POLITECNICO DI TORINO

Master's Degree in Computer Engineering

Cybersecurity

Master's Degree Thesis

Automatic attack graph reasoning and risk analysis

Supervisors:

Prof. Cataldo Basile

Ing. Alessio Viticchié

Candidate:

Giuseppe Venezia

Academic Year 2023/2024

Torino

Table of Contents

List of Figures

Chapter 1

Introduction

The growing trend of digitalization and increased reliance on information technology has made cybersecurity a major global concern in recent years. Major industrial companies overseeing crucial infrastructures are likewise impacted by the growing sophistication and incidence of cyber attacks, in addition to individuals and small businesses. Industrial Control Systems (ICS), formerly isolated, are becoming increasingly connected to IT networks, increasing their susceptibility to hackers. This thesis investigates automated attack graph analysis and risk assessment in industrial systems, emphasizing the convergence of operational technology (OT) and information technology (IT).

The decision to explore this topic was motivated by an interest in the challenges of protecting critical infrastructures; targeted cyberattacks have affected several industrial sectors, making this an increasingly important topic. Because these systems were often designed without considering modern digital threats, they are fragile, and new techniques for ongoing risk management need to be implemented. Given that it addresses physical asset protection, data security, and vital economic production processes, this topic is crucial.

The primary objective of this thesis is to create a continuous risk analysis engine that can locate weak points and vulnerabilities in industrial networks and anticipate and counteract such attacks before they cause the system serious harm. The effort emphasizes building and using automated attack graphs that can be produced by a tool for the continuous monitoring of network security conditions and the self-aware detection of vulnerabilities. In the face of a dynamic threat environment, this approach ensures that industrial systems remain up to date and enables proactive threat management.

From a methodological perspective, the research is based on the use of automated reasoning models to assess the likelihood and impact of attacks. Several risk metrics are analyzed and integrated, allowing a more accurate and dynamic assessment of potential vulnerabilities.

The thesis is divided into six chapters. Following this introduction, the theoretical background is covered in detail in the second chapter, which also includes an overview of OT and IT technologies as well as the threat landscape. The model used for risk analysis is defined in the third chapter, and its implementation is covered in detail in the fourth chapter. The experimental evaluation results are compared between various network scenarios and security methods in the fifth chapter. The sixth chapter provides a viewpoint on the results and potential avenues for further research, thus wrapping off the work.

In summary, although it is still in its early stages, the work has led to the development of a framework for risk analysis that holds significant potential for applications in the future to increase industrial network resilience and proactive threat management.

Chapter 2

Background

Nowadays, security, especially cybersecurity, is a very important aspect that affects all the major entities around us, starting from individual security, such as accounts, emails, and bank accounts, to the security of small and large companies and industries that drive the world. With the ever-more rapid advancement of technology, we now have a variety of tools that allow us to do amazing things. However, some malicious individuals use these tools to their advantage and cause harm to one or many. This issue has always been underestimated over time because, being something that does not directly and physically harm a person, it is difficult to raise awareness about it. Nevertheless, cybersecurity is not a problem to be overlooked or considered only when necessary, because by then it would be too late. It must be analyzed and considered from the outset because every minor error or oversight can be fatal. In this regard, particularly in the context of an industrial network, there is a need to find a way to always know what the weak points in the network are and how a malicious individual could attack and cause damage, whether material or immaterial, such as accessing a large portion, or with particular misfortune, all of the sensitive data, creating a service interruption (DoS), spreading a virus, and more. The key point is to know how to act to prevent this series of unpleasant situations rather than trying to recover and repair them with little hope. To effectively prevent this, we need to understand our network infrastructure, its limits, its strengths, and know how to act to stay safe. Subsequently, we will analyze the main methods to defend ourselves and delve into the context, that is the environment in which we will conduct our analyses.

2.1 Continuous risk analysis

It is crucial to have a sophisticated and multi-layered cybersecurity strategy in place to safeguard a company from bad actors in the face of a constantly changing threat landscape. Even though the company is secure right now, things could change in an instant. Continuous risk management is the most important component of a long-term commitment to safeguarding the organization's mission-critical business data. Cybersecurity risk assessments seek to "identify, estimate, and prioritize risk to organizational operations, assets, individuals, other organizations, and the nation" due to the utilization and functioning of information systems, according to the National Institute of Standards and Technology (NIST) Cybersecurity Framework (CSF). Even while a first evaluation might show that a company is headed in the correct direction, later evaluations might point to weaknesses. Therefore, ongoing risk management needs to be part of every organization's routine activities. Completing a compliance gap assessment or running one more check on a checklist is not continuous risk management. It is not a yearly, periodic, or one-time task. It entails integrating risk analysis into the organization's core processes, integrating security into an organization's technologies, and keeping up with the rapidly evolving cybersecurity landscape [1, 2]. Below there is a summary diagram in Figure 2.1

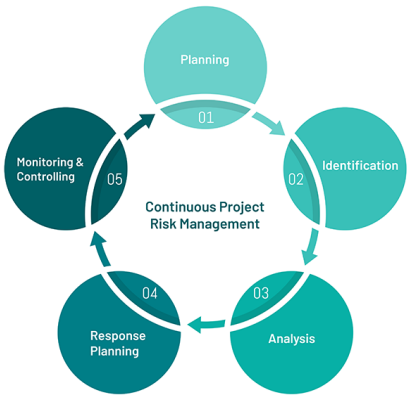

Figure 2.1: Continuous Risk Analysis

Key Elements of Continuous Risk Assessment

To facilitate rolling risk assessments, four components must be present [3]. They are:

- **Continuous Assessment and Monitoring:** the maintenance of current and pertinent risk profiles is ensured by ongoing risk assessment and monitoring. This

method guarantees that risk considerations are ingrained in daily operations by smoothly integrating risk management into current business processes.

- **Dynamic Risk Response and Evaluation:** businesses need to be able to quickly adjust to shifting risk environments. Organizations can analyze risks in real-time or near real-time, utilizing current data and dynamic scenarios to ascertain the impact and likelihood of hazards, and then create proactive plans for mitigation and management.
- **Real-Time Risk Insights and Integration:** a comprehensive understanding of hazards is ensured by integrating risk management with other business processes, such as project management, cybersecurity management, and regulatory compliance.
- **Proactive and Continuous Risk Identification:** by employing several techniques and instruments including automated monitoring, stakeholder feedback, and vulnerability analysis, organizations can consistently identify risks and mitigate their effects early on, hence decreasing the probability of major issues.
- **Adaptation and Resilience:** adaptive system design enables organizations to enhance their resilience and react promptly to shifts in threats. Using automated technologies, dashboards, and regular reporting, continuous monitoring keeps the organization's risk profile current.

Advantages of Continuous Risk Assessments

Using an ongoing risk assessment process has several benefits. It facilitates the prompt identification of hazards as they arise or change, encouraging a proactive strategy that lowers the possibility of serious problems. Organizations become more flexible and agile, enabling them to react quickly to modifications and enhancements to their risk-reduction plans. Constant risk assessment improves decision-making by giving insights into the state of risk in real-time. This makes it easier to prioritize risk mitigation initiatives, allocate resources effectively, and make well-informed decisions. By incorporating risk assessment into regular operations, an organization may develop a culture of risk awareness and make sure that risk management is a shared duty. This methodology moreover expedites the creation and execution of focused risk mitigation strategies, augmenting the organization's resistance to disturbances and fortifying its capacity to endure and rebound from unfavorable incidents. Reputation, stakeholder confidence, and competitive

advantage are all improved by proactively managing risks; costs are reduced by reducing the possibility of monetary losses, legal ramifications, and reputational harm. Establishing credibility and trust with stakeholders, such as clients, financiers, and government agencies, further fortifies the company [2].

Steps to Adopt a Continuous Risk Assessment Process

To effectively implement a continuous risk assessment process [3], organizations should follow these steps:

1. **Define Risk Categories and Evaluation Criteria:** Establish criteria for evaluating and ranking the risks in each of the organization's important risk categories.
2. **Establish a Risk Management Framework:** Establish roles and duties, define your risk appetite, set specific goals, and create rules and processes for risk management.
3. **Implement a Risk Monitoring System:** Front-line managers should be included in the process of identifying and monitoring risks by using technology and data-driven solutions for real-time risk monitoring and information gathering.
4. **Foster a Risk-Aware Culture:** Encourage staff members at all levels to report hazards, exchange ideas, and actively engage in the risk assessment process in order to raise risk awareness and accountability throughout the company.
5. **Continuous Improvement and Monitoring:** Maintain a regular review and improvement of the risk assessment procedure, keep up with internal and external elements influencing the risk environment, and incorporate risk assessment into the processes of strategic planning and decision-making.
6. **Leverage Advanced Technologies:** To improve risk identification, assessment, and reaction planning, make use of artificial intelligence, data analysis, risk dashboards, automated monitoring systems, and simulations.

Limitations of Continuous Risk Strategy

The continuous risk strategy has several drawbacks even though it provides flexibility and agility in handling new threats. The implementation of this strategy necessitates

substantial time, effort, and resource commitment, which may provide difficulties for businesses with constrained resources or intricate risk environments. It takes initiative and forward-thinking to recognize new hazards and include them in the risk assessment process. Accurate and timely data are essential for effective risk management, but they can be difficult to obtain. If handled carelessly, making decisions quickly based on current risk knowledge can lead to complications and mistakes. A culture of adaptation and receptivity to novel risk management techniques is also necessary for the implementation of a continuous risk strategy, as it may encounter ignorance or opposition. Furthermore, it can be difficult to match the continuous risk strategy with the current risk management procedures, particularly in large or complicated organizations. In order to guarantee comprehensive risk management, it is critical to strike a balance between short-term and long-term strategic risks.

Notwithstanding these drawbacks, the continuous risk strategy has certain advantages. To get the greatest outcomes, organizations should carefully customize their risk management strategy to fit their unique situation, risk profile, and resource availability [3].

2.2 Operational Technology

Operational technology (OT) is the hardware and software used in industrial settings to monitor and control processes, infrastructure, and equipment. On the other hand, information technology (IT) includes cloud computing, enterprise data centers, and information processing technologies. OT devices control physical processes, whereas IT systems manage data and applications.

OT security is defined by Gartner as “Practices and technologies used to protect people, assets, and information, monitor and/or control physical devices, processes, and events, and initiate state changes to enterprise OT system”. OT security solutions use a wide range of technologies, including next-generation firewalls (NGFWs) and security information and event management (SIEM) systems, to offer access control and monitoring. Operational technology (OT) includes, among other things, robots, industrial control systems (ICS), supervisory control and data acquisition (SCADA) systems, programmable logic controllers (PLCs), and computer numerical control (CNC) equipment. The main contrast between IT and OT is that the former is concentrated on an organization’s informational operations, while the latter is concentrated on machinery and production [4] [5].

