properties and interactions.
The following is the list of main classes and relationships in the IoTSystem Sub-Ontology.

- **IOTSystem:** an IoT system is a system involving devices that bridge between real-world Physical Entities and Digital Entities, interacting with those Digital Entities via one or more networks over a wide area [4].
  - composedOf: Asset
  - dependsOn: ExternalService

- **IOTUser**: a user of an IoT system, which can be human or non-human [4].
  - uses:SmartDevice
  - usesExtService : ExternalService
  - usesService1: Service
- **Asset:** it is any valuable component of an IoT system that requires protection from potential threats. Assets can be physical, such as devices and infrastructure, or non-physical, like data and software.
  - isProtectedByS: SecurityMechanism
  - isProtectedBy: SecurityService
  - isAffectedBy : CAPEC
  - hasThreat: STRIDE
  - Subclasses: SmartDevice DataSore PhysicalEntity IoTDevice IoTGateway Network Service
- **SmartDevice**: a personal device is a device such as a smartphone, a tablet or a pc designed to help IoT users perform some tasks, or to handle particular types of IT problems.
  - usesService: Service
  - interactsThroughSD: Network
  - isA: Asset
- **Service**: a service is a set of distinct capabilities provided through a defined interface. A service can be composed of other services. A service is typically implemented as software [4].
  - Subclasses: SecurityService InternalService PubliclyAccessibleService
  - usesDataStore: Data store
  - interactsThrough: Network
  - interactsWith: IoTGateway
  - interactsWith: IoTDevice
  - composedOf: Service
  - isA: Asset

- **InternalService**: it is a software component that provides a service to the internal users of an IoT system, typically it is not accessible from the outside.
  - isA: Service
- **PubliclyAccessibleService**: it is a software component that provides a service to the external users of an IoT System, it is accessible from the outside, but also from the inside.
  - isA: Service
- **SecurityService**: is a component that represents security hardware or software device, such as firewalls or an IDS (Intrusion Detection System). It is related to the entity STRIDE by the relation protectsFrom, which represents the STRIDE threat category the service mitigates [29].
  - isA: Service
  - protectsFrom: STRIDE
- **DataStore**: data stores hold data relating to IoT systems, which can be data directly derived from IoT devices or can be data resulting from services acting on IoT device data [4].
  - isA: Asset
- **Network**: a network is an infrastructure that connects a set of Digital Entities, enabling communication of data between them [4].
  - isA: Asset
- **IOTGateway**: IoT gateways are devices which connect Sensing and Controlling Domain (SCD) with other domains. IoT gateways provide functions such as protocol conversion, address mapping, data processing, information fusion, certification, and equipment management [4].
  - interactsThroughGW: Network
  - usesDataStore1: DataStore
  - connects: IoTDevice
  - connectsD: SmartDevice
  - isA: Asset

- **IOTDevice**: an IoT device is a Digital Entity which bridges between real-world Physical Entities and the other Digital Entities of an IoT system through sensing and actuating capabilities [4].
  - interactsThroughIoT: Network
  - isA: Asset
  - Subclasses: Sensor / Actuator

  Figure 4.3 shows how main assets are interconnected inside the IoTSystem Sub-Ontology.

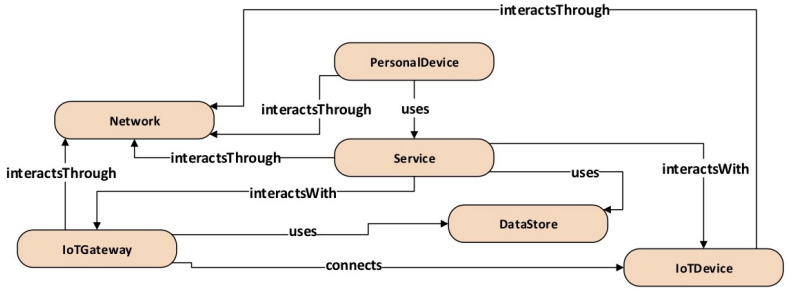

Figure 4.3: Relationships between important assets in IoTSystem Sub-Ontology

- **Sensor**: a Sensor is a specialized IoT device that measures some property of a Physical Entity and outputs digital data representing the measurement that can be transmitted over a network[4].
  - monitors: PhysicalEntity
  - isA: IoTDevice
- **Actuator**: an Actuator accepts digital inputs and performs actions that influence the physical environment. Actuators are specialized IoT devices [4].
  - actsOn: PhysicalEntity
  - isA: IoTDevice
- **PhysicalEntity**: an observable part of the physical environment [4].
  - composedOf: PhysicalEntity
  - isA: Entity

Some meaningful relationships between entities and IoT devices are shown in Figure 4.4.

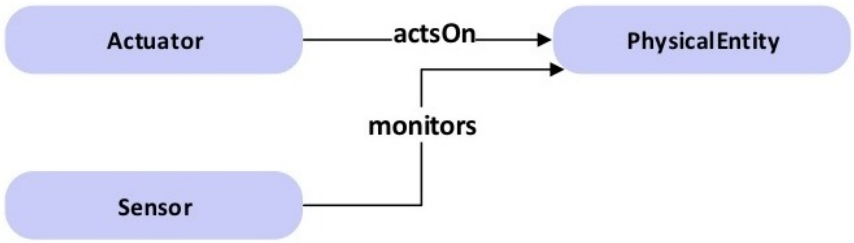

Figure 4.4: Relationships between IoT Devices

- **SecurityMechanism**: this class is used to model security solutions used to protect the specific infrastructure component. It serves two main purposes: (i) describing which security mechanisms are active to understand which threats are mitigated by what, but also (ii) identifying possible threats specific to these type of solutions [29]. These mechanisms are grouped into four categories: EncryptionAlgorithm, CryptographicConcept, SecurityManagementSystem and AuthenticationMethod. Each category addresses specific aspects of security, ensuring that different types of threats, such as those categorized by STRIDE, are mitigated effectively.
  - Subclasses: AuthenticationMethod / CryptographicConcept / EncryptionAlgorithm / SecurityManagementSystem
  - protectsFrom1: STRIDE
- **AuthenticationMethod**: it covers authentication protocols and mechanism. They are used to verify the identity of users and devices within the system. These include traditional methods like passwords and multi-factor authentication (MFA), biometric verification and public key infrastructure (PKI).
  - isA: SecurityMechanism
- **CryptographicConcept**: this class represents all those cryptographic concepts that are not encryption algorithms.
  - isA: SecurityMechanism

- **EncryptionAlgorithm**: it refers to the encryption techniques used to transform data into a secure format, ensuring confidentiality and preventing unauthorized access. Common encryption algorithms include symmetric methods like AES (Advanced Encryption Standard) and asymmetric methods like RSA (Rivest–Shamir–Adleman).
  - isA: SecurityMechanism
- **SecurityManagementSystem**: it represents security solutions like Endpoint Detection and Response (EDR) and monitoring systems [29].
  - isA: SecurityMechanism
- **STRIDE**: STRIDE is a widely adopted threat modeling methodology categorizing threats into six main types: spoofing, tampering, repudiation, information disclosure, denial of service and elevation of privilege. This methodology is essential in analyzing and securing IoT systems. The STRIDE entity serves two purposes: (i) threats are associated with the STRIDE category they belong to, and (ii) security mechanisms adopted in the infrastructure are associated with the STRIDE category they mitigate [29].
  - Subclasses: DenialOfService ElevationOfPrivilege InformationDisclosure Repudiation Spoofing Tampering

Figure 4.5 shows the interaction between the IoT system, its assets and security mechanisms.

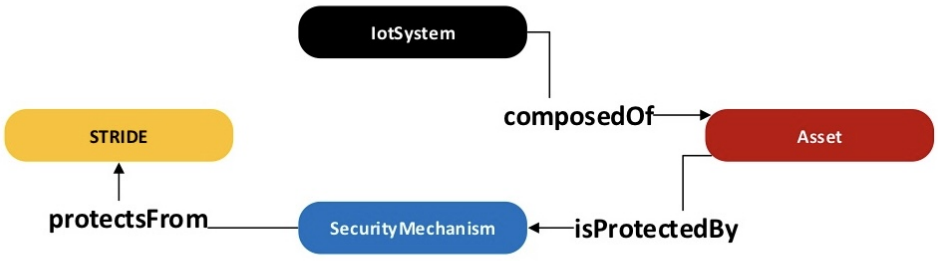

Figure 4.5: High Level Relationships in IoTSystemm Sub-Ontology

- **ExternalService**: the ExternalService class, as the one in ThreMa [29], is used to model the supply chain and dependencies from external providers and suppliers. It is related to ICTEntity through two relationships supply and dependsOn. These services can involve hardware, software, or remotely hosted digital services.
  - Subclasses: MonitorService/ RemoteServiceSupply
  - interactsThroughExt: Network
  - extIsAffectedBy : CAPEC
  - hasThreatE: STRIDE
- **MonitorService**: this class refers to the services that are hosted externally, but are used to monitor the IoTSystem from an external point of view.
  - isA: ExternalService
  - monitors1: IoTSystem
- **RemoteServiceSupply**: represents externally hosted digital services, such as web services or, in general, something-as-a-service used, in some way, by the IoTSystem [29].
  - isA: ExternalService

Figure 4.6 provides a visual representation of the classes and subclasses in the IoTSystem Sub-Ontology.

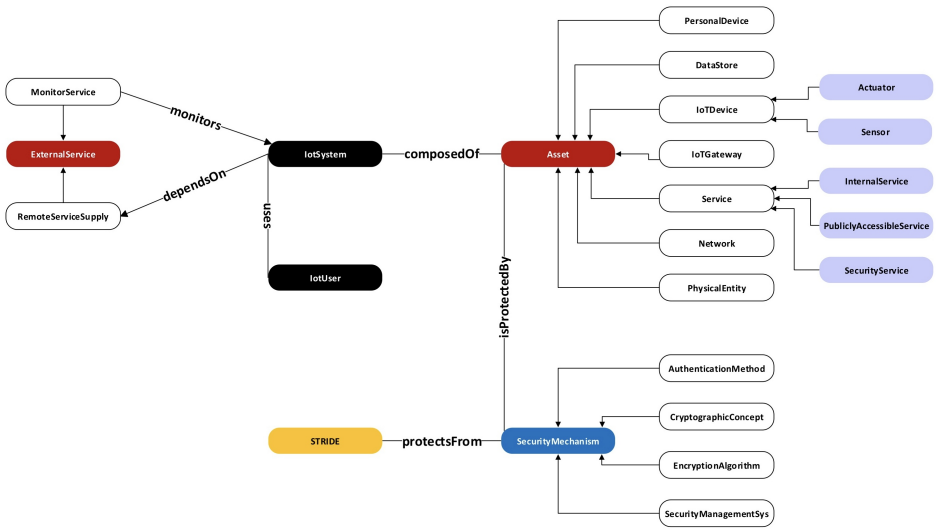

Figure 4.6: Classes and Subclasses of IoTSystem Sub-Ontology

4.1.2 Data Flow Sub-Ontology

The Data Flow Sub-Ontology models how information is transmitted within an IoT system. This sub-ontology is crucial for understanding how components of the system communicate with each other. The ontology has been developed according to the Data Flow ontology in the ThreMA framework [29]. The diagram representing this ontology is illustrated in Figure 4.7.

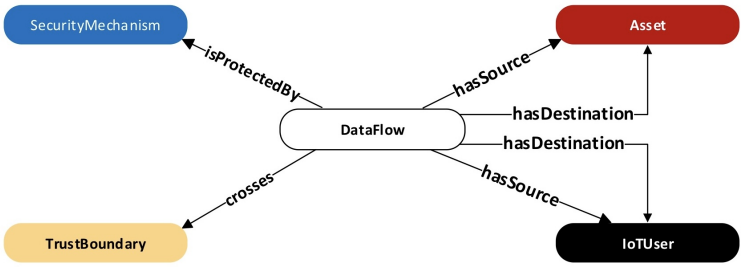

Figure 4.7: Data Flow Sub-Ontology

The main entity is DataFlow, representing the communication path between sources and destinations. This entity is linked to User and Asset classes via hasSource and hasDestination relationships. Moreover, the ontology models security mechanisms, that protect the data flow, and trust boundaries that could be crossed by data flows.
Here is a detailed list of the classes and relationships within the Data Flow Sub-Ontology:

- **DataFlow**: it represents a communication channel between specified source and destination and is modeled by using two relationships [29].
  - hasSourceA: Asset / IoTUser
  - hasDestinationA: Asset / IoTUser
  - isProtectedByD: SecurityMechanism
  - crosses: TrustBoundary
  - dataIsAffectedBy : CAPEC
  - hasThreatD: STRIDE

- **TrustBoundary**: models a change in the level of privileges between the source and destination of a data flow. This can be specified for a DataFlow entity using the relationship crosses [29].