When it comes to industrial equipment, the OT department puts worker safety and

production output first. The team places a strong emphasis on machine uptime and maintenance because OT performance is essential to business profitability. OT devices are often custom-built, may run proprietary protocols, and have specialized software. Since they are in charge of vital infrastructures, they are made to last a long time and can frequently run uninterruptedly for years or even decades. As a result, OT systems and devices may have many software vulnerabilities and receive fewer updates than IT systems.

Key differences between OT and IT

Network infrastructure components including switches, routers, and wireless technology are shared by both OT and IT. To establish a strong network foundation, OT networks can thus profit from the discipline and knowledge that IT has acquired over the years with common network administration and security measures [6].

But there are some significant variations:

- **Form Factor:** smaller, modularized form factors for OT network devices allow for a variety of mounting choices, such as in cars, on walls, light poles, trains, or incorporated in other equipment.

- **Hardening:** to survive harsh industrial circumstances, such as resistance to shock, vibration, water, extreme temperatures, and corrosive environments, OT network infrastructure frequently needs to be ruggedized.

- **Network Interfaces:** OT devices may be able to connect industrial IoT (IIoT) equipment over specialized networks like LoRaWAN or Wi-SUN, depending on their intended use.

- **Protocols:** IoT sensors and machines are connected by OT network devices, which use communication protocols that are uncommon in typical IT networks. Thus, a range of protocols, such as Profinet, Modbus, and Common Industrial Protocol (CIP), must be supported by industrial networking solutions.

An example of the IT-OT structure is represented in Figure 2.2

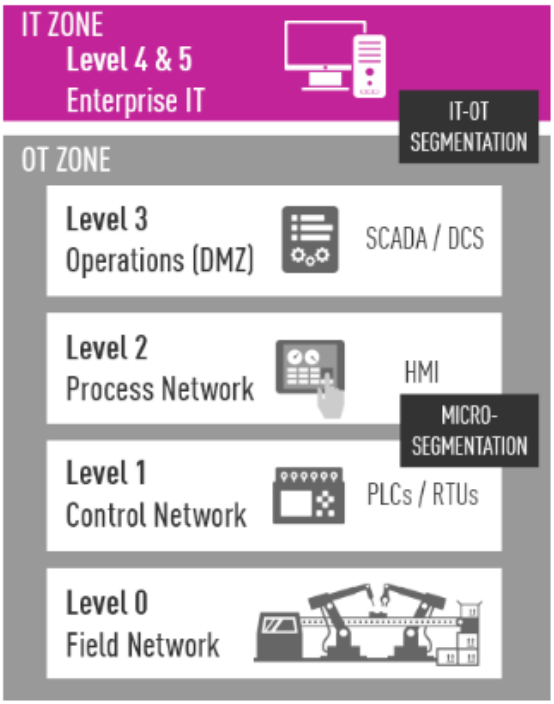

Figure 2.2: IT-OT structure

Components

Industrial control systems (ICS) constitute a fundamental component of operational technology (OT). These systems consist of a multitude of controllers, networks, devices, and systems that monitor various industrial processes. The two most widely used types are Supervisory Control and Data Acquisition (SCADA) and Distributed Control Systems (DCS).

Sensor data is often distributed across multiple sites, and SCADA systems collect and transfer this data to a central computer for management and control. These systems play a crucial role in remote monitoring of large-scale operations such as oil and gas pipelines, water treatment, and electricity production.

Distributed control systems (DCS) oversee local controllers or devices of production systems, on the other hand, from a single location. SCADA systems are designed for large, dispersed networks; in contrast, DCS emphasises on automating tasks inside a restricted area, such as manufacturing facilities or refineries. DCS enhances industrial processes' efficiency and safety by offering real-time data and control.

The tiniest parts of operational technology are various sensors, monitors, actuators, and other technologies that are mounted on or next to OT equipment. Pipelines, fans, programmable logic controllers (PLC), industrial robots, remote processing units (RPU), and generators are some examples of this equipment. The Industrial Internet of Things (IIoT) allows for increased automation, data analysis, and communication [4].

OT-IT Convergence

Information technology (IT) and operational technology (OT) systems must communicate with one another for digital innovation to occur. IT network components like processors, storage, and system management are rapidly being linked to OT network components like industrial networks, SCADA, and control systems. Data gathered by IIOT and physical equipment can be utilized to discover issues and improve productivity through IT-OT integration.

Nevertheless, by utilizing an IT network to link an isolated, previously unconnected OT network to the internet, the OT network and its associated OT equipment become vulnerable to all available threats. OT systems are usually not secure since their initial design was based on the assumption that they would not be susceptible to external threats. Moreover, the expansion of remote access to OT networks by third-party providers increases the attack surface and introduces new vulnerabilities [4, 6].

2.2.1 Security

As evidenced by the results of the most recent poll conducted by the SANS Institute, OT security specialists believe that the risk levels are high. Nearly 74% of OT organizations reported a malware intrusion in the previous year, according to the Fortinet State of Operational Technology Report, demonstrating how serious OT security issues are. These incursions have harmed intellectual property, productivity, income, brand trust, and personal safety [7, 8].

Historically, operational technology security also referred to as security by obscurity, has mostly depended on the independent nature of OT installations. Compared to IT, OT has different goals and a separate infrastructure that needs to be protected. The "Confidentiality, Integrity, Availability" (CIA) design axis is commonly used in IT systems and highlights the need to safeguard user access only after crucial data has been verified. OT systems, on the other hand, cannot function well without "Real-time Control, Functionality

Change Flexibility, Availability, Integrity, Confidentiality" (CAIC), which puts the user's information display first and addresses correctness or confidentiality later. The following challenges have an impact on OT system security:

- **Inherent Insecurity:** OT components are frequently designed with functional aims in mind, rather than meeting fundamental IT security needs. These components might therefore be prone to cyberattacks and unsecure by design;
- **Vendor Dependency:** since most people don't know much about industrial automation, most businesses rely a lot on their OT vendors. This results in vendor lock-in, which makes it more difficult to apply security updates;
- **Critical Assets:** since OT systems are frequently a part of the nation's important infrastructure, they must have improved security features to monitor and regulate vital industrial activities.

The five most important criteria for an excellent IT-OT security system are:

1. Identify assets, classify them, and prioritize their value.
2. Dynamically segment the network.
3. Analyze traffic for threats and vulnerabilities.
4. Control identity and access management.
5. Secure both wired and wireless access.

A security fabric offers three advantages for reliable and efficient IT-OT security [4]:

- **Visibility:** find every device connected to any location on the IT-OT network, assess the level of trust, and keep an eye on the behavior to keep the trust intact. Ascertain active device and traffic profiling and define the attack surface. Traffic visibility provides actionable intelligence, which gives OT security teams the ability to specify permissible traffic, ports, protocols, applications, and services.
- **Control:** make that every OT system and subsystem carries out the specified task. Using multi-factor authentication ensures that the right persons are granted the right access and permissions. Zones of control and layers of separation are provided by network segmentation and micro-segmentation. Threats on the OT network are identified using sandboxing, and damage is stopped by automatic quarantine.

- **Continuous Monitoring:** learn what, where, when, who, and how by continuously analyzing behavior in OT networks to acquire intelligence about known and new dangers. Logging, reporting, analytics, and system-wide activity evaluation are facilitated by a central security tool. Along with event management, security orchestration automation, and response capabilities, it also offers security information. To guarantee ongoing protection, OT security insights are obtained through threat assessments and behavior analysis of users and devices.

2.3 SCADA

Supervisory Control and Data Acquisition (SCADA) systems control, monitor, and analyze industrial equipment and processes. These systems enable the remote and on-site collection of data from industrial equipment. They are composed of both hardware and software components. This makes it possible for businesses to obtain turbine data remotely and control industrial sites—like wind farms—without having to physically be present [9]. Using a SCADA system, businesses can:

- Oversee operations both locally and remotely.
- Acquire, analyze and display real-time data.
- Engage directly with industrial machinery, including motors, pumps, valves, and sensors.
- Record and archive events for future reference or report generation.

Key Components

The following are the main parts of a SCADA system [10]:

- **Sensors and Actuators:** Sensors gather information about physical properties, such as pressure or temperature, and provide it to the SCADA system. Conversely, actuators take advantage of this information to carry out particular tasks. These collectively make up the essential parts of a SCADA system.
- **Supervisory Computers:** These computers collect sensor data, process it, come to conclusions, and instruct actuators. They act as the main hub, establishing

connections to, facilitating communication with, and managing all distant terminals and devices.

- **SCADA Field Controllers:** These gadgets are linked to sensors and actuators and are situated close to the process that needs to be regulated. They gather information from sensors, transmit it to the supervisory computer, and then get instructions to transfer it to actuators from the supervisory computer. SCADA field controllers are made to function well in challenging conditions and usually feature a large number of inputs and outputs. The two primary categories are:
  - *Programmable Logic Controllers (PLCs):* Specialised industrial computers that process and deliver commands to actuators after receiving input data from sensors.
  - *Remote Telemetry Units (RTUs):* Devices that employ serial, Ethernet, or wireless communication protocols to gather data from remote areas, process it, and send it to the supervisory computer. RTUs are typically attached to sensors and actuators and placed in the field, close to the process they are controlling.
- **Human-Machine Interface (HMI):** A piece of hardware or software that interfaces between engineers and operators and the SCADA system. The HMI assists users in comprehending, monitoring, and controlling the process through the use of graphical information displays and actuator commands.
- **Communication Infrastructure:** This includes supervisory computers, field controllers, sensors, actuators, and other SCADA components, as well as the network of devices and communication protocols that are used to transfer data between them.

Why a SCADA system is essential

Among the many advantages of SCADA systems is their ability to optimize production and guarantee adherence to industry standards. They are also useful for troubleshooting since maintenance is essential to preserving an error-free, uninterrupted manufacturing flow.

Understanding the importance of a SCADA system requires taking a look back at the days of manual industrial plant monitoring. During manufacturing, personnel were required to be present on-site to supervise, manage, and resolve any new problems that arose. This method was not only expensive but also could have put the staff at risk [11].

Types of data gathered by SCADA system

If there is a connection to the plant's equipment, a SCADA system can gather a variety of data from it, such as temperature, pressure, and speed. The PLCs or RTUs then convert this raw data into understandable information so that human operators can take appropriate action.

Since SCADA systems gather data in real-time as well as historical data, they are especially helpful. While historical data is used for reporting and enhancing plant performance, live data is frequently used for maintenance needs and real-time tracking [11].

The SCADA system can transform data into comprehensible information even if the equipment it is monitoring is not made by the same company. The equipment must, above all, support an accessible communication protocol that can be used by the PLCs or RTUs. This feature makes it possible for information to be exchanged uniformly between plants. A concrete example could be the network shown below in Figure 2.3.

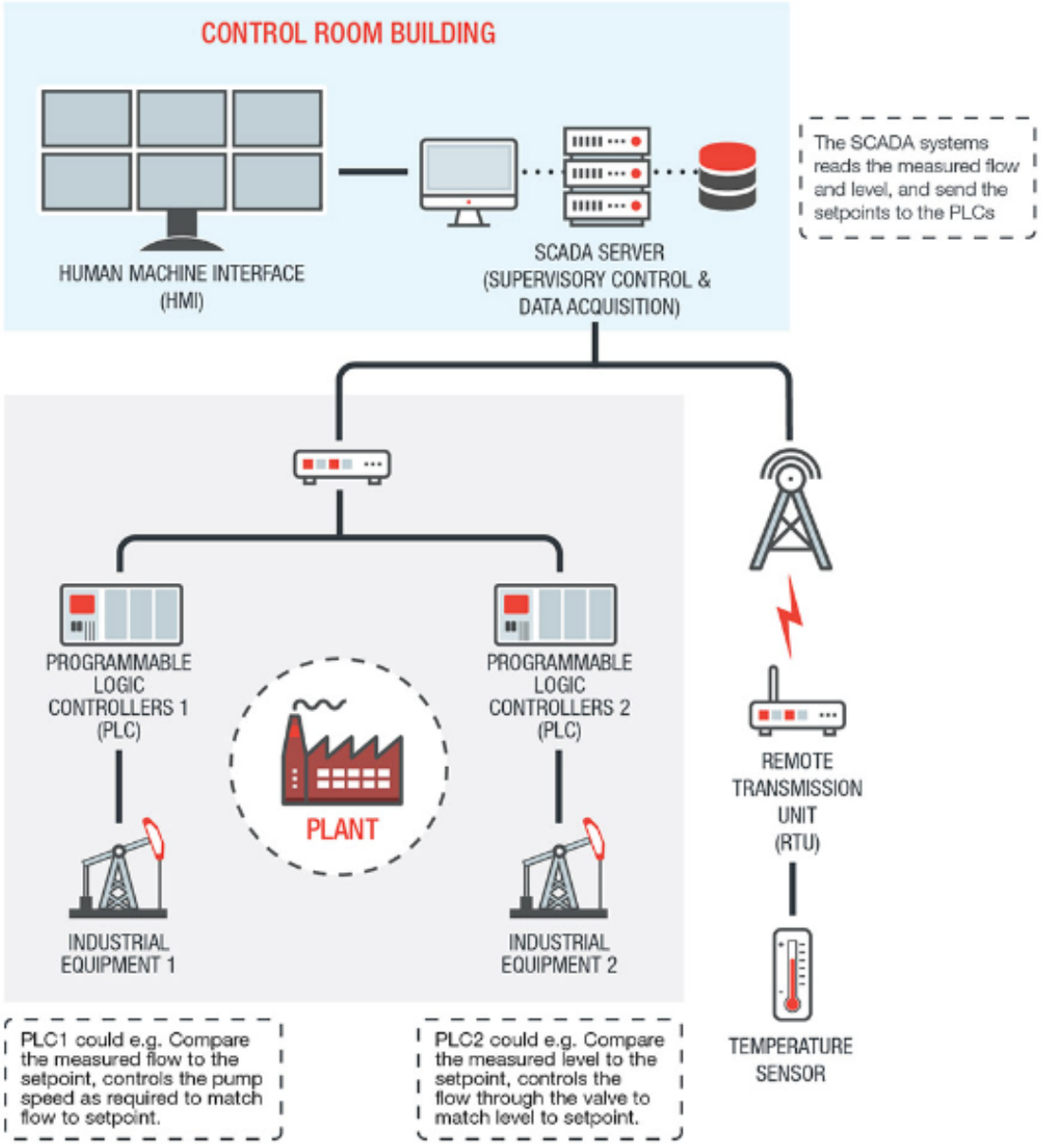

Figure 2.3: SCADA example

2.3.1 SCADA Security

Cyberattacks are one kind of security danger that SCADA systems are susceptible to. These risks can have serious repercussions, such as losing control over industrial processes, disrupting services, or even physically damaging equipment. Considering how important

SCADA security is, a comprehensive defensive strategy is necessary. It is essential to put in place a thorough security program with multiple essential components to reduce risks and defend against potential threats.

First, the division of the SCADA network into smaller segments is known as network segmentation, and it is crucial. This restricts the potential impact of any security incident and increases the difficulty of an attacker's sidestepping the network. Another crucial component is access control, which necessitates stringent guidelines to restrict who has access to SCADA systems and what they may do. Role-based access controls, user authentication, and the least privilege principle can all help achieve this.

Since security monitoring entails putting mechanisms in place to identify and address security issues, it is also essential. Examples of this include intrusion detection systems, well-defined incident response strategies, and security information and event management (SIEM) systems. Furthermore, patch management is essential to preserving SCADA system security. By keeping these systems and the software they're connected with updated with the newest security patches and upgrades, you may help stop attackers from taking advantage of known vulnerabilities.

A comprehensive security program should include security awareness training as well. Ensuring that staff and stakeholders receive training on identifying and mitigating security threats is crucial in ensuring that everyone is prepared to safeguard the system. To find and fix vulnerabilities in SCADA systems and increase its defense against possible assaults, penetration testing must be done regularly [10].

2.4 PLC vulnerabilities and deep lateral movement inside OT networks

Attack groups that target networks related to operational technology (OT) have mostly concentrated on getting beyond segmentation layers to gain access to field controllers, such as PLCs, and modify the programs (called ladder logic) that are operating on them. Researchers do warn that these controllers should be regarded as separate perimeter devices, though. Attackers may be able to move laterally through point-to-point and other non-routable connections they have with other low-level devices due to vulnerabilities in their firmware [12, 13]. Subsequently, in Figure 2.4 there is a diagram representing a PLC.

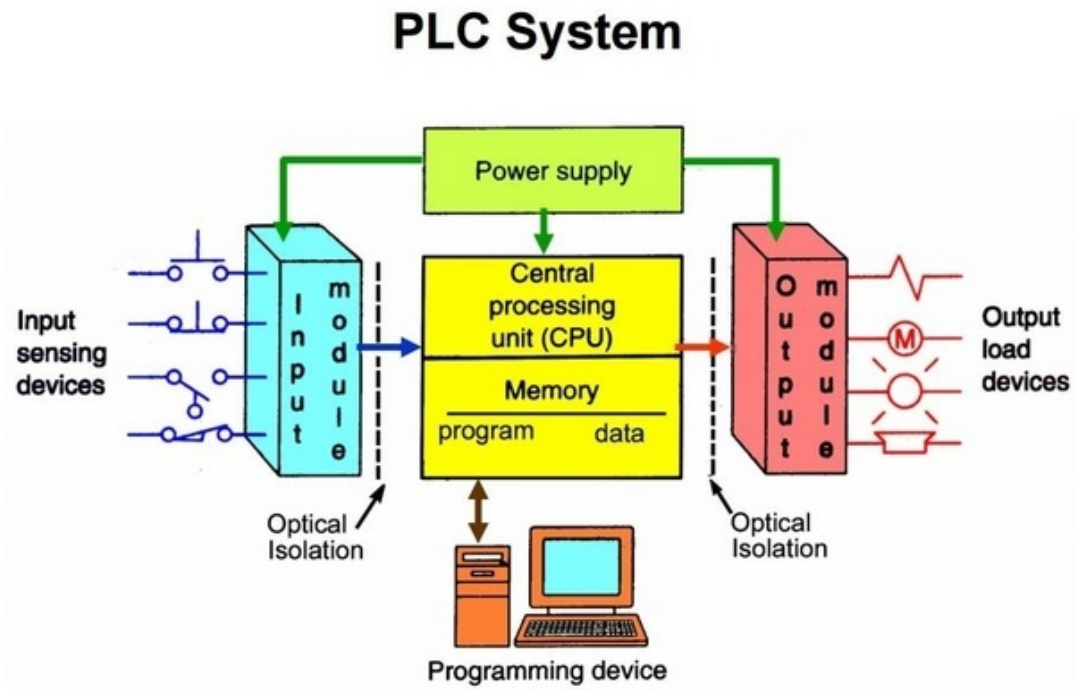

Figure 2.4: PLC System

Authentication bypass and remote code execution

A noteworthy vulnerability that allows attackers to take control of an authenticated Modbus session and perform unauthorized Modbus activities on the controller is identified as CVE-2022-45789 and has a severity rating of 8.1 on the CVSS scale. Modbus is a de facto standard for industrial devices and was first created by Modicon in the late 1970s as a data communications protocol. A proprietary version based on Modbus, called UMAS, also has this flaw.

The UMAS Enhanced Cyber Reserve mechanism, Modicon's attempt to establish authentication between the PLC and the engineering client that programs it, is the source of the problem. Due to flaws found earlier, authentication and encryption were added to the protocol in multiple iterations after it was first developed. Still, Forescout researchers have discovered that there are security flaws in the existing system.

"We found that the corrected Enhanced Cyber Reserve method is still essentially flawed after further investigation", the researchers said. First, the PLC does not explicitly connect nonces to any specific client or reservation ID; rather, it maintains them as global variables.

Furthermore, these nonces are only updated upon the conclusion of a session or receipt of an explicit request for a UMAS nonce exchange. A nonce is a secret number used just once during an operation in cryptography. A man-in-the-middle attacker can intercept these nonces and a reservation ID in the UMAS implementation and create a fake authentication request that the PLC will accept without terminating the initial session. This is possible because nonces in the UMAS implementation only refresh when a session ends [12].

A different vulnerability, identified as CVE-2022-45788, has a high criticality rating of 7.5 on the CVSS scale and permits remote code execution on the PLC. Because it makes use of an undocumented Modbus UMAS instruction (service code 0x50) that permits direct memory tampering with little traceability, this vulnerability is especially worrying.

While Schneider Electric characterizes CVE-2022-45788 as having to do with downloading malicious project files, Forescout researchers pointed out that this vulnerability involves an entirely different and undocumented set of functionalities that allow modifying internal PLC memory without requiring the download of a project or altering the PLCs' runtime state. This distinction is important since the majority of PLC attacks target the program code that drives actuators, pumps, motors, and valves that are coupled to affect physical processes. A PLC restart may be necessary if new logic is uploaded or safety mechanisms are triggered by code modifications.

On the other hand, this vulnerability lets attackers generate virtual memory pages and load a proprietary command set into them so they may manipulate internal memory blocks directly (read, write, copy, resize, etc.). This is similar to live patching an operating system process that is currently running [14].

Remote access to PLCs bypasses traditional OT security controls

The organization of the structure is divided into different functional levels in a standard OT setting. Devices that directly affect physical processes, such as actuators, sensors, and valves, are found at Level 0, the lowest level. PLCs (Programmable Logic Controllers), which oversee and govern the operations of the lower-level devices, are part of Level 1's control system [**<empty citation>**].