- **IoTUser**: represents users interacting with the system, which could be either human or digital entities. The detailed description of the IoTUser class can be found in the IoT Sub-Ontology section.

- **SecurityMechanism**: represents security solutions used to protect the data flow. A full description of this class is provided in the IoT Sub-Ontology section, as it is a shared component.

4.1.3 Threats Sub-Ontology

The Threats Sub-Ontology models the threats targeting an IoT. Threats are organized into categories based on the type of component they target. This sub-ontology is inspired by the threat ontology presented in ThreMA [29].

However, new categories have been introduced. These categories are specific to IoT environments. As in ThreMA, the threat categories rely on the MITRE-CAPEC knowledge-base, focusing on the "Domains of Attack" view [26]. CAPEC threats are organized following different levels of abstraction: each category contains Meta Attack Patterns which contain Standard Attack Patterns. Each Standard Attack Pattern is composed of Detailed Attack Patterns. The ontology also integrates the STRIDE model. This model categorizes threats into six main types: Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service and Elevation of Privileges [27]. Also, CAPEC threats are mapped to STRIDE categories. The CAPEC-STRIDE mapping is based on the work presented in the CAPEC-STRIDE Mapping project [40].

Figure 4.8 illustrates the STRIDE threat model categories and threat categories identified in the threat sub-ontology. Each threat category is a class in the sub-ontology. Each category comes along with a list of CAPEC IDs that belong to that specific class. Each CAPEC ID is a subclass.

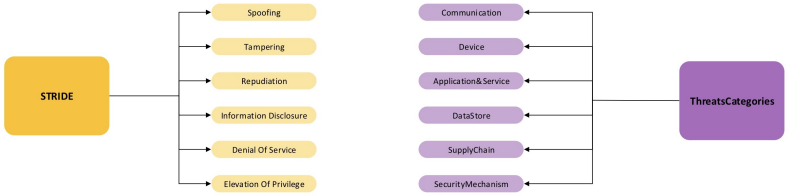

Figure 4.8: STRIDE Model Categories and Threats Categories

The objective is to ensure that the ontology links each threat category to specific attack patterns from CAPEC. Below is the detailed list of the classes in threat sub-ontology, along with their descriptions and CAPEC ID classes.

- **Communication**: represents threats related to network and gateway components in the IoT system. This category includes threats that could disrupt communication or compromise data integrity.
  - Targets: Network / IoTGateway / DataFlow
  - List of CAPEC IDs (instances of the CommunicationCAPEC class):
    - CAPEC-22: Exploiting Trust in Client
    - CAPEC-94: Adversary in the Middle (AiTM)
    - CAPEC-117: Interception
    - CAPEC-125: Flooding
    - CAPEC-148: Content Spoofing
    - CAPEC-151: Identity Spoofing
    - CAPEC-154: Resource Location Spoofing
    - CAPEC-161: Infrastructure Manipulation
    - CAPEC-169: Footprinting
    - CAPEC-192: Protocol Analysis
    - CAPEC-216: Communication Channel Manipulation
    - CAPEC-224: Fingerprinting
    - CAPEC-227: Sustained Client Engagement
    - CAPEC-272: Protocol Manipulation
    - CAPEC-292: Host Discovery
    - CAPEC-594: Traffic Injection
    - CAPEC-607: Obstruction

- **Device**: refers to threats targeting physical entities and IoT devices, including sensors, actuators and other hardware components. Threats in this category include hardware-based attacks.
  - Targets: PhysicalEntity, IoTDevice and Smart Device
  - List of CAPEC IDs (instances of the DeviceCAPEC class):
    - CAPEC-74: Manipulating State
    - CAPEC-116: Excavation
    - CAPEC-150: Collect Data from Common Resource Locations
    - CAPEC-154: Resource Location Spoofing
    - CAPEC-188: Reverse Engineering
    - CAPEC-212: Functionality Misuse
    - CAPEC-441: Malicious Logic Insertion
    - CAPEC-522: Malicious Hardware Component Replacement
    - CAPEC-607: Obstruction
    - CAPEC-624: Hardware Fault Injection
- **Service**: represents threats related to the software components of an IoT system, including applications and services.
  - Targets: Service and ExternalService
  - List of CAPEC IDs (instances of the ServiceCAPEC):
    - CAPEC-21: Exploitation of Trusted Identifiers
    - CAPEC-22: Exploiting Trust in Client
    - CAPEC-28: Fuzzing
    - CAPEC-49: Password Brute Forcing
    - CAPEC-55: Rainbow Table Password Cracking
    - CAPEC-70: Try Common or Default Usernames and Passwords
    - CAPEC-74: Manipulating State
    - CAPEC-94: Adversary in the Middle (AiTM)
    - CAPEC-111: JSON Hijacking (aka JavaScript Hijacking)
    - CAPEC-112: Brute Force
    - CAPEC-113: Interface Manipulation
    - CAPEC-114: Authentication Abuse
    - CAPEC-115: Authentication Bypass
    - CAPEC-116: Excavation
    - CAPEC-122: Privilege Abuse

    - CAPEC-123: Buffer Manipulation
    - CAPEC-137: Parameter Injection
    - CAPEC-153: Input Data Manipulation
    - CAPEC-160: Exploit Script-Based APIs
    - CAPEC-173: Action Spoofing
    - CAPEC-188: Reverse Engineering
    - CAPEC-212: Functionality Misuse
    - CAPEC-227: Sustained Client Engagement
    - CAPEC-240: Resource Injection
    - CAPEC-242: Code Injection
    - CAPEC-244: XSS Targeting URI Placeholders
    - CAPEC-248: Command Injection
    - CAPEC-388: Application API Button Hijacking
    - CAPEC-440: Hardware Integrity Attack
    - CAPEC-441: Malicious Logic Insertion
    - CAPEC-460: HTTP Parameter Pollution (HPP)
    - CAPEC-549: Local Execution of Code
    - CAPEC-565: Password Spraying
    - CAPEC-572: Artificially Inflate File Sizes
    - CAPEC-586: Object Injection

- **DataStore**: represents threats associated with data storage components in the IoT system.
  - Targets: DataStore
  - List of CAPEC IDs (instances of the DataStoreCAPEC class):
    - CAPEC-7: Blind SQL Injection
    - CAPEC-66: SQL Injection
    - CAPEC-74: Manipulating State
    - CAPEC-84: XQuery Injection
    - CAPEC-110: SQL Injection through SOAP Parameter Tampering
    - CAPEC-113: Interface Manipulation
    - CAPEC-147: XML Ping of the Death
    - CAPEC-248: Command Injection
    - CAPEC-441: Malicious Logic Insertion

- **SupplyChain**: focuses on threats that come from dependencies on external services and suppliers.
  - Targets: ExternalService (HardwareSupply / SoftwareSupply / RemoteServiceSupply)
  - List of CAPEC IDs (instances of the SupplyChainCAPEC class):
    * CAPEC-116: Excavation
    * CAPEC-184: Software Integrity Attack
    * CAPEC-188: Reverse Engineering
    * CAPEC-438: Modification During Manufacture
    * CAPEC-439: Manipulation During Distribution
    * CAPEC-440: Hardware Integrity Attack
    * CAPEC-443: Malicious Logic Inserted Into Product by Authorized Developer
    * CAPEC-563: Add Malicious File to Shared Webroot
    * CAPEC-607: Obstruction
    * CAPEC-624: Hardware Fault Injection
- **SecurityMechanism**: includes threats targeting security mechanisms used to protect the IoT system itself.
  - Targets: AuthenticationMethod / CryptographicConcept / EncryptionAlgorithm / SecurityManagementSystem
  - List of CAPEC IDs (instances of the SecurityMechanismCAPEC class):
    * CAPEC-21: Exploitation of Trusted Identifiers
    * CAPEC-112: Brute Force
    * CAPEC-114: Authentication Abuse
    * CAPEC-115: Authentication Bypass
    * CAPEC-116: Excavation
    * CAPEC-122: Privilege Abuse
    * CAPEC-233: Privilege Escalation
    * CAPEC-458: Flash Memory Attacks
    * CAPEC-554: Functionality Bypass
    * CAPEC-560: Use of Known Domain Credentials
    * CAPEC-624: Hardware Fault Injection

4.1.4 Mapping CAPEC-STRIDE

As previously said, CAPEC (Common Attack Pattern Enumeration and Classification) and STRIDE are two frameworks that classify potential threats. The CAPEC-STRIDE Mapping project [40] provides a mapping between CAPEC attack patterns and corresponding STRIDE categories. This mapping connects high-level threats with specific attack patterns that can be observed in the real world.
In this ontology, the mapping is performed using the property *isLabeledWithSTRIDE* as in ThreMa [29], linking instances of CAPEC to their corresponding STRIDE categories.
In the ontology there is a class called STRIDE with the following instances :

- Spoofing
- Tampering
- Repudiation
- InformationDisclosure
- DenialOfService
- ElevationOfPrivilege

Figure 4.9 illustrates the mapping between some STRIDE instances (Tampering, InformationDisclosure, DenialOfService) and CAPEC attack patterns instances (CAPEC-74, CAPEC-116, CAPEC-607), along with their corresponding threat categories (Device, SupplyChain). For example: The CAPEC-74 attack pattern, that belongs to Device threat class, *isLabeledWithSTRIDE* Tampering.

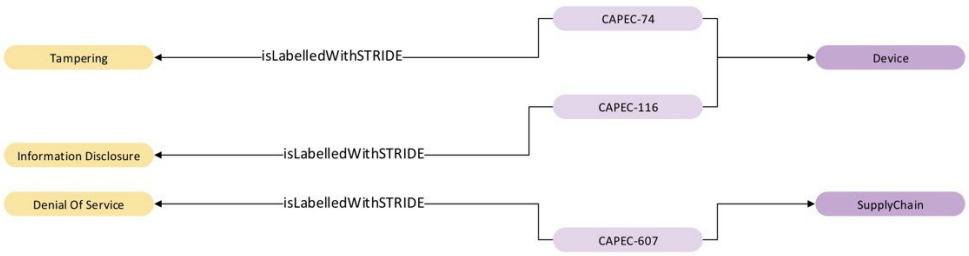

Figure 4.9: CAPEC-STRIDE Mapping Examples

The following is a detailed list of CAPEC instances mapped to the corresponding STRIDE instances:

- **Denial of Service**:
  - CAPEC-125: Flooding
  - CAPEC-147: XML Ping of the Death
  - CAPEC-227: Sustained Client Engagement
  - CAPEC-607: Obstruction
- **Elevation of Privilege**:
  - CAPEC-7: Blind SQL Injection
  - CAPEC-21: Exploitation of Trusted Identifiers
  - CAPEC-22: Exploiting Trust in Client
  - CAPEC-49: Password Brute Forcing
  - CAPEC-55: Rainbow Table Password Cracking
  - CAPEC-66: SQL Injection
  - CAPEC-70: Try Common or Default Usernames and Passwords
  - CAPEC-84: XQuery Injection
  - CAPEC-94: Adversary in the Middle (AiTM)
  - CAPEC-110: SQL Injection through SOAP Parameter Tampering
  - CAPEC-112: Brute Force
  - CAPEC-114: Authentication Abuse
  - CAPEC-115: Authentication Bypass
  - CAPEC-122: Privilege Abuse
  - CAPEC-137: Parameter Injection
  - CAPEC-233: Privilege Escalation
  - CAPEC-240: Resource Injection
  - CAPEC-242: Code Injection
  - CAPEC-244: XSS Targeting URI Placeholders
  - CAPEC-248: Command Injection
  - CAPEC-388: Application API Button Hijacking
  - CAPEC-460: HTTP Parameter Pollution (HPP)
  - CAPEC-549: Local Execution of Code