Level 2 SCADA (Supervisory Control and Data Acquisition) systems are situated above these. These systems provide an overview, allow for remote control and management, and monitor and collect data from the PLCs. At this stage, companies usually put in place the initial security measures, erecting a kind of barrier.

Ascending the ladder, Level 3 encompasses engineering workstations (EWSs), databases, application servers, logging servers, and operation management software. The IT network of the company, which includes finance and enterprise resource planning (ERP) technologies, is represented by Level 4. The OT and IT networks are divided by additional security measures positioned in the space between Levels 3 and 4.

An attacker would typically need to breach an engineering workstation in this structure, advance to the OT section at Level 3, and exploit this access to get access to the PLCs or take control of a SCADA system through its human-machine interface (HMI). However, because remote management is required, shortcuts are frequently taken.

For instance, wireless or cellular IoT gateways are often used to link PLCs in remote sites to a company's control center. A recent study from industrial cybersecurity company Otorio has shown that these gateways may contain remotely exploitable vulnerabilities. It is dangerous to link PLCs and SCADA systems straight to the internet, as this is sometimes the case.

The existence of packaged units (PUs), which are comprehensive subsystems made for particular uses like water treatment, chemical injection, HVAC, or gas turbines, is another aspect. These PUs are sold by manufacturers to operators, who incorporate them into their control systems. PUs serve as black boxes that are remotely managed by the system provider, however they frequently control PLCs and other components. Because of this, asset owners usually only have a limited amount of control—they can just read output data or give a few orders.

Attackers can thus gain access to a PLC through a variety of means, such as by taking the long road from enterprise IT to OT or by taking advantage of detours like remote access gateways and protocols that the asset owner has purposefully put in place. Typically, attackers look for the least resistance.

Thankfully, access to a PLC does not always result in damaging assaults happening right away. Safety instrumented systems (SIS) are typically found in industrial facilities. They autonomously monitor physical processes to make sure they stay within safe operating boundaries. There may be built-in limitations on how much particular parameters can be modified in HMIs and PLCs. Furthermore, nested deployments are possible in which a PLC simply receives data from other PLCs or PUs and does not directly operate equipment [7].

2.4.1 Deep lateral movement

Manufacturers provide a large number of Level 0-2 topologies that connect PLCs, WAN radio modems, Safety Instrumented Systems (SIS) like Figure 2.5, Packaged Units (PUs), wireless industrial gateways, and serial or point-to-point Ethernet connections. Fieldbus couplers, which act similarly to network switches but are more advanced and support a wide range of protocols and connection types, are usually used in these connections.

However, the security of these direct links is intrinsically weak. Because they are not routable by design, vendors and asset owners frequently ignore them while doing threat modeling. The system may become vulnerable as a result of this oversight.

Forescout researchers claim that because of all the connections in these systems, a PLC frequently functions as both a perimeter device and a field controller. Through what is known as "deep lateral movement", attackers can use these non-routable links to their advantage and obtain access to additional Level 1 devices via these unsafe connections.

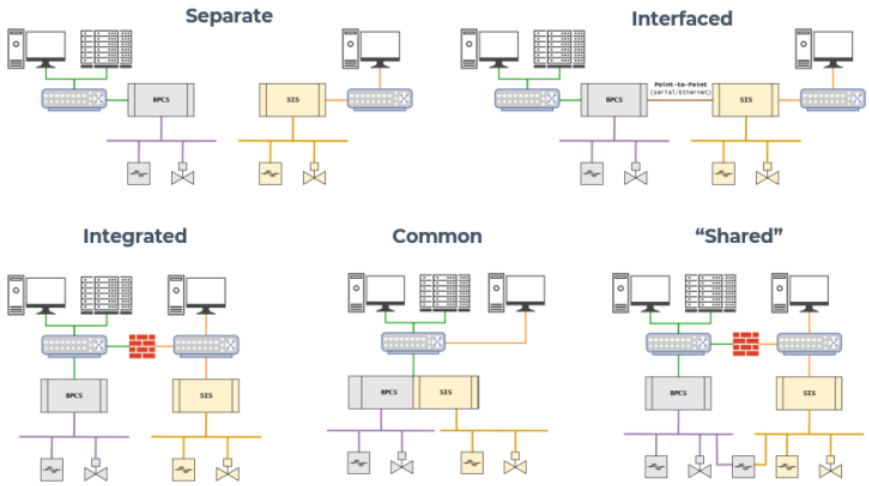

Figure 2.5: Basic Process Control Systems - SIS architecture

The researchers clarified, ”A hacked device’s effects go beyond anything its link or first-order connectivity can do right now”. An attacker may still be able to use a connection that seems unimportant, even if the device only exposes a small number of Modbus registers or is linked to it. They may utilize the unimportant-looking tool as a springboard to more important objectives. Through this strategy, attackers can obtain deep access and perhaps carry out operations that greatly increase the impact of an assault [12].

2.5 Industrial Control System

The term ”industrial control systems” (ICS) is broad and includes a variety of control system types and related instruments. The equipment, networks, and controls needed to run and automate industrial operations are all part of these systems. Depending on the business, an ICS's specific features and layout may change, however, all ICS systems strive to effectively handle duties using electronic means. These days, almost all industrial sectors and vital infrastructure, including manufacturing, transportation, electricity, and water treatment, rely heavily on ICS devices and protocols. Large, networked systems and compact models with a few modular controllers arranged on a single panel are examples of ICS [15].

Field Devices in an ICS are often in charge of managing local operations. They get supervisory directives from distant stations. Industrial Control System Types:

- **SCADA**

- **Distributed Control System (DCS):** A single location's production processes are managed by this system. Setpoints are transmitted to controllers in a DCS, which subsequently give commands to actuators or valves to maintain the intended conditions. Field data can be archived for later use, applied to basic process control, or incorporated into more sophisticated control schemes for the entire plant. Quick access to production and operating data is made possible by DCS, which manages numerous local controllers or devices via a centralized supervisory control loop. A DCS lessens the effect of a single problem on the entire system by sharing control among several units.

A hybrid method is frequently used to implement an ICS environment, incorporating features from both DCS and SCADA systems to maximize their respective advantages.

Evolution of ICS and fusion between OT and IT

Critical infrastructures including power plants, water and wastewater systems, and the transport sector used to function in a somewhat isolated manner, with many of them being ”silent” or cut off from outside networks. Even the computerized ones were usually restricted to internal networks and kept apart from the outer world.

Nevertheless, currently, the majority of these vital infrastructure parts, mechanical or engineering, are either directly or indirectly linked to the Internet. Industrial Control Systems (ICS) now seamlessly integrate Information Technology (IT) and Operational Technology (OT) thanks to technical advancements.

Companies can benefit from improved integration and visibility throughout their supply chains, which include crucial assets, logistics, planning, and operational procedures, thanks to the IT and OT convergence. By streamlining procedures, increased visibility helps businesses maintain their competitiveness. But this convergence also brings with it new risks. Cybercriminals can more easily access points when IT and OT are integrated, and many organizations discover that their OT infrastructure is not sufficiently secured against cyberattacks [16, 15].

Components of an Industrial Control System (ICS) Environment

Programmable Logic Controller (PLC)

This kind of hardware serves as a crucial control element for the entire system in Supervisory Control and Data Acquisition (SCADA) and Distributed Control Systems (DCS) systems. With the use of feedback control devices like actuators and sensors, it oversees local operations.

A Programmable Logic Controller (PLC) in SCADA systems provides local control and data collecting in a manner akin to that of Remote Terminal Units (RTUs). PLCs act as local controllers in a supervisory control system in DCS. Furthermore, PLCs are frequently the main elements of smaller control system designs [17].

Remote Terminal Unit (RTU)

In industrial control systems (ICS), a remote terminal unit (RTU) is a microprocessor-based device that connects hardware to supervisory control and data acquisition (SCADA) or distributed control systems (DCS) systems. RTUs essentially act as a conduit between SCADA or DCS systems and physical processes.

Often called remote telemetry units (RTUs) or remote telecontrol units, these devices are usually put in remote areas to collect telemetry data and remotely monitor and operate field devices including robotic arms, actuators, sensors, and valves. They gather, process, and send sensor data to a centralized control system from input streams. RTUs automatically

manage connections to local or distant controls to provide smooth, continuous monitoring.

RTUs gather data from field sensors, both analog and digital, compile it, and then send it to a central SCADA system. This enables one to keep an eye on the state of the equipment, position, temperature, fluid levels, and security.

An RTU's hardware consists of integrated diagnostic features, communication protocols, and the configuration software required to link data output streams. RTUs frequently have battery backups and are typically powered by alternating current (AC) with direct current (DC) converters. They are made to function in challenging conditions, like those involving high temperatures, exposure to chemicals, intense vibrations, dust, or high pressure. RTUs are more dependable under these circumstances than PLCs, which could malfunction more quickly.

Although they are both a part of the industrial Internet of Things (IIoT) and are utilized in industrial settings, RTUs and PLCs have different functions. Usually, PLCs are utilized for local automation and control of machinery such as motors, pumps, and valves. They are programmed in languages like Ladder Logic or Structured Text and require fast wired connections. PLCs frequently have integrated screens that allow human operators to view and manage equipment.

RTUs, on the other hand, depend on wireless connections and are made for remote monitoring across wide regions. Operators can preprogramme or configure them using a straightforward web interface; no specialized software or programming knowledge is required. RTUs are more resilient than PLCs; they can function in harsh environments and frequently have backup power sources, such as solar panels or batteries, to guarantee ongoing functioning.

Compared to PLCs, RTUs are more intricate and adaptable, operating as standalone computers with CPUs, memory, and storage. Because of their ability to interact with both DCS and SCADA systems, they are appropriate for applications that need dependable and durable remote monitoring and control. However, because of their superior durability and advanced capabilities, RTUs are typically more expensive than PLCs.

It's crucial to take the application and surroundings into account while deciding between an RTU and a PLC. PLCs are more suited for smaller, localized control duties in milder circumstances, whereas RTUs are good for monitoring equipment over large geographic areas under tough situations.

Control Loop

In an industrial system, PLCs, sensors, and actuators are essential hardware elements that are involved in each control loop. Process variables (PVs) must be kept at a desired set point (SP) by these control loops in order to guarantee system stability and produce consistent process results.

A control loop detects mistakes and applies corrective measures in the same manner as a feedback system. This is accomplished by connecting different hardware parts in a series that controls one or more variables. These parts must be connected correctly because any error could cause the automated process to stop and force a transition to manual control.

In a control loop, sensors are the first measurement devices that translate process variables into analog or digital signals. The controller reads these signals and compares them to the intended set point. Transducers, sophisticated sensors that modify signals, can change these values even more. For example, they can change current readings into voltage.

Standardizing signals across the control loop through the use of transmitters enables remote monitoring and control of variables like pressure, flow, and temperature. After interpreting the sensor signals and comparing them to the predetermined point, the controller decides what control action is required. An error detector evaluates whether the measured PV matches the SP at the start of the process.

After that, the controller sends out an error signal, which causes a control action to be sent out to fix the process. Usually, electrical or pneumatic signals are used to process these control activities. After receiving these signals, the last control elements, actuators included, modify the process variable to the desired parameter. This system's response time is essential for efficient control, especially when handling SP variations.

Actuators are essential components of the final control elements because they have a direct impact on the control process. By processing control actions and managing the transition from the present PV to the target SP, they carry out the required modifications [18].

Human Machine Interface (HMI)

In an industrial control system (ICS), a Graphical User Interface (GUI) program plays a critical role in enabling human operators to interact with controller hardware. It gives operators the ability to monitor and establish setpoints, control algorithms, and

modify controller parameters in addition to displaying historical data and real-time status information from devices within the ICS environment.

Operators can communicate with automation and control systems through the use of Human-Machine Interfaces (HMIs), which are specialized computer-based solutions. HMIs, which act as a link between the operator and the automation system, usually come with a touchscreen or graphical user interface (GUI) that shows data in real-time, including alerts, system status, and sensor readings. HMIs offer vital information about the state of processes and mechanical performance, whether they are incorporated within machinery, shown on computer monitors, or accessed by tablets.

HMI optimizes industrial processes by digitizing and centralizing data and displaying critical information for operators to examine on graphs, charts, or digital dashboards. Through a single console, they link with SCADA, ERP, and MES systems and manage alarms as well. This connection is improved by the trend towards high-performance HMI design, which concentrates on the most important indicators. Operators can detect and address problems more rapidly because of its simplified design, which also allows them to make better decisions. High-performance HMIs purposefully have straightforward, uncomplicated indicators devoid of extraneous controls or visuals [19].

Remote Diagnostics and Maintenance

Using sensors, software, and other technologies, Remote Monitoring and Diagnostics (RMD) is a maintenance operations and management system that allows for remote equipment and system monitoring and diagnosis. By enabling maintenance staff to identify and resolve problems without physically being there, remote monitoring and maintenance (RMD) systems assist in minimizing downtime, increasing productivity, and saving maintenance expenses. Furthermore, real-time data and analytics are provided by RMD systems, enabling maintenance staff to make deft judgments. Numerous devices and systems, including industrial machinery, HVAC systems, electrical systems, and more, can be equipped with these systems.

Control Server

An essential function of the Control Server is to coordinate activities inside the ICS. To collect data and provide control commands, it interfaces with field devices such as Supervisory Control and Data Acquisition (SCADA) systems, Remote Terminal Units

(RTUs), and Programmable Logic Controllers (PLCs). The server gives operators real-time visibility into the system's state by continuously monitoring process parameters including temperature, pressure, and fluid levels. This helps operators identify and address any anomalies or failures as soon as they occur.

The Control Server records, processes, and stores data gathered from multiple sensors and devices in addition to monitoring. This information is used to create reports and set off alarms based on predetermined criteria. It is stored in databases for future examination. The server creates alarms when monitored data surpasses critical limits. These alarms can be forwarded to other systems for prompt remedial action or shown to operators.

Since ICS frequently oversees vital infrastructure, security safeguards are built into the Control Server to thwart unauthorized access and shield the network from online attacks. Data encryption, firewalls, and authentication are a few examples of these security precautions. The Control Server can also be connected to other enterprise systems, such as Manufacturing Execution Systems (MES) or Enterprise Resource Planning (ERP), to facilitate more comprehensive and coordinated control over industrial operations.

To ensure that operations are safe, effective, and compliant with industry standards, an ICS's Control Server is an essential component that centralizes industrial process management, monitoring, and control.

SCADA Server or Master Terminal Unit (MTU)

The Master Terminal Unit (MTU) is the primary device in a SCADA system that sends commands to Remote Terminal Units (RTUs) that are situated in remote locations. The MTU collects, saves, processes, and presents data as graphs, tables, and curves to give human operators the knowledge they need to make control choices. The primary difference between the RTU and MTU, despite their bidirectional connectivity, is that the RTU is limited to gathering and storing field data; it is unable to start conversations. The MTU is always the one to start communication, whether it's through programs that run automatically or operator commands.

The RTU transmits the requested data when the MTU makes a specific data request. The MTU is the "master" in this connection, and the RTU is the "slave". When the MTU has the relevant data, it uses the appropriate protocols to connect with operator interfaces such as printers and CRTs. Peer-to-peer communication replaces the master-slave model at this point. The MTU is essentially the center of everything that happens in the SCADA

system.

Intelligent Electronic Device (IED)

Intelligent Electronic Devices (IEDs), which offer advanced capabilities for monitoring, control, and protection, are essential components of contemporary Industrial Control Systems (ICS) and power automation. These microprocessor-based electronic parts, which include regulators and circuit controllers, talk to each other via different protocols like Fieldbus and real-time Ethernet.

IEDs are essential to systems like SCADA and Distributed Control Systems (DCS), which provide precise, dependable, and effective administration of electric power systems. They are made with sophisticated algorithms and digital signal processing methods for specialized purposes, like fault detection, protection, and control.

IED sensors monitor voltage, current, power, and frequency, among other factors. IEDs carry out particular tasks based on these measures, providing operators with real-time data that enables them to make well-informed decisions regarding the generation, transmission, and distribution of power. IEDs have completely changed the way we manage, monitor, and control electric power systems by carrying out intricate calculations and interacting with other devices in the system [20].

Data Historian

An Industrial Control System (ICS) environment's data historian is a centralized database created to record all process information and export it to the corporate Information System (IS). The collected data is then used for process analysis, statistical process control, and enterprise-level planning. This system, which is a collection of time-series database systems specially made for gathering and storing data about industrial activities, is often referred to as a process historian, operational historian, or just "historian".

Data historians were first designed for use in industrial settings, but they are now widely used in many other industries as vital tools for analytics, supervisory control, performance monitoring, and quality assurance. They make it possible for engineers, data scientists, industrial facility managers, and machinery operators to access and use data gathered from different automated systems and sensors for business analytics, process tracking, and performance monitoring, among other things.

Additional features, such as reporting capabilities that let users create manual or

automated reports, are frequently included with modern data historians. Although data historians and conventional databases, such as SQL, are comparable, data historians provide more features than just data storage. In addition to gathering raw data, they can also process it, compile it into insightful reports, and send it to further storage facilities. A standard data historian is essentially a highly customized time-series database designed to fulfill the demands of industrial automation [21].

A complete representation of an ICS system is in the Figure 2.6

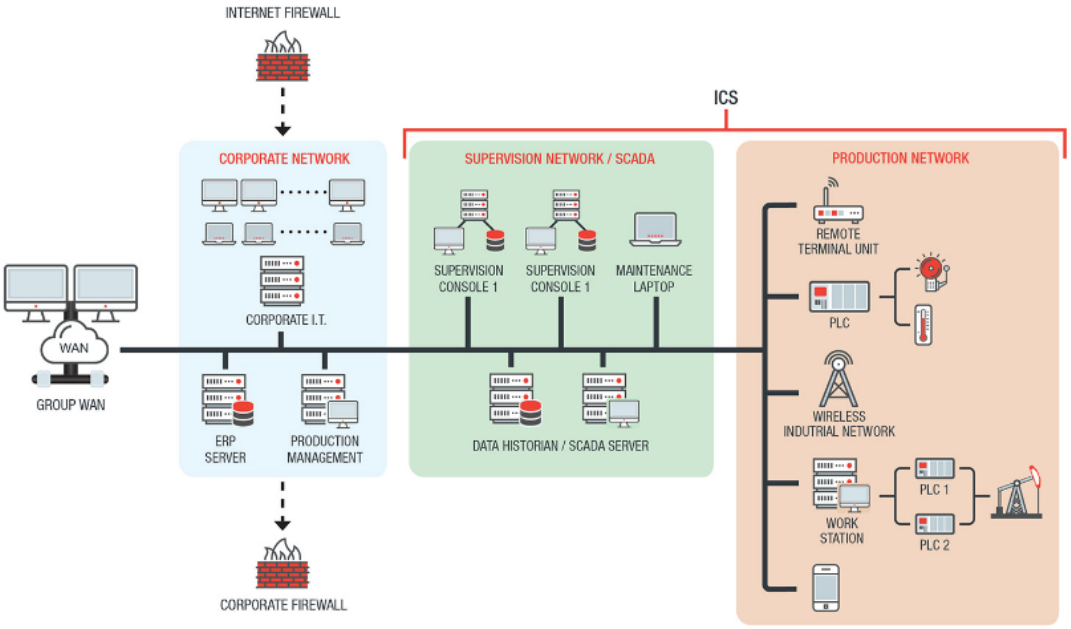

Figure 2.6: ICS System

Communication within ICS Systems

Several communication protocols are used by devices and control modules in ICS systems to exchange information. These protocols are designed with particular applications in mind, like building automation, power systems automation, and process automation. They were created to guarantee equipment from various manufacturers would work together. Certain protocols, however, can only work if the apparatus and the protocols are made by the same company.

Process Field Bus (PROFIBUS)

PROFIBUS supports many communication formats, such as RTU to MTU, MTU to MTU, and RTU to RTU field communications. There are primarily two types:

- **PROFIBUS DP (Decentralized Peripherals):** The fieldbus that is most frequently utilized in industrial settings worldwide is PROFIBUS DP. With speeds of up to 12 Mbps, it provides speedier data transmission, data can be sent in less than a millisecond. It also supports plug-and-play capability, which simplifies installation and allows for seamless integration with extensive automation systems. High-speed data transfer, cheap initial implementation costs, and dependable installations are guaranteed by PROFIBUS DP, especially in situations where power supply and cabling are unimportant.

  PROFIBUS is a great option for a variety of automated environments because of its adaptability. Because PROFIBUS may be used in a variety of contexts and is not restricted to any one application, it eliminates the need for several protocols inside a single facility, unlike other field buses.

- **PROFIBUS PA (Process Automation):** A PROFIBUS DP variation created especially for process automation, PROFIBUS PA is meant to complement rather than replace traditional systems. In contrast to PROFIBUS DP, PROFIBUS PA provides an option for explosion safety and permits the transmission of power and data over the same two wires. PROFIBUS DP and PA are the same in terms of usage and protocol.

  PROFIBUS PA has fewer cabling requirements, better asset management, and the capacity to make modifications without changing the setup.

Distributed Network Protocol (DNP3)

DNP3, or IEEE Std 1815, is a comprehensive protocol that outlines the guidelines for computer-to-computer communication. A master and a remote unit are the two kinds of endpoints that DNP3 defines as being able to communicate with each other [22]. Below is a quick synopsis of each:

- The Master: A control center's master computer is utilized there. This robust computer stores all of the data from distant units so that it can be processed and shown.