  - CAPEC-560: Use of Known Domain Credentials
  - CAPEC-563: Add Malicious File to Shared Webroot
  - CAPEC-565: Password Spraying
  - CAPEC-586: Object Injection
- **Repudiation**:
  - NONE
- **Information Disclosure**:
  - CAPEC-111: JSON Hijacking (aka JavaScript Hijacking)
  - CAPEC-116: Excavation
  - CAPEC-117: Interception
  - CAPEC-150: Collect Data from Common Resource Locations
  - CAPEC-169: Footprinting
  - CAPEC-184: Software Integrity Attack
  - CAPEC-188: Reverse Engineering
  - CAPEC-192: Protocol Analysis
  - CAPEC-212: Functionality Misuse
  - CAPEC-216: Communication Channel Manipulation
  - CAPEC-224: Fingerprinting
  - CAPEC-292: Host Discovery
  - CAPEC-554: Functionality Bypass
- **Spoofing**:
  - CAPEC-148: Content Spoofing
  - CAPEC-151: Identity Spoofing
  - CAPEC-154: Resource Location Spoofing
  - CAPEC-173: Action Spoofing
- **Tampering**:
  - CAPEC-28: Fuzzing
  - CAPEC-74: Manipulating State
  - CAPEC-113: Interface Manipulation

- CAPEC-123: Buffer Manipulation
- CAPEC-153: Input Data Manipulation
- CAPEC-160: Exploit Script-Based APIs
- CAPEC-161: Infrastructure Manipulation
- CAPEC-272: Protocol Manipulation
- CAPEC-438: Modification During Manufacture
- CAPEC-439: Manipulation During Distribution
- CAPEC-440: Hardware Integrity Attack
- CAPEC-441: Malicious Logic Insertion
- CAPEC-443: Malicious Logic Inserted Into Product by Authorized Developer
- CAPEC-458: Flash Memory Attacks
- CAPEC-522: Malicious Hardware Component Replacement
- CAPEC-572: Artificially Inflate File Sizes
- CAPEC-594: Traffic Injection
- CAPEC-624: Hardware Fault Injection

4.2 Inference Rules and Reasoning Mechanisms

This section explains the inference rules for the automation of the threat modeling process applied to the developed IoT ontology. These rules are implemented using the Semantic Web Rule Language (SWRL). Applying these rules, an automatic reasoning engine can infer potential threats based on the modeled relationships. The aim is to perform threat identification in different IoT systems.
SWRL rules follow a logical structure in the form of:

$$\text{antecedent} \Rightarrow \text{consequent}$$

This structure means that if the conditions in the antecedent are met, the consequent must also hold true. Both antecedent and consequent are expressed using ontology entities, classes and relationships. Each rule is designed to detect specific threats associated with the different categories identified in the ontology (e.g., Communication, Device, Service).

4.2.1 Logic and Structure of Inference Rules

The logic of the inference rules determines when a component of the IoT infrastructure is exposed to threats based on its relationships with other components or its configuration.
Authors of ThreMa [29] suggest, for instance, that in the SecurityMechanism category, a rule specifies that if an asset is protected by a specific mechanism (e.g., an encryption algorithm), the asset could still be vulnerable to threats associated with that security mechanism. Even when protective measures are active, potential risks related to their misuse are still considered. This is important because if a security solution is not applied in the correct way, then it can become a source of vulnerabilities.

4.2.2 Inference Rules

The following list shows the inference rules applied to each threat category in the ontology. These rules define the conditions such that specific IoT components are threatened based on their interactions and relationships.
Additionally, there are three rules whose objective is to demonstrate how each IoT system element is associated with the STRIDE categories by which it is threatened. These rules ensure that the corresponding STRIDE threat category is inferred for every asset affected by a specific CAPEC attack pattern. These rules are the key element of the automated reasoning process to identify potential threats.

Communication Rules (network, gateway, data flow)

The following list outlines the rules associated with the Communication threat category and each rule is paired with a concise explanation. These rules are also summarized in Table 4.1 for reference.

- **Network Communication Threat**:

$$\texttt{Network(?n)} \land \texttt{CommunicationCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?n, ?c)}$$

  Explanation: If an entity is a network, then it is affected by a communication threat.

- **IoT Gateway Communication Threat**:

$$\texttt{IOTGateway(?g)} \land \texttt{CommunicationCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?g, ?c)}$$

  Explanation: If an entity is an IoT gateway, then it is affected by a communication threat.

- **DataFlow Threat**:

$$\texttt{DataFlow(?df)} \land \texttt{CommunicationCAPEC(?c)} \rightarrow \texttt{dataIsAffectedBy(?df, ?c)}$$

Explanation: If a DataFlow exists, then it is affected by a communication threat.

- **Service Interacting Through Network**:

$$\texttt{Service(?s)} \land \texttt{interactsThrough(?s, ?n)} \land \texttt{Network(?n)} \land \texttt{CommunicationCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?s, ?c)}$$

Explanation: If a service interacts through a network, then the service is affected by a communication threat.

- **IoT Device Interacting Through Network**:

$$\texttt{IoTDevice(?i)} \land \texttt{interactsThroughIoT(?i, ?n)} \land \texttt{Network(?n)} \land \texttt{CommunicationCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?i, ?c)}$$

Explanation: If an IoT device interacts through a network, then the device is affected by a communication threat.

Device Rules (Physical Entity, IoTDevice)

The following list outlines the rules associated with the Device threat category and each rule is paired with a concise explanation. These rules are also summarized in Table 4.2 for reference.

- **Physical Entity Threat**:

$$\texttt{PhysicalEntity(?p)} \land \texttt{DeviceCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?p, ?c)}$$

Explanation: If an entity is a physical entity, then it is affected by a Device-CAPEC threat.

- **IoT Device Threat**:

$$\texttt{IOTDevice(?d)} \land \texttt{DeviceCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?d, ?c)}$$

Explanation: If an entity is an IoT device, then it is affected by a DeviceCAPEC threat.

- **Sensor Threat**:

$$\texttt{Sensor(?s)} \land \texttt{DeviceCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?s, ?c)}$$

Explanation: If an entity is a sensor, then it is affected by a DeviceCAPEC threat.

- **Actuator Threat**:

$$\texttt{Actuator(?a)} \land \texttt{DeviceCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?a, ?c)}$$

- **Smart Device Threat** :

$$\texttt{SmartDevice(?s)} \land \texttt{DeviceCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?s, ?c)}$$

Explanation: If an entity is a smart device, then it is affected by a Device-CAPEC threat.

Service Rules

The following list outlines the rules associated with the Service threat category and each rule is paired with a concise explanation. These rules are also summarized in Table 4.3 for reference.

- **Service Threat**:

$$\texttt{Service(?s)} \land \texttt{ServiceCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?s, ?c)}$$

Explanation: If an entity is a service, then it is affected by a ServiceCAPEC threat.

- **External Service Threat**:

$$\texttt{ExternalService(?s)} \land \texttt{ServiceCAPEC(?c)} \rightarrow \texttt{extIsAffectedBy(?s, ?c)}$$

Explanation: If an entity is an external service, then it is affected by a ServiceCAPEC threat.

- **Service interacting with IoTDevice Threat**:

$$\texttt{Service(?s)} \land \texttt{interactsWith(?s, ?d)} \land \texttt{IoTDevice(?d)} \land \texttt{ServiceCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?d, ?c)}$$

Explanation: If a service interacts with an IoT device, then the IoTDevice is affected by a ServiceCAPEC threat.

- **External Service interacting with IoTDevice Threat**:

$$\texttt{ExternalService(?s)} \land \texttt{interactsWithExt(?s, ?d)} \land \texttt{IoTDevice(?d)} \land \texttt{ServiceCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?d, ?c)}$$

  Explanation: If an external service interacts with an IoT device, then the IoTDevice is affected by a ServiceCAPEC threat.

DataStore Rules (Data Store)

The following list outlines the rules associated with the Data Store threat category and each rule is paired with a concise explanation. These rules are also summarized in Table 4.4 for reference.

- **Data Store Threat**:

$$\texttt{DataStore(?ds)} \land \texttt{DataStoreCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?ds, ?c)}$$

  Explanation: If an entity is a data store, then it is affected by a DataStore-CAPEC threat.

- **Service Using Data Store Threat**:

$$\texttt{Service(?s)} \land \texttt{usesDataStore(?s, ?ds)} \land \texttt{DataStore(?ds)} \land \texttt{DataStoreCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?s, ?c)}$$

  Explanation: If a service uses a data store, then the service is affected by a DataStoreCAPEC threat.

- **IoT Gateway Using Data Store Threat**:

$$\texttt{IOTGateway(?g)} \land \texttt{usesDataStore1(?g, ?ds)} \land \texttt{DataStore(?ds)} \land \texttt{DataStoreCAPEC(?c)} \rightarrow \texttt{isAffectedBy(?g, ?c)}$$

  Explanation: If an IoT gateway uses a data store, then it is affected by a DataStoreCAPEC threat.

SupplyChain Rules (External Service)

The following list outlines the rules associated with the Supply Chain threat category and each rule is paired with a concise explanation. These rules are also summarized in Table 4.5 for reference.

- **External Service Threat**:

$$\texttt{ExternalService(?e)} \land \texttt{SupplyChainCAPEC(?c)} \rightarrow \texttt{extIsAffectedBy(?e, ?c)}$$

Explanation: If an entity is an external service, then it is affected by a SupplyChainCAPEC threat.

- **Monitor Service Threat**:

$$\texttt{MonitorService(?hs)} \land \texttt{SupplyChainCAPEC(?c)} \rightarrow \texttt{extIsAffectedBy(?hs, ?c)}$$

Explanation: If an entity is a Monitor Service, then it is affected by a SupplyChainCAPEC threat.

- **Remote Service Supply Threat**:

$$\texttt{RemoteServiceSupply(?rss)} \land \texttt{SupplyChainCAPEC(?c)} \rightarrow \texttt{extIsAffectedBy(?rss, ?c)}$$

Explanation: If an entity is a remote service supply, then it is affected by a SupplyChainCAPEC threat.

- **IoT System Threat from External Service Dependency**:

$$\texttt{IOTSystem(?sys)} \land \texttt{dependsOn(?sys, ?es)} \land \texttt{ExternalService(?es)} \land \texttt{SupplyChainCAPEC(?c)} \rightarrow \texttt{extIsAffectedBy(?es, ?c)}$$

Explanation: If an IoT system depends on an external service, then the external service is affected by a SupplyChainCAPEC threat.

SecurityMechanism Rules (Security Mechanism)

The following list outlines the rules associated with the Security Mechanism threat category and each rule is paired with a concise explanation. These rules are also summarized in Table 4.6 for reference.

- **SecurityService Threat**:

$$\texttt{SecurityService(?ss)} \land \texttt{isProtectedBy(?a, ?ss)} \land \texttt{Asset(?a)} \land \texttt{SecurityMechanismCAPEC(?smc)} \rightarrow \texttt{isAffectedBy(?a, ?smc)}$$

Explanation: If an Asset is protected by a security service, then the asset is affected by a SecurityMechanismCAPEC threat.

- **Authentication Method Threat**:

$$\texttt{AuthenticationMethod(?am)} \land \texttt{isProtectedByS(?a, ?am)} \land \texttt{Asset(?a)} \land \texttt{SecurityMechanismCAPEC(?smc)} \rightarrow \texttt{isAffectedBy(?a, ?smc)}$$

  Explanation: If an Asset is protected by an authentication method, then the asset is affected by a SecurityMechanismCAPEC threat.

- **Cryptographic Concept Threat**:

$$\texttt{CryptographicConcept(?cc)} \land \texttt{isProtectedByS(?a, ?cc)} \land \texttt{Asset(?a)} \land \texttt{SecurityMechanismCAPEC(?smc)} \rightarrow \texttt{isAffectedBy(?a, ?smc)}$$

  Explanation: If an Asset is protected by a cryptographic concept, then the asset is affected by a SecurityMechanismCAPEC threat.

- **Encryption Algorithm Threat**:

$$\texttt{EncryptionAlgorithm(?ea)} \land \texttt{isProtectedByS(?a, ?ea)} \land \texttt{Asset(?a)} \land \texttt{SecurityMechanismCAPEC(?smc)} \rightarrow \texttt{isAffectedBy(?a, ?smc)}$$

  Explanation: If an Asset is protected by an encryption algorithm, then the asset is affected by a SecurityMechanismCAPEC threat.

- **Security Management System Threat**:

$$\texttt{SecurityManagementSystem(?sms)} \land \texttt{isProtectedByS(?a, ?sms)} \land \texttt{Asset(?a)} \land \texttt{SecurityMechanismCAPEC(?smc)} \rightarrow \texttt{isAffectedBy(?a, ?smc)}$$

  Explanation: If an asset is protected by a security management system, then the asset is affected by a SecurityMechanismCAPEC threat.