- The Remote Unit: The remote unit, often called the slave, is a field computer. These computers gather data from several field devices, including voltage and current sensors, and forward it to the master station. Any gadget that has direct communication with the master, like an RTU, IED, voltage or flow meter for water, a solar inverter, or any other kind of controlled station, can also be considered a DNP3 remote unit.

Additionally, DNP3 specifies data variables according to their kind and behavior, and it assigns a priority to each variable according to whether or not it represents a change from its starting state. During an integrity poll, which asks the remote unit to transmit the values and statuses of configured points to the master, the master sets these parameters. Following this configuration procedure, the remote unit broadcasts events in a selected manner according to whether the data has changed since the last poll. These transmissions usually happen in cycles, but they can sometimes happen on their own when specific circumstances are met.

The need for data protection along the entire transmission line grows when DNP3 is used in a system. Unauthorized access must also be prevented to the system. To this purpose, TLS encryption and secure authentication techniques are frequently used in tandem by DNP3-based applications:

- TLS Encryption: Through data encryption that only the inside system can decrypt, TLS encryption safeguards linked systems over TCP/IP channels. TLS encryption is a widely used security mechanism against information exposure, unauthorized access, and message modification. It is well-defined by the DNP3 standard and the associated IEC 62351 Part 3 standard.

- Secure Authentication: A master or remote unit may request authentication to use this optional approach. Setting output commands or reading acknowledgment messages are two examples of crucial functions that are frequently behind authentication and have an impact on the functionality of the system. Bidirectional authentication is based on the challenge-response model.

Modbus

One of the earliest ICS protocols, Modbus was first introduced in 1979 and has long been a standard for serial communication with PLCs. In ICS environments, Modbus has emerged as the standard communication protocol. Modbus Servers are the devices that provide the

information, and Modbus Clients are the devices that request the information. Modbus is currently the most extensively used protocol for linking industrial electrical devices and has established the industry standard for communication. Signals from instrumentation and control devices are typically sent over Modbus to a primary controller or data-gathering system. It can be utilized, for instance, in a system that sends data to a computer about temperature and humidity measurements. In supervisory control and data acquisition (SCADA) systems, Modbus is frequently used to link a supervisory computer and a remote terminal unit (RTU) [23]. There are two main ways that Modbus is implemented:

- **Serial Modbus:** Using a sequence of data bits, the Modbus serial communication structure was first designed to facilitate communication over RS-232 and RS-485 serial networks. Positive and negative voltages are used to represent these bits on the serial port's Rx (receive) and Tx (transmit) terminals. The master-slave model underlies this device-to-device interaction, which involves two distinct transmission types: RTU and ASCII.

- **Modbus-TCP:** Modbus may now communicate over Ethernet networks thanks to a variation of the original protocol called Modbus TCP, which runs over TCP/IP networks. Modbus networks with a variety of device types can be created heterogeneously thanks to the TCP transport protocol. Because more sophisticated networks are more versatile, multiple segments can coexist on the same network.

  Because Modbus TCP works with contemporary Ethernet networks, it is becoming more and more popular. The only prerequisite for it to operate in a client/server communication architecture is for all networked nodes to be within the same IP address range. But network communication latency and other related issues have made changes necessary to guarantee that queries and answers stay in sync and that a slave device doesn't receive inaccurate data [24].

Open Platform Communication (OPC)

The industry standard for safe and dependable data interchange in industrial automation and other sectors is OPC. It guarantees a seamless data transmission across devices from various suppliers and is platform-independent. The OPC Foundation is in charge of this standard's creation and upkeep.

A number of specifications created by software developers, industry providers, and end users make up the OPC standard. The interfaces between servers and clients as well as

between servers themselves are defined by these specifications. They encompass a range of tasks, such as access to historical data, monitoring of alarms and events, and real-time data access, among other applications.

Data modeling and security faced new difficulties as manufacturing systems adopted service-oriented architectures. The OPC Foundation created the OPC UA (Unified Architecture) specifications to overcome these issues. OPC UA provides an open-platform architecture with a wealth of features that are scalable, extendable, and future-proof, well suited to the changing requirements of contemporary industrial systems [25].

Building Automation and Control Networks (BACnet)

Building automation systems (BAS) use the network protocol BACnet to exchange data more easily between different components and devices. Ensuring interoperability across the many systems and devices within building automation is one of its main benefits. In order to accomplish this, BACnet offers a standardized technique for controlling every activity and feature in a network communication system.

Through the representation of information as objects, where each object corresponds to data or information about a component or device, the BACnet standard facilitates interoperability. In addition to enabling the production of new items or the alteration of existing ones to suit particular user demands, this makes the information available over a network. Furthermore, BACnet harmonizes device connectivity, facilitating efficient information sharing independent of the manufacturer. Additionally, it supports several network protocols, such as IP, which enables BACnet-compatible devices to link to and function within an IP network, also referred to as BACnet/IP.

Because of these characteristics, BACnet is a reliable and adaptable protocol for building automation systems, enabling easy integration and communication across various gadgets and technological platforms [26].

Common Industrial Protocol (CIP)

The producer-consumer communication model is used by the media-independent CIP (Common Industrial Protocol), which is purely object-oriented at its upper tiers. The attributes (data), services (commands), connections, and behaviors (the connection between attribute values and services) that characterize each CIP object. To facilitate general-purpose network communications, network services like file transfer, and common

automation features including analog and digital input/output devices, HMI, motion control, and position feedback, CIP has a large object library.

The same object or set of objects implemented in several devices must act consistently across those devices to guarantee interoperability. The "Object Model" of a device—a term that describes the particular arrangement of objects employed by that device—is what makes it consistent. The producer-consumer communication paradigm, which maximizes the utilization of network resources, is the foundation of the CIP Object paradigm. With this paradigm, there is no requirement for the data to be sent from one source to several destinations; instead, a sending device (the producer) can exchange application information with multiple receiving devices (the consumers) at the same time.

With a common application layer and a single, media-independent platform, CIP enables smooth communication between the plant floor and the organization. It is a flexible solution for industrial automation because of its scalable and coherent architecture, which makes it possible to integrate device configuration, I/O control, and data gathering across numerous networks [27].

Ethernet for Control Automation Technology (EtherCAT)

High-performance Ethernet Fieldbus technology known as EtherCAT® (Ethernet for Control Automation Technology) is intended for dependable, effective, and economical communication in a variety of industrial automation applications. EtherCAT uses a master/slave design for communication between devices on the same network, based on the IEEE 802.3 Ethernet standard. The master device acts as the controller in this configuration, giving commands to the slave devices and receiving data from them. Additionally, slave-to-slave and master-to-master communication are supported by EtherCAT.

The technique works with a wide range of devices, including actuators, motors, distributed I/O, sensors, and other controllers. EtherCAT's high-speed, low-latency connection is one of its main advantages. Its frame processing and data transfer techniques are distinct from previous Fieldbus systems, allowing for outstanding bandwidth utilization. Because EtherCAT's topology design guarantees deterministic cycle times, it can be used in situations where strict real-time requirements are necessary. Furthermore, EtherCAT facilitates the use of Distributed Clocks (DC), which offer accurate nanosecond-level network synchronization.

EtherCAT is perfect for high-speed and safety-critical applications including machine

tools, medical robots, testing and measurement, and factory automation because of its versatility, performance, and dependability [28].

2.5.1 Security of Industrial Control Systems

Industrial system protection is a difficult task. Given that many of these systems were developed before the increased pervasiveness of cyber threats, they lack internal and external security safeguards. The first stage in every organization's security process for industrial control systems is to recognize and comprehend prevalent hazards [15]. There are a few major threats to be aware of today:

- **External Threats and Targeted Attacks:** Since ICS frequently deal with vital industries including manufacturing, distribution, chemical engineering, and healthcare, they are frequently the focus of attacks by hacktivists, terrorist organizations, and other bad actors. Other common threats include intellectual property theft and industrial espionage.

- **Internal Threats:** A major vulnerability of many Industrial Control Systems to internal threats is the absence of strong authentication and encryption. Using basic tools like USB devices, irate workers or disgruntled contractors might take advantage of their access to steal confidential information or implant harmful viruses.

- **Human Errors:** While mistakes are unavoidable, they can have particularly large and expensive effects when they occur in a network of industrial control systems. The biggest danger to industrial control systems (ICS) networks is frequently thought to be human error, which includes configuration problems, PLC programming errors, and neglect to monitor vital metrics or alerts.

What Draws Attackers to Target Industrial Control Systems

When selecting a company to target, threat actors have varied reasons for doing so. These threat actors frequently have financial gain, political causes, or even military objectives as their driving forces when they launch strikes. Attacks may originate from competitors, state-sponsored entities, insiders with nefarious intentions, or even hacktivists [29].

How ICS are attacked

Reconnaissance is the first stage of an attack on an Industrial Control System (ICS), during which the attacker looks around to acquire information. The subsequent phase entails utilizing diverse strategies to establish a presence within the intended network. By now, the techniques and strategies are not unlike those employed in targeted attacks. Malware will be deployed by attackers using ICS setups and vulnerabilities. The assault may result in modifications to current controls and configurations, processes, or functions when these vulnerabilities are found and exploited.

Several variables, such as the intended damage and the security measures in place, affect how difficult it is to conduct an ICS attack. For instance, it is typically simpler to carry out a denial-of-service assault that interferes with the ICS than it is to manipulate a service and conceal its immediate consequences from controllers. Even though there are many ways to harm an ICS, as more devices are incorporated into ICS environments, new strategies are probably going to surface.

Vulnerabilities exploited in ICS

Since both operational technology (OT) and information technology (IT) are included in Industrial Control Systems (ICS), classifying vulnerabilities aids in the identification and application of efficient mitigation techniques. A security guide for industrial control systems (ICS) is available from the National Institute of Standards and Technology (NIST), which divides vulnerabilities into multiple categories. These include problems with policies and procedures, as well as vulnerabilities relating to different platforms like networks, hardware, operating systems, and ICS applications:

- Policy and Procedure Vulnerabilities
- Platform Configuration Vulnerabilities
- Platform Hardware Vulnerabilities
- Platform Software Vulnerabilities
- Malware Protection Vulnerabilities
- Network Configuration Vulnerabilities
- Network Hardware Vulnerabilities

- Network Perimeter Vulnerabilities
- Communication Vulnerabilities
- Wireless Connection Vulnerabilities
- Network Monitoring and Logging Vulnerabilities

Depending on how it is set up and used, every ICS environment may have flaws. An ICS environment's size can also play a role; the bigger the environment, the higher the chance of errors happening. Furthermore, there may be more weaknesses in an ICS environment where threat actors could take advantage of them because of the introduction of technologies like Industrial Internet of Things (IIoT) devices and the replacement of outdated systems with more current ones.

Industrial IoT and How It Affects ICS

When merging Human Internet of Things (HIoT) and Industrial Internet of Things (IIoT) devices, there may be a spike in platform-specific vulnerabilities and malware, much like how the advent of smartphones was followed by a comparable surge in problems. Because every IoT device needs to be properly secured and safeguarded, managing them in an ICS context poses serious security problems. Should sufficient security measures not be implemented, the entire ecosystem of the ICS becomes extremely susceptible to attacks. The application of IIoT brings with it several particular difficulties that must be resolved:

1. **Technology Fragmentation:** Network operations are made more difficult by the variety of devices running separate or distinct operating systems. Keeping track of different patch schedules might be especially difficult. For instance, communication problems between legacy and modern software may arise in an ICS that combines both. Furthermore, threat actors may be able to access the ICS network by taking advantage of weaknesses in unpatched legacy systems.
2. **Complexity of M2M and IoT Application Development:** The process of developing Machine to Machine (M2M) and IoT applications for ICS is more complex than that of mass-produced HIoT devices since it calls for specific expertise in IT, communications, hardware, and software development, and IT.
3. **Prevalence of Legacy Systems and Communication Protocols:** Numerous industrial settings continue to rely on antiquated technologies and procedures. For

example, the DECOR program, which is necessary for airplane takeoff and landing, still runs on Windows 3.1. Furthermore, there is still widespread use for outdated communication protocols like PROFIBUS. To enable data and command exchange, these systems must be interconnected via standards-based protocol gateways.

Possible Effects of Cyberattacks on ICS Components

The way that a target operates and the intentions of the cybercriminals engaged determine how a cyberattack affects industries that use ICS. The following effects could affect the targeted entity's internal and external customers:

1. **Alterations to Systems or Application Configurations:** Modified systems, operating systems, or application configurations can have unanticipated or undesirable effects. This might be used to hide dangerous activity or virus behavior, and it might also have an impact on the targeted system's output.

2. **Modifications to Programmable Logic Controllers (PLC), Remote Terminal Units (RTU), and Other Controllers:** Changes to controller modules and other devices might result in harmed facilities or equipment, much like system modifications. These alterations may also result in process breakdowns and impair control of vital functions.

3. **Misinformation Reported to Operations:** Actions that are harmful or unneeded may be taken as a result of inaccurate information. This situation has the potential to alter programmable logic and to hide malicious activity, such as code injection or the occurrence itself.

4. **Compromised Safety Controls:** When safety controls and other safeguards are tampered with, they may stop working as intended, seriously endangering the lives of staff members and sometimes even outside clients.

2.5.2 Defensive Strategies for Industrial Control Systems

Securing Industrial Sectors

Strategies for preventing cyberattacks and data breaches ought to be a crucial component of a company's everyday operations [30]. The fundamental idea of a good defense is to

anticipate compromise and take countermeasures, even though no defense is impenetrable to determined opponents completely:

- Rapidly identify and respond to security breaches in progress.
- Contain breaches to minimize the loss of sensitive data.
- Analyze incidents and apply lessons learned to strengthen defenses and prevent recurrence.
- Proactively secure all potential vulnerabilities to prevent future attacks.

Network Segmentation

In the manufacturing sector, the Purdue Model for Control Hierarchy, in Figure 2.7, is a well-known framework for classifying machinery and devices into hierarchical functions. The following levels and zones within this logical framework have been recognized by the Committee for Manufacturing and Control Systems Security of the International Society for Automation (ISA-99) [30]:

Six levels of activities and five zones are identified by the framework:

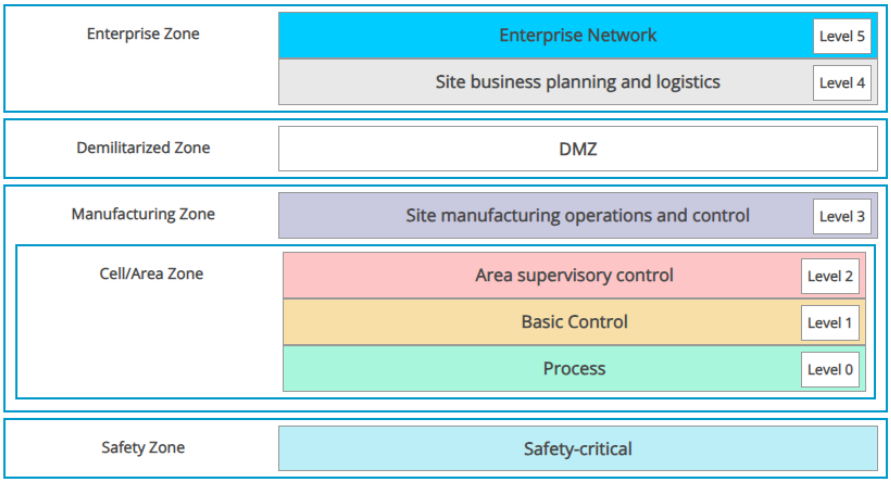

Figure 2.7: The Purdue Model for Control Hierarchy

Safety Zone

- *Safety-Critical:* This comprises the tools, sensors, and other machinery that an Industrial Automation and Control System (IACS) uses to control its safety features.

Manufacturing Zone (4 Levels)

- *Cell/Area Zone*
  - *Process (Level 0):* This is the area where machines like drives, motors, and robots, as well as devices like sensors and actuators, interact with one or more controllers.
  - *Basic Control (Level 1):* The IACS can be operated in part or its entirety by interacting with multidiscipline controllers, dedicated HMIs, and other applications.
  - *Area Supervisory Control (Level 2):* This covers the control room, IACS network/application administration, controller status monitoring, and other control-related programs like historians and supervisory control.
- *Manufacturing Zone*
  - *Site Manufacturing Operations and Control (Level 3):* Detailed production scheduling, reliability assurance, site-wide control optimization, security management, network management, and possibly other essential IT services like DHCP, LDAP, DNS, and file servers are among the tasks associated with overseeing workflow for end-product production that are included in this level.

Demilitarized Zone (DMZ)

- *DMZ:* Between the Manufacturing and Enterprise zones, this acts as a buffer zone where services and data can be transferred. It makes organizational control easy to segment and ought to be set up so that no traffic passes through the DMZ, all traffic should start or end there.

Enterprise Zone (2 Levels)

- *Site Business Planning and Logistics (Level 4):* This includes local IT services like phone, email, printing, security/monitoring, and scheduling systems as well as manufacturing facility IT services including material flow applications, scheduling systems, and manufacturing execution systems (MES).
- *Enterprise Network (Level 5):* Corporate-level programs like document management, ERP, and CRM are included in this level, along with services like VPN entry points and internet access.

Security Strategies

As observed in traditional office-based settings, attacks on ICS environments might result in financial losses or business disruptions, but they can also have more serious repercussions, including damage, death, or even catastrophic events, especially in public service systems. In order to identify the different types and levels of risk associated with ICS systems and to put the necessary protections in place, security teams must conduct a thorough assessment of the systems.

Collaborative Network Environments

For the IT team, collaborative network environments present special difficulties. For the IT team to do risk assessment and decide the appropriate design of IT solutions, they must be involved from the outset of planning and development. IT may only be able to offer ad hoc, tactical solutions if they do not completely comprehend the conditions and specifications of the collaboration agreement. IT solutions might not fulfill the necessary compliance standards if IT is not involved in the planning and development stages. Granting access to digital assets inadvertently raises the possibility of security lapses that may breach third-party contracts [30]. Considerations for new IT partnerships include:

- Insider threat complacency/ignorance/malice
- No operating agreement terms for digital assets
- No standardized operating agreements with partners
- Application licensing agreements
- Export compliance laws
- Risks of intellectual property leakage
- Privacy regulations
- Alterations to the terms of operation over time, etc.

To prevent security breaches caused by unauthorized parties having access to the corporate network, IT must carefully set up digital borders. Various partners will require various access privileges to project data, corporate data, applications, etc. The necessary departments, including IT and legal, should assess requests from third parties. Strict

execution of the IT solutions, appropriate documentation, and routinely planned compliance checks and revalidation, which will be predicated on risk assessment, are all necessary. Among the factors to examine in a risk assessment are:

- Partner reputation
- International or domestic partnerships
- Dangers to cyber security in the operating nation
- Corruption in the country of operations
- Joint operations risk scenarios
- Type of legal joint venture entity: to accommodate various joint venture working environments and the risks that go along with them, IT should have pre-defined operation models.

Security best practices include:

- Identifying intellectual property and safeguarding them
- Restricting access to intellectual property to those who need to know, and
- Training employees to protect intellectual property

Strategies for securing the corporate network include:

- Putting in place Network Access Control (NAC) to provide a safe facade. This makes it possible for users and devices to authenticate before being granted access to the company network.
- Putting identity awareness into practice entails creating, logging, and maintaining user and device identities as well as the access control policies that go along with them. Every kind of network user and device is defined and managed by the stored identity.
- Employ identity-aware firewalls that allow the management of the servers and network by access regulations set up for every user or device that connects.
- Integrating access control and identity awareness into a final network architecture solution that can enforce access regulations across wired, wireless, and VPN networks, independent of how and where users connect, will strengthen policy enforcement.

2.6 Knowledge representation and reasoning

Big data is still a widely used technology in cybersecurity since it is thought that combining ecosystem telemetry and network data into a big data engine will improve the capacity to spot hostile activity [31, 32]. This theory has two serious shortcomings:

1. **Dependence on Data Quality:** The caliber of the data that big data analytics tools obtain from sources is crucial to their functionality. The analysis will be useless if there are key indicators of malicious activity missing from the data.

2. **Lack of Context:** Without the right context, analysis is unable to demonstrate the significance of threats and is ineffective in aiding in defense, detection, and repair.

Conventional data sources like Syslog and NetFlow frequently fail to identify important warning signs of malicious activity, instead presenting conventional data analytics engines with an activity that appears to be uncharacterized ambient traffic. This data is used by Security Information and Event Management (SIEM) systems, which use traffic and volume comparisons with pre-established thresholds to find signs of breach. Unfortunately, because malicious actors have learned to get around these thresholds and frequently make use of default settings that are disclosed in user guides and are typically kept untouched, these systems regularly overlook vital behavior.

Traditional logs, flows, and baselines miss a lot of telling data components when insiders operate more covertly and malware changes. Coordinated attacks of today also have multiple stages and vectors. Conventional big data analytics frequently examine isolated or combined events without considering their context, thus it misses minute patterns and linked behavior sequences that hackers employ to enhance their Attack-in-Depth tactics.

The current generation of the cyber death chain paradigm, known as Attack-in-Depth, entails delivering payloads, propagating across networks, keeping persistence on endpoints, and eventually exfiltrating or deleting information assets.

It is imperative to move towards contextual data analytics to properly counter these Attack-in-Depth threats. A powerful analytic engine needs to be able to identify subtle signs of malicious activity, which can only be found through sophisticated and context-aware analytics.