- **Protected Data Flow Threat**:

$$\texttt{DataFlow(?d)} \land \texttt{isProtectedByS(?d, ?sm)} \land \texttt{SecurityMechanism(?a)} \land \texttt{SecurityMechanismCAPEC(?sm)} \rightarrow \texttt{isAffectedBy(?d, ?sm)}$$

  Explanation: If a data flow is protected by a security mechanism, then the data flow is affected by a SecurityMechanismCAPEC threat.

- **Data Flow Threat**:

$$\texttt{DataFlow(?df)} \land \texttt{crosses(?df, ?tb)} \land \texttt{TrustBoundary(?tb)} \land \texttt{SecurityMechanismCAPEC(?smc)} \rightarrow \texttt{dataIsAffectedBy(?df, ?smc)}$$

  Explanation: If a data flow crosses a trust boundary, then the data flow is affected by a SecurityMechanismCAPEC threat.

STRIDE Threat Inference Rules

The following list outlines the rules associated with STRIDE categories and each rule is paired with a concise explanation. These rules are also summarized in Table 4.7 for reference.

- **Asset STRIDE Threat**:

$$\texttt{CAPEC(?c)} \land \texttt{Asset(?a)} \land \texttt{isAffectedBy(?a, ?c)} \land \texttt{STRIDE(?s)} \land \texttt{isLabeledWith(?c, ?s)} \rightarrow \texttt{hasThreat(?a, ?s)}$$

  Explanation: If an Asset is affected by a specific CAPEC attack pattern and if that CAPEC attack pattern is labeled with a STRIDE category, then the Asset has that particular STRIDE threat.

- **Data Flow STRIDE Threat**:

$$\texttt{CAPEC(?c)} \land \texttt{DataFlow(?d)} \land \texttt{dataIsAffectedBy(?d, ?c)} \land \texttt{STRIDE(?s)} \land \texttt{isLabeledWith(?c, ?s)} \rightarrow \texttt{hasThreatD(?d, ?s)}$$

  Explanation: If a data flow is affected by a specific CAPEC attack pattern and if that CAPEC attack pattern is labeled with a STRIDE category, then the data flow has that particular STRIDE threat.

- **External Service STRIDE Threat**:

$$\texttt{CAPEC(?c)} \land \texttt{ExternalService(?e)} \land \texttt{extIsAffectedBy(?e, ?c)} \land \texttt{STRIDE(?s)} \land \texttt{isLabeledWith(?c, ?s)} \rightarrow \texttt{hasThreatE(?e, ?s)}$$

  Explanation: If an external service is affected by a specific CAPEC attack pattern and if that CAPEC attack pattern is labeled with a STRIDE category, then the External Service has that particular STRIDE threat.

Table 4.1: Communication Category Rules

SWRL Rule	Explanation
Network(?n) ^ CommunicationCAPEC(?c) → isAffectedBy(?n, ?c)	If an entity is a network, then it is affected by a communication threat.
IOTGateway(?g) ^ CommunicationCAPEC(?c) → isAffectedBy(?g, ?c)	If an entity is an IoT gateway, then it is affected by a communication threat.
DataFlow(?df) ^ CommunicationCAPEC(?c) → dataIsAffectedBy(?df, ?c)	If a DataFlow exists, then it is affected by a communication threat.
Service(?s) ^ interactsThrough(?s, ?n) ^ Network(?n) ^ CommunicationCAPEC(?c) → isAffectedBy(?s, ?c)	If a service interacts through a network, then the service is affected by a communication threat.
IoTDevice(?i) ^ interactsThroughIoT(?i, ?n) ^ Network(?n) ^ CommunicationCAPEC(?c) → isAffectedBy(?i, ?c)	If an IoT device interacts through a network, then the device is affected by a communication threat.
DataFlow(?df) ^ crosses(?df, ?t) ^ TrustBoundary(?t) ^ CommunicationCAPEC(?c) → isAffectedBy(?df, ?c)	If a DataFlow crosses a TrustBoundary, then the DataFlow is affected by a communication threat.
DataFlow(?df) ^ hasSource(?df, ?src) ^ Asset(?src) ^ hasDestination(?df, ?dest) ^ Asset(?dest) ^ CommunicationCAPEC(?c) → isAffectedBy(?df, ?c)	If a DataFlow has a source Asset and a destination Asset, then it is affected by a communication threat.
DataFlow(?df) ^ isProtectedBy(?df, ?ea) ^ EncryptionAlgorithm(?ea) ^ CommunicationCAPEC(?c) → isAffectedBy(?df, ?c)	If a DataFlow is protected by an Encryption Algorithm, then it is affected by a communication threat.
DataFlow(?df) ^ isProtectedBy(?df, ?cc) ^ CryptographicConcept(?cc) ^ CommunicationCAPEC(?c) → isAffectedBy(?df, ?c)	If a DataFlow is protected by a Cryptographic Concept, then it is affected by a communication threat.

Table 4.2: Device Category Rules

SWRL Rule	Explanation
PhysicalEntity(?p) ^ DeviceCAPEC(?c) → isAffectedBy(?p, ?c)	If an entity is a physical entity, then it is affected by a DeviceCAPEC threat.
IOTDevice(?d) ^ DeviceCAPEC(?c) → isAffectedBy(?d, ?c)	If an entity is an IoT device, then it is affected by a DeviceCAPEC threat.
Sensor(?s) ^ DeviceCAPEC(?c) → isAffectedBy(?s, ?c)	If an entity is a sensor, then it is affected by a DeviceCAPEC threat.
Actuator(?a) ^ DeviceCAPEC(?c) → isAffectedBy(?a, ?c)	If an entity is an actuator, then it is affected by a DeviceCAPEC threat.
SmartDevice(?s) ^ DeviceCAPEC(?c) → isAffectedBy(?s, ?c)	If an entity is a Smart Device, then it is affected by a DeviceCAPEC threat.

Table 4.3: Service Category Rules

SWRL Rule	**Explanation**
Service(?s) ˆ ServiceCAPEC(?c) → isAffectedBy(?s, ?c)	If an entity is a service, then it is affected by a ServiceCAPEC threat.
ExternalService(?s) ˆ ServiceCAPEC(?c) → extIsAffectedBy(?s, ?c)	If an entity is an external service, then it is affected by a ServiceCAPEC threat.
Service(?s) ˆ interactsWith(?s, ?d) ˆ IoTDevice(?d) ˆ ServiceCAPEC(?c) → isAffectedBy(?d, ?c)	If a service interacts with an IoT device, then the IoT device is affected by a ServiceCAPEC threat.
ExternalService(?s) ˆ interactsWithExt(?s, ?d) ˆ IoTDevice(?d) ˆ ServiceCAPEC(?c) → isAffectedBy(?d, ?c)	If an external service interacts with an IoT device, then the IoT device is affected by a ServiceCAPEC threat.

Table 4.4: Data Store Category Rules

SWRL Rule	**Explanation**
DataStore(?ds) ˆ DataStoreCAPEC(?c) → isAffectedBy(?ds, ?c)	If an entity is a data store, then it is affected by a DataStoreCAPEC threat.
Service(?s) ˆ usesDataStore(?s, ?ds) ˆ DataStore(?ds) ˆ DataStoreCAPEC(?c) → isAffectedBy(?s, ?c)	If a service uses a data store, then the service is affected by a DataStoreCAPEC threat.
IOTGateway(?g) ˆ usesDataStore1(?g, ?ds) ˆ DataStore(?ds) ˆ DataStoreCAPEC(?c) → isAffectedBy(?g, ?c)	If an IoT gateway uses a data store, then it is affected by a DataStoreCAPEC threat.

Table 4.5: Supply Chain Category Rules

SWRL Rule	**Explanation**
ExternalService(?e) ˆ SupplyChainCAPEC(?c) → extIsAffectedBy(?e, ?c)	If an entity is an external service, then it is affected by a SupplyChainCAPEC threat.
MonitorService(?hs) ˆ SupplyChainCAPEC(?c) → extIsAffectedBy(?hs, ?c)	If an entity is a Monitor Service, then it is affected by a SupplyChainCAPEC threat.
RemoteServiceSupply(?rss) ˆ SupplyChainCAPEC(?c) → extIsAffectedBy(?rss, ?c)	If an entity is a remote service supply, then it is affected by a SupplyChainCAPEC threat.
IOTSystem(?sys) ˆ dependsOn(?sys, ?es) ˆ ExternalService(?es) ˆ SupplyChainCAPEC(?c) → extIsAffectedBy(?es, ?c)	If an IoT system depends on an external service, then the external service is affected by a SupplyChainCAPEC threat.

Table 4.6: Security Mechanism Category Rules

SWRL Rule	**Explanation**
SecurityService(?ss) ^ isProtectedBy(?a, ?ss) ^ Asset(?a) ^ SecurityMechanismCAPEC(?smc) → isAffectedBy(?a, ?smc)	If an Asset is protected by a security service, then the asset is affected by a SecurityMechanismCAPEC threat.
AuthenticationMethod(?am) ^ isProtectedByS(?a, ?am) ^ Asset(?a) ^ SecurityMechanismCAPEC(?smc) → isAffectedBy(?a, ?smc)	If an Asset is protected by an authentication method, then the asset is affected by a SecurityMechanismCAPEC threat.
CryptographicConcept(?cc) ^ isProtectedByS(?a, ?cc) ^ Asset(?a) ^ SecurityMechanismCAPEC(?smc) → isAffectedBy(?a, ?smc)	If an Asset is protected by a cryptographic concept, then the asset is affected by a SecurityMechanismCAPEC threat.
EncryptionAlgorithm(?ea) ^ isProtectedByS(?a, ?ea) ^ Asset(?a) ^ SecurityMechanismCAPEC(?smc) → isAffectedBy(?a, ?smc)	If an Asset is protected by an encryption algorithm, then the asset is affected by a SecurityMechanismCAPEC threat.
SecurityManagementSystem(?sms) ^ isProtectedByS(?a, ?sms) ^ Asset(?a) ^ SecurityMechanismCAPEC(?smc) → isAffectedBy(?a, ?smc)	If an asset is protected by a security management system, then the asset is affected by a SecurityMechanismCAPEC threat.
DataFlow (?d) ^ isProtectedByD (?d, ?sm) ^ SecurityMechanism (?sm) ^ SecurityMechanismCAPEC (?smc) → dataIsAffectedBy (?d, ?smc)	If a data flow is protected by a security mechanism, then the data flow is affected by a SecurityMechanismCAPEC threat.
DataFlow(?df) ^ crosses(?df, ?tb) ^ TrustBoundary(?tb) ^ SecurityMechanismCAPEC(?smc) → dataIsAffectedBy(?df, ?smc)	If a data flow crosses a trust boundary, then the data flow is affected by a SecurityMechanismCAPEC threat.

Table 4.7: STRIDE Category Rules for Assets, Data Flows, and External Services

SWRL Rule	**Explanation**
CAPEC(?c) ^ Asset(?a) ^ isAffectedBy(?a, ?c) ^ STRIDE(?s) ^ isLabeledWith(?c, ?s) → hasThreat(?a, ?s)	If an Asset is affected by a specific CAPEC attack pattern and if that CAPEC attack pattern is labeled with a STRIDE category, then the Asset has that particular STRIDE threat.
CAPEC(?c) ^ DataFlow(?d) ^ dataIsAffectedBy(?d, ?c) ^ STRIDE(?s) ^ isLabeledWith(?c, ?s) → hasThreatD(?d, ?s)	If a data flow is affected by a specific CAPEC attack pattern and if that CAPEC attack pattern is labeled with a STRIDE category, then the data flow has that particular STRIDE threat.
CAPEC(?c) ^ ExternalService(?e) ^ extIsAffectedBy(?e, ?c) ^ STRIDE(?s) ^ isLabeledWith(?c, ?s) → hasThreatE(?e, ?s)	If an external service is affected by a specific CAPEC attack pattern and if that CAPEC attack pattern is labeled with a STRIDE category, then the External Service has that particular STRIDE threat.

Chapter 5

Case Study

In this chapter, we will present the application and validation of the ontology and the inference rules to perform automated threat modeling on an IoT infrastructure. The aim is to analyze the effectiveness of the work focusing on the results obtained. We delve into the outcomes of Threat Modeling, providing a detailed analysis of the case study.

5.1 HArMoNICS Case Study

A case study is a detailed examination of a particular case in the real-world context, used to test and validate a developed tool or framework by applying it to the chosen scenario.

In this thesis, "HArMoNICS" has been considered as a case study [41]; this scenario has been proposed within the European project SPARTA. HArMoNICS (High Assurance Microgrid Network Infrastructure Case Study) is a case study infrastructure meant to provide a playground for testing security tools. It represents a digital replica of a real Smart Polygeneration Microgrid (SPM) located in Italy. Although most of the components are based on or inspired by the real system, HArMoNICS has been enriched with further security-relevant features. The HArMoNICS case study is based on a smart building scenario, composed of both IT and OT elements. The scenario is inspired by the Zero-Emission Building (ZEB), which is hosted inside the Genoa University Campus, located in Savona (Italy). It is described by the authors as designed and implemented in order to mimic real Intelligent Infrastructures, and therefore it includes a number of technologies that may appear within the perimeter of a smart infrastructure. The graphical representation of HArMoNICS infrastructure can be visualized in figure 5.1.

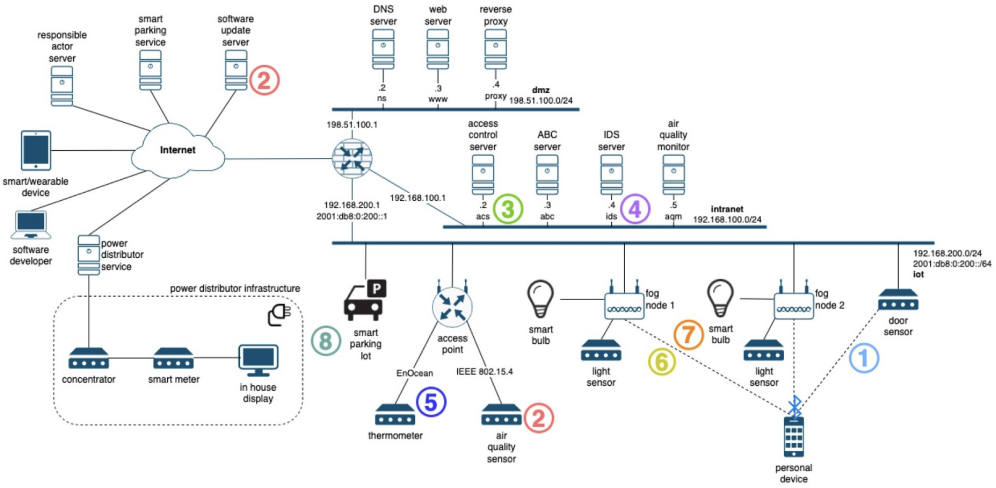

Figure 5.1: HArMoNICS Infrastructure Representation from [41]

5.1.1 Ontology Validation

The first step in testing the work of this thesis is validating the IoT Ontology. The components of HArMoNICS are mapped to the classes of the developed ontology. Each element of HArMoNICS represents an individual of a class in the ontology. The second step is to identify and add the most relevant relationships between elements in HArMoNICS. The aim of these two steps is to ensure that identified classes in the IoT ontology are comprehensive enough to represent all elements of a real IoT infrastructure. The mapping between the instances of HArMoNICS and classes of the ontology is shown in the list below.

- **HArMoNICS**:
  - **instance of**: IoTSystem
  - **composedOf**: DMZ Intranet IoTSegment
  - **dependsOn**: SmartParkingService PowerDistributorService Software-UpdateServer
- **DMZ Segment** In figure 5.2 there is a diagram that illustrates the DMZ Segment within the HArMoNICS infrastructure.
  - **DMZ**: it stands for Demilitarized Zone and it is a network segment that hosts publicly accessible services of the HArMoNICS infrastructure [41].
    - **instance of**: Network
    - **isProtectedBy**: RouterFirewall

- **DNSServer**: Provides domain name resolution services within the DMZ segment.
  - **instance of**: PubliclyAccessibleService
  - **interactsThrough**: DMZ
- **WebServer**: Hosts web applications and services that are accessible from external users connecting through the DMZ.
  - **instance of**: PubliclyAccessibleService
  - **interactsThrough**: DMZ
- **ReverseProxy**: Facilitates traffic between external users and internal services, providing load balancing and security in the DMZ.
  - **instance of**: PubliclyAccessibleService
  - **interactsThrough**: DMZ

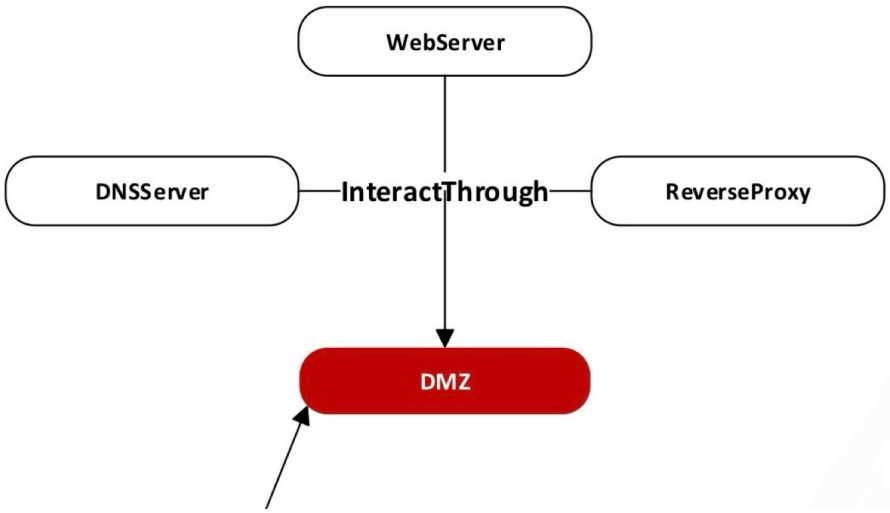

Figure 5.2: DMZ Segment of HArMoNICS Case Study

- **Intranet Segment**

  In figure 5.3 there is a diagram which illustrates the DMZ Segment within the HArMoNICS infrastructure.

  - **Intranet**: Internal network that hosts critical services for the infrastructure. Services in this network are not accessible from outside users.
    - **instance of**: Network
    - **isProtectedBy** : RouterFirewall

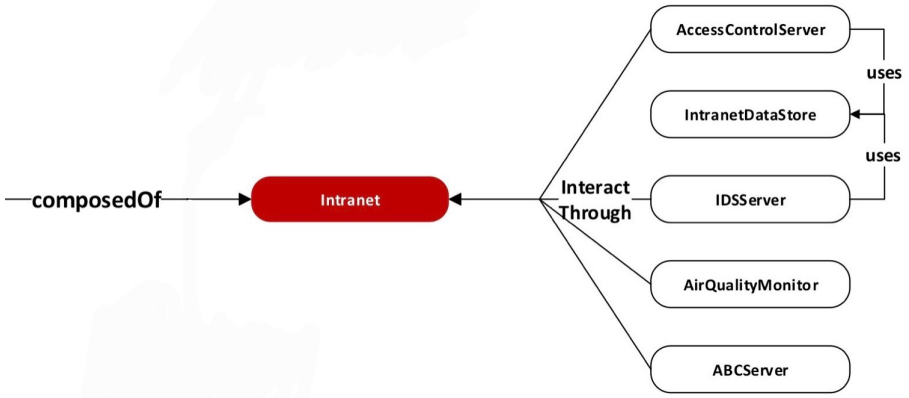

Figure 5.3: Intranet Segment of HArMoNICS Case Study

- **AccessControlServer**: Manages authentication and authorization services for internal users.
  - **instance of**: SecurityService
  - **interactsThrough**: Intranet
  - **usesDataStore**: IntranetDataStore
  - **isProtectedByS**: MFASecureAccess
  - **protectsFrom**: Spoofing, Elevation of Privilege, Information Disclosure
- **ABCServer**: Internal server managing general activity within the network.
  - **instance of**: InternalService
  - **interactsThrough**: Intranet
- **IDSServer**: a server that performs intrusion detection in the internal network.
  - **instance of**: SecurityService
  - **interactsThrough**: Intranet
  - **usesDataStore**: IntranetDataStore
  - **isProtectedByS**: EDRMonitoring
  - **protectsFrom**: Tampering, Repudiation, Denial of Service

- **AirQualityMonitor**: a device that checks the level of some air contaminants and helps to assess if the air is safe to breathe.
  - **instance of**: InternalService
  - **interactsThrough**: Intranet
- **IntranetDataStore:**
  - **istance of**: DataStore

- **IoT Segment** In figure 5.4 there is a diagram which illustrates the IoT Segment within the HArMoNICS infrastructure.

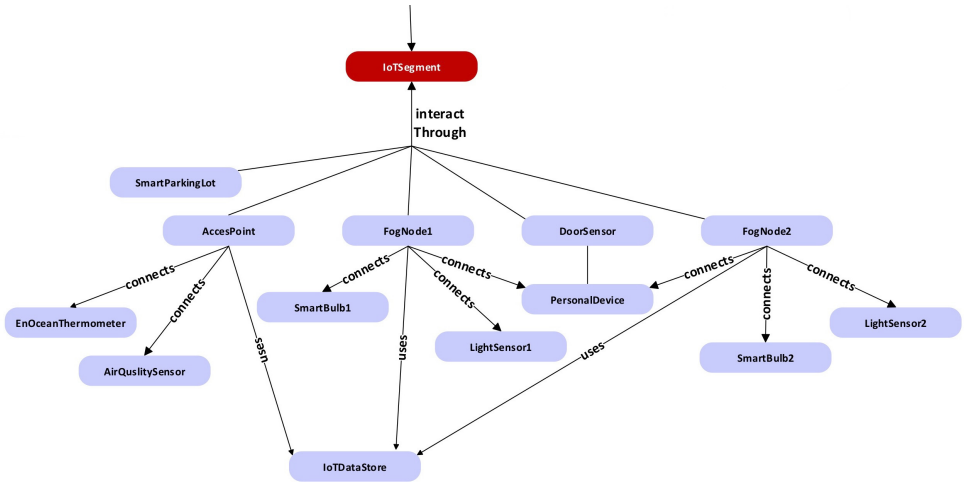

Figure 5.4: IoT Segment of HArMoNICS Case Study

- **IoTSegment**: Network segment dedicated to IoT devices such as sensors, actuators, and gateways [41].
  - **istance of**: Network
  - **isProtectedBy**: RouterFirewall
- **AccessPoint**: Connects IoT devices to the IoT network, acting as an entry point for wireless devices [41].
  - **istance of**: IoTGateway
  - **interactsThroughGW**: IoTSegment
  - **usesDataStore1**: IoTDataStore
  - **connects**: EnOceanThermometer
  - **connects**: AirQualitySensor

- **SmartParkingLot**: it is a system that monitors vehicles' presence in the parking area through sensors.
  - **istance of**: IoTDevice
  - **interactsThroughIoT**: IoTSegment
  - **isProtectedByS**: AES256
- **EnOceanThermometer**: it is a sensor that measures the temperature of the environment and transmits the data to IoT systems for analysis.
  - **istance of**: Sensor
  - **interactsThroughIoT**: IoTSegment
  - **monitors**: SmartBuilding
- **AirQualitySensor**: Detects air quality levels and transmits the data to IoT systems for analysis [41].
  - **istance of**: Sensor
  - **interactsThroughIoT**: IoTSegment
  - **monitors**: SmartBuilding
- **FogNode1**: Processes IoT data locally, reducing latency and processing time [41].
  - **istance of**: IoTGateway
  - **interactsThroughGW**: IoTSegment
  - **usesDataStore1**: IoTDataStore
  - **connects**: SmartBulb1
  - **connects**: LightSensor1
  - **connectsD**: PersonalDevice
- **FogNode2**: Similar to Fog Node 1, it provides localized processing for IoT data [41].
  - **istance of**: IoTGateway
  - **interactsThroughGW**: IoTSegment
  - **usesDataStore1**: IoTDataStore
  - **connects**: SmartBulb2
  - **connects**: LightSensor2
  - **connectsD**: PersonalDevice
- **SmartBulb1**: Actuator that controls lighting in smart environments [41].
  - **istance of**: Actuator
  - **interactsThroughIoT**: IoTSegment
  - **actsOn**: SmartBuilding

- **SmartBulb2**: Another actuator controlling lighting in smart environments [41].
  - **istance of**: Actuator
  - **interactsThroughIoT**: IoTSegment
  - **actsOn**: SmartBuilding
- **LightSensor1**: Measures light intensity and provides data for smart lighting systems [41].
  - **istance of**: Sensor
  - **interactsThroughIoT**: IoTSegment
  - **monitors**: SmartBuilding
- **LightSensor2**: Similar to Light Sensor 1, it measures light levels in smart environments [41].
  - **istance of**: Sensor
  - **interactsThroughIoT**: IoTSegment
  - **monitors**: SmartBuilding
- **DoorSensor**: Detects the opening and closing of doors in smart environments [41].
  - **istance of**: Sensor
  - **interactsThroughIoT**: IoTSegment
  - **monitors**: SmartBuilding
- **PersonalDevice**: A device used by a user, such as a smartphone or wearable device, that interacts with IoT systems.
  - **istance of**: SmartDevice
  - **interactsThroughSD**: IoTSegment
  - **isProtectedByS**: BiometricAccess
- **IoTDataStore:**
  - **istance of**: DataStore
  - **isProtectedByS**: AES256
- **SmartBuilding**: It is a physical component in the ontology, but it has been added for completeness. Sensors in the IoT segment monitor aspects of the building, such as temperature, air quality, and light levels, while actuators respond by adjusting these parameters.
  - **instance of**: PhysicalEntity

- **External Components**

  In figure 5.5 there is a diagram which illustrates the DMZ Segment within the HArMoNICS infrastructure.

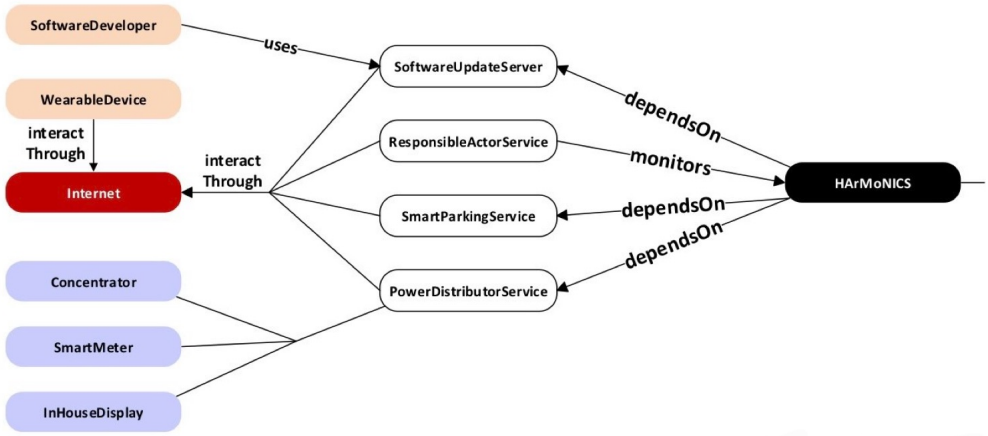

Figure 5.5: External Components in HArMoNICS

  - **Internet**:
    - **istance of**: Network
  - **ResponsibleActorServer**: Monitors and manages internal services, ensuring proper governance of the system [41].
    - **istance of**: MonitorService
    - **interactsThroughExt**: Internet
    - **monitors1**: HArMoNICS
  - **SmartParkingService**: Externally hosted service providing parking availability information to the IoT system [41].
    - **istance of**: RemoteServiceSupply
    - **interactsThroughExt**: Internet
  - **SoftwareUpdateServer**: Provides external software updates to IoT devices in the network [41].
    - **istance of**: RemoteServiceSupply
    - **interactsThroughExt**: Internet

  - **PowerDistributorService**: Externally hosted service providing power distribution and management data to the system [41].
    * **istance of**: RemoteServiceSupply
    * **interactsThroughExt**: Internet
    * **interactsWithExt**: Concentrator
    * **InteractsWithExt**: SmartMeter
    * **InteractsWithExt**: InHouseDisplay
  - **WearableDevice**: A personal device, often equipped with IoT applications, used by human users to interact with the IoT system.
    * **istance of**: SmartDevice
    * **interactsThroughSD**: Internet
  - **SoftwareDeveloper**: A human user involved in developing and maintaining IoT applications and services.
    * **istance of**: IoTUser
    * **usesExtService**: SoftwareUpdateServer

- **Power Distribution and Smart Infrastructure**
  - **Concentrator**: Aggregates and processes data from various IoT devices in the power distribution system [41].
    * **istance of**: IoTDevice
    * **interactsThroughIoT**: Internet
  - **SmartMeter**: Measures and reports energy usage in smart homes or infrastructures [41].
    * **istance of**: IoTDevice
    * **interactsThroughIoT**: Internet
  - **InHouseDisplay**: Displays real-time energy usage and other smart infrastructure data to users in a smart home or building [41].
    * **istance of**: IoTDevice
    * **interactsThroughIoT**: Internet

- **Additional Security Services and Security Mechanisms**
  - **RouterFirewall**: Protects the boundaries between the different network segments (Intranet, IoT, DMZ) from unauthorized access [41].
    * **instance of**: SecurityService
    * **protectsFrom**: Spoofing, Tampering, Denial of Service

  - **VPNServer**: Provides secure connections between external users and the internal network, allowing remote access to services while maintaining security [41].
    * **instance of**: SecurityService
    * **protectsFrom**: Information Disclosure, Spoofing, Elevation of Privilege
  - **MFASecureAccess**: Multi-Factor Authentication method for verifying the identity of users or devices accessing the system. This could be applied to critical components to ensure secure authentication.
    * **instance of**: AuthenticationMethod
    * **protectsFrom1**: Spoofing, Elevation of Privilege
  - **BiometricAccess**: Biometric authentication is used to verify physical users accessing smart devices in the building.
    * **instance of**: AuthenticationMethod
    * **protectsFrom1**: Spoofing, Elevation of Privilege
  - **AES256**: A symmetric encryption algorithm (Advanced Encryption Standard) used to protect sensitive data.
    * **instance of**: EncryptionAlgorithm
    * **protectsFrom1**: Information Disclosure, Tampering
  - **RSA2048**: An asymmetric encryption algorithm for secure communication between components.
    * **instance of**: EncryptionAlgorithm
    * **protectsFrom1**: Information Disclosure, Spoofing, Tampering
  - **EDRMonitoring**: Endpoint Detection and Response system used to monitor and detect threats on the internal network.
    * **instance of**: SecurityMechanism
    * **protectsFrom1**: Tampering, Repudiation, Denial of Service

- **Examples of possible TrustBoundary** The instances of TrustBoundary presented in this list have been introduced for analysis and validation purposes. They represent the change in the level of privileges between the source and destination of a data flow belonging to different network segments.
  - **TB-DMZ-Intranet**
  - **TB-IoTSegment-Intranet**
  - **TB-IoTSegment-DMZ**

  - **TB-Internet-DMZ**
  - **TB-Internet-IoTSegment**
  - **TB-Internet-Intranet**

- **Examples of possible Data Flow** The instances of DataFlow presented in this list have been introduced for analysis and validation purposes.

  - **DF1 between ABCServer and IntranetDataSore**: This data flow represents communication between the ABCServer and the IntranetDataStore within the Intranet segment. The ABCServer, responsible for managing general activities, stores data in the IntranetDataStore.
    * **instance of**: DataFlow
    * **hasSourceA**: ABCServer
    * **hasDestinationA**: IntranetDataStore
    * **isProtectedByD**: MFASecureAccess
  - **DF2 between WebServer and DNSServer**: the WebServer communicates with the DNSServer to resolve domain names for external users accessing its services. This communication remains within the DMZ, as both servers are in the same network segment.
    * **instance of**: DataFlow
    * **hasSouceA**: WebServer
    * **hasDestinationA**: DNSServer
  - **DF3 between FogNode1 and IoTDataStore**: FogNode1 processes IoT data and communicates with the IoTDataStore for storage purposes, all within the IoT segment. The communication is encrypted using AES256.
    * **instance of**: DataFlow
    * **hasSouceA**: FogNode1
    * **hasDestinationA**: IoTDataStore
    * **isProtectedByD**: AES256
  - **DF4 between SmartParkingLot and SmartParkingService**: the SmartParkingLot, a system within the IoT segment that monitors vehicle presence, communicates with the SmartParkingService. The SmartParkingLot sends parking availability data to the external service.
    * **instance of**: DataFlow
    * **hasSouceA**: SmartParkingLot
    * **hasDestinationA**: SmartParkingService
    * **crosses**: TB-Internet-IoTSegment

- **DF5 between WebServer and AccessControlServer**: the WebServer is part of the DMZ (Demilitarized Zone) segment and the AccessControlServer is located within the Intranet. The WebServer hosts publicly accessible services and needs to communicate with the AccessControlServer to verify authentication and authorization requests for internal users.
  - **instance of**: DataFlow
  - **hasSouceA**: WebServer
  - **hasDestinationA**: AccessControlServer
  - **crosses**: TB-DMZ-Intranet
  - **isProtectedByD**: RSA2048
- **DF6 between AirQualitySensor and AirQualityMonitor**: in this case, the AirQualitySensor, located in the IoTSegment, sends data about air quality to the AirQualityMonitor, which is part of the Intranet segment.
  - **instance of**: DataFlow
  - **hasSouceA**: AirQualitySensor
  - **hasDestinationA**: AirQualityMonitor
  - **crosses**: TB-IoTSegment-Intranet
  - **isProtectedByD**: AES256

5.1.2 Threat Modeling Result

Once the ontology model for the HArMoNICS IoT infrastructure was constructed, the automated threat modeling process began starting the Pellet [42] reasoner available in Protégé. By applying the predefined SWRL rules composing the Threat Modeling Logic, we aimed to automatically identify the CAPEC IDs associated with each individual in the HArMoNICS infrastructure. Protégé then generated a list of CAPEC IDs linked to each individual by the relationship *isAffectedBy*.
In addition to identifying CAPEC threats, the reasoner also inferred the corresponding STRIDE threat categories for each individual in HArMoNICS. This was achieved by applying SWRL rules that map each CAPEC attack pattern to a relevant STRIDE category. The inferred property *hasThreat* was automatically added to the individuals in the ontology, indicating which STRIDE categories (e.g., Tampering, Denial of Service) threaten each component.
Firstly, an example will be provided to demonstrate what a user would see in Protégé, highlighting how both the CAPEC IDs and STRIDE categories are linked to an individual in the ontology. The images will help clarify how the results are generated in practice. Then, general results will be presented, organized by classes of individuals in the ontology.

Finally, Table 5.1 will provide an overview of the CAPEC IDs and STRIDE categories identified for various classes in the HArMoNICS. The table highlights the specific individuals within each class (such as IoT devices, data stores, networks, and services), the number of CAPEC IDs and the STRIDE threats associated with them.

Practical Example of Threat Modeling in Protégé

This section provides a practical example of how CAPEC IDs and STRIDE threats are mapped to an individual in the HArMoNICS ontology using Protégé. In this example, the chosen individual is the *AirQualitySensor*in the HArMoNICS infrastructure.

In Figure 5.6, the *AirQualitySensor* individual is shown in the Protégé interface. The ontology hierarchy on the left shows the AirQualitySensor as an instance of the Sensor class, which is a subclass of IoTDevice class. The center pane highlights the object properties and relationships associated with the AirQualitySensor. The reasoner has inferred several CAPEC IDs that affect this sensor, which are listed under the property assertions section on the right. These attack patterns include, for instance, CAPEC-188 (Reverse Engineering), CAPEC-292 (Host Discovery), and CAPEC-148 (Content Spoofing), all linked to the individual through the relationship *isAffectedBy*. Additionally, the figure illustrates how the AirQualitySensor interacts with other elements in the HArMoNICS ontology, such as the IoTSegment and the SmartBuilding through the relationships *interactsThroughIoT* and *monitors*.

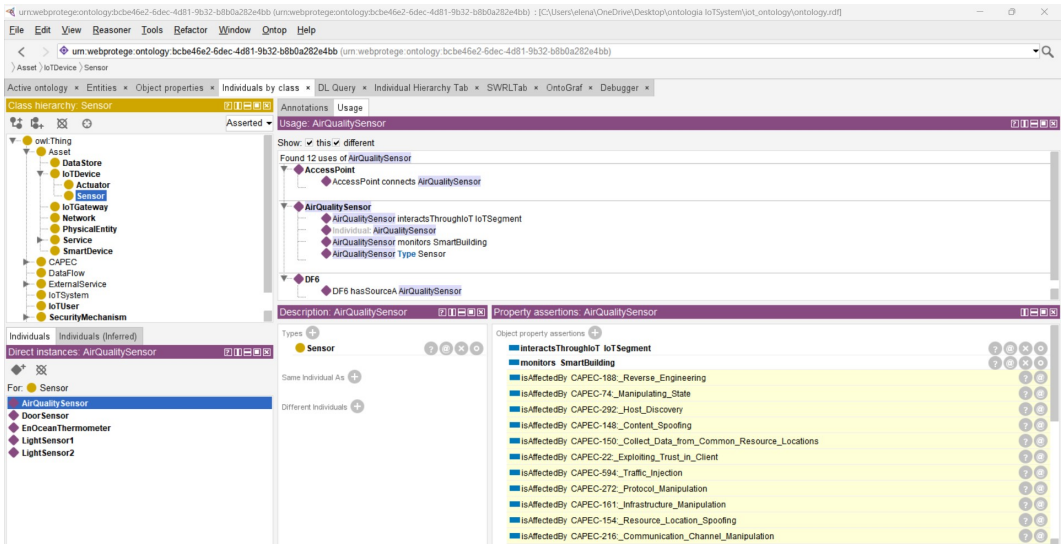

Figure 5.6: CAPEC IDs Mapped to the AirQualitySensor in Protégé

Figure 5.7 provides a more detailed view of the *AirQualitySensor* individual in the Protégé interface, focusing on the Property assertions related to both the CAPEC IDs and the inferred STRIDE threats categories, such as Spoofing, Denial of Service, Elevation of Privilege, and others, which are linked to the AirQualitySensor using the *hasThreat* property.

Figure 5.7: CAPEC IDs and STRIDE Threats Mapped to the AirQualitySensor in Protégé

CAPEC IDs and STRIDE threats for IoT Devices

The following list presents CAPEC IDs and STRIDE threats identified for IoT devices, including sensors and actuators, in HArMoNICS.

- **Concentrator, InHouseDisplay, SmartMeter:**
  - CAPEC- 292, 114, 112, 148, 150, 22, 594, 161, 28, 137, 227, 522, 460, 117, 115, 212, 49, 244, 55, 169, 153, 192, 154, 216, 173, 160, 624, 94, 586, 116, 565, 607, 441, 70, 21, 242, 224, 125, 240, 572, 123, 111, 440, 122, 388
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure
- **SmartParkingLot:**
  - CAPEC- 188, 74, 292, 114, 112, 148, 150, 22, 594, 554, 272, 161, 154, 216, 560, 624, 94, 227, 522, 117, 116, 115, 212, 607, 441, 21, 169, 224, 125, 192, 458, 151, 122, 233
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure

- **Smart Bulbs, Air Quality Sensor, Door Sensor, EnOcean Thermometer, Light Sensors:**
  - CAPEC- 188, 74, 292, 148, 150, 22, 594, 272, 161, 154, 216, 624, 94, 227, 522, 117, 116, 212, 607, 441, 169, 224, 125, 192, 151
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure

CAPEC IDs and STRIDE threats for Data Stores

The following are the CAPEC IDs and STRIDE threats identified for the data stores in the system:

- **IntranetDataStore:**
  - CAPEC- 74, 441, 147, 66, 84, 248, 113, 110, 7
  - STRIDE Threats: Denial Of Service, Elevation of Privilege, Tampering
- **IoTDataStore:**
  - CAPEC- 147, 441, 66, 114, 112, 21, 554, 7, 560, 624, 458, 84, 122, 113, 248, 110, 116, 115, 233
  - STRIDE Threats: Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure

CAPEC IDs and STRIDE threats for IoT Gateways

The following are the CAPEC IDs and STRIDE threats identified for the IoT gateway devices:

- **Access Point, FogNode1, FogNode2:**
  - CAPEC- 74, 292, 147, 66, 148, 22, 594, 272, 161, 7, 154, 216, 94, 84, 227, 117, 607, 441, 169, 224, 192, 151, 113, 248, 110
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure

CAPEC IDs and STRIDE threats for Networks

The following are the CAPEC IDs and STRIDE threats identified for the network components:

- **Internet:**
  - CAPEC- 292, 607, 148, 22, 594, 169, 272, 125, 224, 161, 154, 216, 192, 94, 151, 227, 117
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure
- **DMZ, Intranet, IoTSegment:**
  - CAPEC- 292, 607, 148, 22, 594, 169, 272, 125, 224, 161, 154, 216, 192, 94, 151, 227, 117, 116, 115, 607, 21, 169, 224, 125, 192, 458, 151, 122, 233
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure

CAPEC IDs and STRIDE threats for Physical Entities

The following are the CAPEC IDs and STRIDE threats identified for the physical entity:

- **SmartBuilding:**
  - CAPEC- 188, 74, 607, 441, 624, 150, 522, 116, 212, 154
  - STRIDE Threats: Spoofing, Denial Of Service, Tampering, Information Disclosure

CAPEC IDs and STRIDE threats for Services

The following are the CAPEC IDs and STRIDE threats identified for the services including Internal services, publicly accessible services and security services:

- **ABC server, Air Quality Monitor, DNS Server, Reverse Proxy, Web Server:**
  - CAPEC- 292, 114, 112, 148, 22, 594, 161, 28, 137, 227, 460, 117, 115, 212, 49, 244, 55, 169, 153, 192, 151, 113, 248, 188, 74, 549, 272, 154, 216, 173, 160, 94, 586, 116, 565, 607, 441, 70, 21, 242, 224, 125, 240, 572, 123, 111, 440, 122, 388
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure

- **VPN Server, Router Firewall:**
  - CAPEC- 188, 74, 114, 112, 22, 549, 173, 28, 160, 137, 94, 586, 227, 460, 116, 115, 212, 49, 565, 244, 70, 441, 21, 55, 242, 153, 240, 572, 123, 111, 440, 122, 113, 248, 388
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure
- **IDS Server:**
  - CAPEC- 292, 147, 114, 112, 148, 22, 594, 161, 28, 137, 84, 227, 460, 117, 115, 212, 49, 244, 55, 169, 153, 192, 151, 113, 248, 188, 74, 66, 549, 272, 7, 154, 216, 173, 160, 94, 586, 116, 565, 607, 441, 70, 21, 242, 224, 125, 240, 572, 123, 111, 440, 122, 388, 110
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure
- **Access Control Server:**
  - CAPEC- 292, 147, 114, 112, 148, 22, 594, 161, 560, 28, 137, 84, 227, 460, 117, 115, 212, 49, 244, 55, 169, 153, 192, 458, 151, 113, 248, 188, 74, 66, 549, 554, 272, 7, 154, 216, 173, 160, 624, 94, 586, 116, 565, 607, 441, 70, 21, 242, 224, 125, 240, 572, 123, 111, 440, 122, 388, 110, 233
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure

CAPEC IDs and STRIDE threats for Smart Devices

The following are the CAPEC IDs and STRIDE threats identified for the smart devices:

- **Wearable Device:**
  - CAPEC- 188, 74, 607, 441, 624, 150, 522, 116, 212, 154
  - STRIDE Threats: Spoofing, Denial Of Service, Tampering, Information Disclosure
- **Personal Device:**
  - CAPEC- 188, 74, 607, 441, 624, 150, 522, 116, 212, 154, 560, 624, 458, 122, 522, 116, 115, 212, 233
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure

CAPEC IDs and STRIDE threats for Data Flow

The following are the CAPEC IDs and STRIDE threats identified for data flows:

- **DF1, DF3, DF4, DF5, DF6:**
  - CAPEC- 292, 607, 148, 22, 594, 169, 272, 125, 224, 161, 154, 216, 192, 94, 151, 227, 117, 116, 115, 607, 21, 169, 224, 125, 192, 458, 151, 122, 233
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure
- **DF2:**
  - CAPEC- 292, 607, 148, 22, 594, 169, 272, 125, 224, 161, 154, 216, 192, 94, 151, 227, 117
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure

CAPEC IDs and STRIDE threats for External Services

The following are the CAPEC IDs and STRIDE threats identified for all external services including monitor services and remote service supply:

- **Responsible Actor Server, Power Distributor Service, Smart Parking Service, Software Update Server:**
  - CAPEC- 184, 188, 74, 114, 112, 22, 443, 549, 173, 28, 160, 137, 624, 94, 586, 227, 460, 438, 116, 115, 212, 439, 49, 565, 244, 607, 70, 441, 21, 55, 242, 153, 240, 572, 123, 111, 440, 122, 113, 248, 388, 563
  - STRIDE Threats: Spoofing, Denial Of Service, Elevation of Privilege, Tampering, Information Disclosure

Table 5.1: CAPEC IDs and STRIDE Threats for Different Classes and Individuals in HArMoNICS

Class	**Individuals**	**#Threats**
IoT Device	Concentrator, InHouseDisplay, SmartMeter	44 STIDE
IoT Device	SmartParkingLot	33 STIDE
Sensor, Actuator	Smart Bulbs, Air Quality Sensor, Door Sensor, EnOcean Thermometer, Light Sensors	26 STIDE
Data Store	IntranetDataStore	9 TDE
Data Store	IoTDataStore	19 TIDE
IoT Gateway	Access Point, FogNode1, FogNode2	25 STIDE
Network	Internet	17 STIDE
Network	DMZ, Intranet, IoTSegment	32 STIDE
Physical Entity	SmartBuilding	10 STID
Internal Service	ABC server, Air Quality Monitor	62 STIDE
Publicly Accessible Service	DNS server, Reverse proxy, Web server	62 STIDE
Security Service	VPN Server, Router Firewall	40 STIDE
Security Service	IDS Server	50 STIDE
Security Service	Access Control Server	66 STIDE
Smart Device	Wearable Device	10 STID
Smart Device	Personal Device	17 STIDE
Data Flow	DF1, DF3, DF4, DF5, DF6	29 STIDE
Data Flow	DF2	17 STIDE
Monitor Service	Responsible Actor Server	44 STIDE
Remote Service Supply	Power Distributor Service, Smart Parking Service, Software Update Server	44 STIDE

Chapter 6

Conclusion and Future Work

This thesis aimed to design and implement an ontology-based framework for automated threat modeling in IoT environments.
The main contributions of this work can be categorized into three areas:

- Development of an IoT Ontology: A comprehensive ontology was created using OWL 2 and Protégé, providing a formal structure to represent IoT systems and their security aspects.

- Automated Threat Identification: SWRL inference rules were implemented to automate the threat modeling process. These rules are based on the relationships and properties of the defined components. They enable automated reasoning that identifies relevant CAPEC attack patterns and maps them to STRIDE categories for each system element. This approach significantly reduces the need for manual analysis and supports the risk assessment process.

- Validation Through Case Study: The framework was applied to the HArMoN-ICS infrastructure, a digital model of a smart polygeneration microgrid, to assess its practical effectiveness. The case study validated the framework's ability to model all the components of the system using the ontology and to automatically create a threat model identifying CAPEC attack patterns and corresponding STRIDE categories associated with specific components and data flows in the IoT infrastructure. The results confirm the IoT ontology's potential to provide a structured view of the system under analysis and its potential threats. The framework can support security teams in identifying mitigation more effectively.

The framework demonstrates the potential of this ontology-driven framework to significantly improve the threat modeling in IoT environments, however, some limitations were identified.

First, the reliance on publicly available knowledge bases like CAPEC introduces certain constraints. These sources might not always capture the latest threats, particularly in IoT environments, where the scenario evolves rapidly. These databases are also generic and they often lack the detailed information needed to accurately assess threats in the specific context of application.
Additionally, the SWRL inference rules identify only known threats based on defined attack patterns. This means that these rules currently cannot recognize unknown threats and they do not include social engineering or physical security threats which may be important in some IoT infrastructures.

Considering the limitations above, several future developments and enhancements are now presented.
One significant area for improvement is expanding the sources of threat information by including data repositories such as Cyber Threat Intelligence (CTI) and IoT-specific threat databases to extend the ontology's coverage of potential threats [43]. Additionally, integrating machine learning techniques would allow the framework to consider unknown attack patterns and threats [44]. Finally, adding real-time threat monitoring would allow the framework to adapt to changes in the IoT systems under analysis [45].

In conclusion, this thesis presented an ontology-driven framework for automated threat modeling in IoT environments. The developed ontology provides a structured vocabulary to represent IoT systems, linking specific threats to system components and was designed to integrate with the cybersecurity process. The framework enhances IoT security, offering valuable support from initial design to operational evaluations.
The main strength of the framework is the use of SWRL inference rules combined with the IoT ontology, enabling automated threat detection that aligns with advanced methodologies. The successful application of the framework to the HArMoNICS infrastructure demonstrated its ability to generate threat models, linking each component to attack patterns and threat categories.
Starting from ontology-based research in ICT security, this work provides a solid foundation for the risk assessment process applied to the Internet of Things making it a valuable tool to improve IoT security.

Bibliography

[1] Nan Sun, Ming Ding, Jiaojiao Jiang, Weikang Xu, Xiaoxing Mo, Yonghang Tai, and Jun Zhang. «Cyber Threat Intelligence Mining for Proactive Cybersecurity Defense: A Survey and New Perspectives». In: *IEEE Communications Surveys and Tutorials* 25.3 (2023), pp. 1748–1774. DOI: 10.1109/COMST.2023.3273282 (cit. on p. 1).

[2] Igor Linkov and Alexander Kott. «Fundamental Concepts of Cyber Resilience: Introduction and Overview». In: *Cyber Resilience of Systems and Networks*. Ed. by Alexander Kott and Igor Linkov. Cham: Springer International Publishing, 2019, pp. 1–25. ISBN: 978-3-319-77492-3. DOI: 10.1007/978-3-319-77492-3_1. URL: https://doi.org/10.1007/978-3-319-77492-3_1 (cit. on p. 1).

[3] A. Holst. *Number of Internet of Things (IoT) connections worldwide from 2022 to 2023, with forecasts from 2024 to 2033*. en. URL: https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/ (cit. on p. 2).

[4] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC). *Internet of Things (IoT) – Reference Architecture*. Standard ISO/IEC 30141:2018. Geneva, Switzerland: International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), 2018 (cit. on pp. 2, 25–30).

[5] Geoffrey Nunberg. *The Advent of the Internet*. 12th April, Courses. 2012. URL: https://www.scirp.org/reference/referencespapers?referenceid=1482944 (cit. on p. 3).

[6] Evangelos A. Kosmatos, Nikolaos D. Tselikas, and Anthony C. Boucouvalas. «Integrating RFIDs and Smart Objects into a Unified Internet of Things Architecture». In: *Advances in Internet of Things* 1.1 (2011), pp. 5–12. DOI: 10.4236/ait.2011.11002. URL: https://www.scirp.org/journal/paperinformation?paperid=4696 (cit. on p. 4).

[7] Somayya Madakam, R Ramaswamy, and Siddharth Tripathi. «Internet of Things (IoT): A Literature Review». In: *Journal of Computer and Communications* 3 (2015), pp. 164–173 (cit. on pp. 4–7).

[8] R. Aggarwal and M. Lal Das. «RFID Security in the Context of "Internet of Things"». In: *First International Conference on Security of Internet of Things*. Kerala, India, 2012, pp. 51–56. DOI: 10.1145/2490428.2490435. URL: http://dx.doi.org/10.1145/2490428.2490435 (cit. on pp. 4, 5).

[9] Trevor Harwood. *Internet of Things (IoT) History*. Accessed: 2024-08-26. 2019. URL: https://www.postscapes.com/iot-history/ (cit. on p. 4).

[10] Matthew Gigli and Simon Koo. «Internet of Things: Services and Applications Categorization». In: *Advances in Internet of Things* 1.2 (2011), pp. 27–31. DOI: 10.4236/ait.2011.12004. URL: http://dx.doi.org/10.4236/ait.2011.12004 (cit. on p. 5).

[11] John Doe, Jane Smith, and Alice Brown. «Risk-based resource allocation and optimization of aviation security». In: *Journal of Air Transport Management* 28 (2012), pp. 15–23. DOI: 10.1016/j.jairtraman.2012.01.005. URL: https://www.sciencedirect.com/science/article/pii/S2212671612000200 (cit. on p. 5).

[12] D. Moeinfar, H. Shamsi, and F. Nafar. «Design and Implementation of a Low-Power Active RFID for Container Tracking at 2.4 GHz Frequency». In: *Advances in Internet of Things* 2.2 (2012), pp. 13–22. DOI: 10.4236/ait.2012.22003 (cit. on p. 5).

[13] Bicknell. *IPv6 Internet Broken, Verizon Route Prefix Length Policy*. 2009 (cit. on p. 5).

[14] K. Pahlavan, P. Krishnamurthy, A. Hatami, M. Ylianttila, J.P. Makela, R. Pichna, and J. Vallstron. «Handoff in hybrid mobile data networks». In: *IEEE Personal Communications* 7.2 (2000), pp. 34–47. DOI: 10.1109/98.839330 (cit. on p. 6).

[15] Xian-Yi Chen and Zhi-Gang Jin. «Research on Key Technology and Applications for Internet of Things». In: *Physics Procedia* 33 (2012). 2012 International Conference on Medical Physics and Biomedical Engineering (ICMPBE2012), pp. 561–566. ISSN: 1875-3892. DOI: https://doi.org/10.1016/j.phpro.2012.05.104. URL: https://www.sciencedirect.com/science/article/pii/S1875389212014174 (cit. on p. 7).

[16] Cameron Faulkner. «What is NFC? Everything you need to know». In: *Tech Radar* (May 2017). URL: https://www.techradar.com/news/what-is-nfc (cit. on p. 7).

[17] Th. Arampatzis, John Lygeros, and Stamatis Manesis. «A Survey of Applications of Wireless Sensors and Wireless Sensor Networks». In: *Intelligent Control, 2005. Proceedings of the 2005 IEEE International Symposium on, Mediterranean Conference on Control and Automation*. IEEE. Mediterrean Conference on Control and Automation: IEEE Xplore, July 2005. DOI: 10.1109/.2005.1467103. URL: https://doi.org/10.1109/.2005.1467103 (cit. on p. 8).

[18] Michael Chorost. «The Networked Pill». In: *MIT Technology Review* (Mar. 2008) (cit. on p. 8).

[19] Ashish Ghosh, Debasrita Chakraborty, and Anwesha Law. «Artificial intelligence in Internet of things». In: *CAAI Transactions on Intelligence Technology* 3.4 (2018), pp. 208–218 (cit. on p. 8).

[20] Rossouw von Solms and Johan van Niekerk. «From Information Security to Cyber Security». In: *Computers and Security* 38 (2013), pp. 97–102 (cit. on pp. 8–10).

[21] M Jimenez, P Sanchez, F Rosique, B Alvarez, and A Iborra. «A tool for facilitating the teaching of smart home applications». In: *Computing Applications in Engineering Education* (2011). DOI: 10.1002/cae.20521. URL: http://dx.doi.org/10.1002/cae.20521 (cit. on p. 9).

[22] Wenjun Xiong and Robert Lagerström. «Threat modeling – A systematic literature review». In: *Computers and Security* 84 (2019), pp. 53–69. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2019.03.010. URL: https://www.sciencedirect.com/science/article/pii/S0167404818307478 (cit. on p. 10).

[23] Exabeam Team. *Top 8 Threat Modeling Methodologies and Techniques*. https://www.exabeam.com/blog/infosec-trends/top-8-threat-modeling-methodologies-and-techniques/. Accessed: 2024-08-31. 2024 (cit. on p. 11).

[24] Natalya F Noy and Deborah L McGuinness. «Ontology Development 101: A Guide to Creating Your First Ontology». In: Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880. 2001 (cit. on p. 12).

[25] Fabio De Rosa, Nicolo Maunero, Luca Nicoletti, Paolo Prinetto, and Martina Trussoni. «Ontology for Cybersecurity Governance of ICT Systems». In: (2022) (cit. on pp. 12, 17–19).

[26] MITRE. *Common Attack Pattern Enumeration and Classification (CAPEC)*. Accessed: 2024-08-18. 2023. URL: https://capec.mitre.org/ (cit. on pp. 14, 35).

[27] Microsoft. *STRIDE: A Model for Identifying Computer Security Threats*. Accessed: 2024-08-18. 2023. URL: https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-threats (cit. on pp. 14, 35).

[28] Microsoft. *Getting started with the threat modeling tool*. Accessed: 2024-09-02. 2024. URL: https://docs.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-getting-started (cit. on p. 14).

[29] Fabio De Rosa, Nicolò Maunero, Paolo Prinetto, Federico Talentino, and Martina Trussoni. «ThreMA: Ontology-based Automated Threat Modelling for ICT Infrastructures». In: *IEEE Access* 4 (2022), pp. 1–13. DOI: 10.1109/ACCESS.2017.DOI (cit. on pp. 17–19, 26, 29, 31–35, 40, 44).

[30] Nicolo Maunero, Fabio De Rosa, and Paolo Prinetto. «Towards Cybersecurity Risk Assessment Automation: an Ontological Approach». In: *Proceedings of the Conference on Cybersecurity* (2023) (cit. on pp. 17, 19).

[31] Blake E. Strom, Andy Applebaum, Doug P. Miller, Kathryn C. Nickels, Adam G. Pennington, and Cody B. Thomas. *MITRE ATTaCK: Design and Philosophy*. Technical Report. MITRE Corporation, 2018 (cit. on p. 17).

[32] Stanford Center for Biomedical Informatics Research. *Protégé*. https://protege.stanford.edu. Accessed: 2024-09-01. 2024 (cit. on p. 17).

[33] Valentina Casola, Alessandra De Benedictis, Carlo Mazzocca, and Rebecca Montanari. «Toward Automated Threat Modeling of Edge Computing Systems». In: *2021 IEEE International Conference on Cyber Security and Resilience (CSR)*. IEEE. 2021, pp. 135–140 (cit. on p. 20).

[34] Jason R.C. Nurse, Sadie Creese, and David De Roure. «Security Risk Assessment in Internet of Things Systems». In: *IT Pro* 20.5 (2017), pp. 20–26 (cit. on p. 21).

[35] Bruno A. Mozzaquatro et al. «An Ontology-Based Cybersecurity Framework for the Internet of Things». In: *Sensors* 18.9 (2018), p. 3053 (cit. on p. 22).

[36] Valentina Casola, Alessandra De Benedictis, Massimiliano Rak, and Umberto Villano. «Toward the automation of threat modeling and risk assessment in IoT systems». In: *Internet of Things* 7 (2019), p. 100056 (cit. on p. 22).

[37] Kamalanathan Kandasamy, Sethuraman Srinivas, Krishnashree Achuthan, and Venkat P Rangan. «IoT cyber risk: a holistic analysis of cyber risk assessment frameworks, risk vectors, and risk ranking process». In: *EURASIP Journal on Information Security* 2020.1 (2020), pp. 1–18 (cit. on p. 23).

[38] Mardiana binti Mohamad Noor and Wan Haslina Hassan. «Current research on Internet of Things (IoT) security: a survey». In: *Computer Networks* 148 (2019), pp. 283–294 (cit. on p. 23).

[39] Muhammad Aslam Jarwar, Jeremy Watson, Uchenna Daniel Ani, and Stuart Chalmers. «Industrial Internet of Things Security Modelling using Ontological Methods». In: *Proceedings of the 12th International Conference on the Internet of Things (IoT)*. ACM, 2022, pp. 163–170. DOI: 10.1145/3567445.3571103 (cit. on p. 24).

[40] Ostering. *CAPEC-STRIDE Mapping Project*. https://ostering.com/blog/2022/03/07/capec-stride-mapping/. Accessed: 2024-08-17. 2022 (cit. on pp. 35, 40).

[41] Gianluca Roascio et al. «HArMoNICS: High-Assurance Microgrid Network Infrastructure Case Study». In: *IEEE Access* 10 (2022), pp. 115372–115383. DOI: 10.1109/ACCESS.2022.3218412 (cit. on pp. 55, 56, 59–64).

[42] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. «Pellet: A practical OWL-DL reasoner». In: *Journal of Web Semantics* 5.2 (2007), pp. 51–53 (cit. on p. 66).

[43] S Smolyakova and E Khodayarseresht. «Traditional IOCs Meet Dynamic App-Device Interactions for IoT-Specific Threat Intelligence». In: *IEEE Internet of Things Journal* (2024) (cit. on p. 75).

[44] MF bin Zolkipli and SSBA Harris. «The Evolution of Threat Intelligence: Trends and Innovations in Cyber Defense». In: *International Journal of Applied Engineering and Management* (2023) (cit. on p. 75).

[45] MS Pour, D Watson, and E Bou-Harb. «Sanitizing the IoT Cyber Security Posture: An Operational CTI Feed Backed by Internet Measurements». In: *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. 2021. URL: https://ieeexplore.ieee.org/abstract/document/9505129/ (cit. on p. 75).