Analytics Engines

Analytic engines must employ algorithms built to function within the confines of a particular danger envelope to detect harmful behaviors, both structured and unstructured. Telemetry data from a wide range of the external danger environment, along with behavioral patterns seen both inside and outside the network, should define this threat envelope. To stop asset breaches and detect and isolate threats as soon as a network is affected, these engines must also be able to operate in real time. To make this capability possible, contextual analytics is essential.

An analytics engine is a potent tool that gives businesses a strong platform for processing and analyzing data. Large volumes of data from many sources can be effectively stored, managed, and analyzed by businesses utilizing sophisticated algorithms and powerful computers to carry out intricate data transformations and produce insightful findings. It performs as a cross between big data processing technologies and conventional data warehouses. Provides a unified, scalable platform for tasks related to analytics, machine learning, and data processing. Using parallel processing and distributed computing, the engine effectively manages massive amounts of data. It has capabilities for ingesting, transforming, and analyzing data in addition to supporting a large variety of data formats.

These engines are essential to knowledge representation and reasoning because they make it possible to extract relevant patterns and correlations from massive datasets, which aids in well-informed decision-making. Real-time data processing and analysis skills are especially important in cybersecurity, as rapid threat identification is essential to prevent breaches and preserve the integrity of the system [32].

2.6.1 Reasoning Models

Fundamentally, analytics engines usually adhere to one of four main methods of reasoning:

Deductive Reasoning

The principle of deductive inference, which derives particular conclusions from general norms, is the foundation of deductive thinking. For instance, regardless of what is in A or B, if $A = B$ and $B = C$, then $A = C$. Deductive reasoning follows a path from a general principle to a particular outcome. The conclusion must be true if the initial claims are accurate. Deductive reasoning has two fundamental weaknesses: it is frequently

tautological (e.g., malware contains malicious code and is always true) and unaffected by contextual inputs.

Using deductive reasoning as the foundation for detection analytics is a faulty approach to trying to forecast the future since, in security analytics, A only equals B most of the time and occasionally it can equal D. Therefore, A cannot always equal C. A hypothetical assurance that it will be violated at least once.

Common signature-based solutions, including endpoint security and IDS/IPS, are often deductive in nature.

Inductive Reasoning

The opposite of deductive reasoning is inductive reasoning. With inductive reasoning, specific data are used to draw broad generalizations. We proceed from the specific to the universal inductive inference. We gather a lot of data, look for trends, extrapolate, and develop hypotheses or explanations.

The analytics produced by inductive reasoning-based analytics engines are similar to probability theory. Inductive reasoning permits a false conclusion even in cases where all of the premises are true. As an illustration, consider this: "Grandpa Harold is in charge. Since Harold is bald, all grandfathers must also be bald. The evidence does not logically support the conclusion.

For predicting the future, inductive reasoning is superior to deductive reasoning, but it is still prone to error and can provide much more widely disparate outcomes. Heuristics are used by advanced IDS/IPS systems to detect malicious activity. A heuristic is a rule that offers an expedient way to solve challenging situations. When deciding with little time or knowledge, heuristics are employed. Heuristics typically help you make wise decisions.

Heuristics are often used in computer science to extrapolate, from limited data (e.g., known signatures), the likelihood of dangerous behaviors.

Security and SOC analysts spend all day chasing down rabbit holes since there isn't enough proof to show that a positive is a positive.

Bayesian Reasoning

Recursive Bayesian Estimation (RBE), often known as Bayesian reasoning in security systems, this anomaly-oriented analytical method is used to give a less tactical picture of events over a longer period (e.g., 30 days). "Standard deviation" is a statistical term

used to express how much a group of data values vary or are dispersed. A standard deviation close to zero indicates that the data points frequently tend to be very close to the specified mean value, whereas a high standard deviation indicates that the data points are scattered throughout a wider range of values. When a result deviates from the norm by three standard deviations, the majority of Bayesian-based security analytics systems classify it as an ”anomaly”. Finding a "normal" pattern of behavior by analyzing minute variations in activity inside the corporate infrastructure is the aim of Bayesian Reasoning. The outcome is a baseline that will be used as a future ”benchmark” to gauge all network activity and/or behaviors. This baselining is frequently incorrect and can have unexpected consequences, none of which will lead to threats being correctly recognized. There are three major issues with this strategy:

1. The baseline creates a false premise if the network and/or the systems being established are already compromised before the baseline is generated.

2. The behaviors of an insider who is already engaged on a network will be seen as nominal and will become part of the ”normal” baseline.

3. A baseline evaluation is practically impossible without a network lock-down due to the dynamic, varied, and diversified nature of today’s network infrastructure and user behavior, which involves a wide range of devices, protocols, access methods, and entry points.

The dynamic, varied, and diversified nature of today’s network infrastructure and user behavior—involving a wide range of devices, protocols, access methods, and entry points—makes a baseline evaluation all but impossible without a network lockdown.

Abductive Reasoning

A type of logical inference known as abductive reasoning proceeds from an observation to a hypothesis that explains the observed, ideally attempting to identify the most straightforward and plausible explanation.

Abductive reasoning starts with evidence and progresses towards a hypothesis, to put it simply. Both deductive and inductive reasoning begin with a hypothesis and look for supporting data. Bayesian analysis starts with an extrapolation of potentially manipulated data in order to reach a conclusion.

Unlike deductive or inductive reasoning, abductive reasoning does not rely solely on the premises to conclude. Abductive reasoning typically begins with a partial set of observations and proceeds to the most likely explanation for the set. Abductive reasoning produces the type of day-to-day decision-making that makes the best use of available knowledge, which is frequently insufficient, but probably produces the most optimal model from which an automated outcome can be constructed.

2.6.2 Threat Defense

Incomplete, decontextualized, and dumb data is the foundation for any cybersecurity system's ineffective defense against attackers. Our adversaries are already one step ahead of us, as seen by the hacks at SolarWinds, Accellion, Colonial Pipeline, JBS, and Microsoft.

To narrow this gap, we need a coordinated national effort involving the public and private sectors, backed by updated legislation that gives us the ability to successfully combat these dangers. The development of technology is equally important; it's not only about what will happen when malicious actors become proficient in AI and machine learning. When we, the defenders, completely integrate these technologies into our automated detection and reaction systems is the actual question.

Our cybersecurity architecture may be considerably improved by incorporating AI and ML to improve our capacity to quickly identify, address, and neutralize threats. In addition to technology innovation, this calls for ongoing training, exchange of threat intelligence across industries, and strategy adaptation to keep up with changing threats.

To put it simply, our capacity to safeguard our data and systems depends on our willingness to change more quickly than those who would try to take advantage of our weaknesses. We can't afford to lose this race [14, 32].

2.7 Attack Representation

Software vulnerabilities are flaws in source code that can be used to compromise security and cause serious problems like extortion and user data leaks. The software industry and cybersecurity are experiencing serious worries due to the continuous rise in the number of vulnerabilities that have been reported. Conventional vulnerability detection techniques mostly rely on human specialists' predetermined rules for code analysis, which can be time-consuming and inaccurate.

Deep learning (DL)-based approaches have been the cutting edge of vulnerability detection methodologies recently. By eliminating the need for human heuristics, these techniques automatically identify patterns in vulnerable code, increasing the accuracy of detection. DL-based methods typically make use of the source code's structural information and display the code as a code structure graph, such as a Code Property Graph (CPG), Data Flow Graph (DFG), or Control Flow Graph (CFG). To provide a more thorough knowledge of the code, these graph representations can also be enhanced by merging them with an Abstract Syntax Tree (AST) and Natural Code Sequence (NCS).

Effective learning is hampered by the complicated hierarchical information that code structure graphs often encapsulate. To overcome this, the most recent DL-based techniques use a variety of Graph Neural Network (GNN) models, including the Graph Convolution Network (GCN) and the Gated Graph Neural Network (GGNN), to acquire strong representations of code structure graphs.

Attack graphs are useful tools in network security that help with scalable security analysis by showing possible multi-stage attack vectors. Network administrators can determine effective countermeasures using these graphs, which help them understand the origins and implications of security threats. The process of assessing security using attack graphs usually consists of the following steps: first, gathering network attributes like topology, services, vulnerabilities, and configurations; second, using a formal language to map each attack action to a different vulnerability; third, creating attack graphs to show the threat posed by different attack scenarios to critical assets; and, lastly, suggesting damage mitigation measures.

However, the absence of a standard vocabulary amongst domain experts and network managers for exchanging security knowledge is a significant drawback of attack graphs. Administrators may find it difficult to model security issues and make wise judgments due to the volume of data collected from various devices on company networks, particularly if they lack experience in security assessment.

Chapter 3

Model Definition

This chapter will analyze the developed model that aims to produce a ranking that classifies the vulnerabilities present in an industrial network based on their level of danger and risk, starting from certain input parameters, which will be explained in detail later. This model is particularly relevant and has been adopted because it allows to have an organized view of the main weaknesses to which our network may be exposed, according to their risk level. This provides a comprehensive overview of vulnerabilities, from the most critical to the least severe, that require urgent security measures.

The model in question is not based on any previous models or theoretical frameworks found in the literature, as this is a cutting-edge field. For this reason, an in-depth study was conducted, starting with the various inputs necessary for the proper functioning of the final tool, leading up to the mathematical model chosen to process the data and validate the obtained results.

3.1 Input

During the preliminary phase of defining the requirements and parameters, the following inputs were identified:

- A comprehensive description of the network under examination.
- A description of the various attack paths.
- A list of existing CVEs along with their scores.

- A list of existing CWEs with their detailed descriptions and associated references linked to certain CVEs.
- A list of misconfigurations, i.e., non-standardized issues that do not have an official score.
- The key element(s) within the network that should be assigned a maximum asset value (this may require human assistance or could be automated during the network analysis process).

Subsequently, we will analyze each point to better describe how and why these input data were chosen.

3.1.1 Network description

The network description is valuable as it provides an overview of the existing subnets along with detailed descriptions of all the hosts within each subnet. It also includes information about data flows, specifying the source and destination hosts, the protocols used, and the associated ports. However, the most relevant section for our purpose is the one dedicated to vulnerabilities. Each vulnerability is described by a set of common parameters, with additional ones that vary depending on the type of vulnerability. Generally, the following fields are included:

- **id:** it is a unique identifier for each vulnerability.
- **cve:** it represents the reference code of the vulnerability. This code may be present for known vulnerabilities and serves as a globally unique identifier used to track known vulnerabilities in software and systems. Alternatively, it may be left blank in cases where a misconfiguration is present that cannot be linked to a specific vulnerability.
- **type:** indicates the type of vulnerability. It may refer to a service, a layer 2 (L2) network protocol, an end-to-end (E2E) protocol, or a data flow.
- **host/src_host/dst_host:** where *host* indicates the vulnerable host (device or system), *src_host* refers to the source host in the case of vulnerabilities related to communication protocols, and *dst_host* refers to the destination host when dealing with vulnerabilities in communication between devices.

- **service:** the name of the service that is vulnerable. This applies only if the type of vulnerability is related to a service.
- **subnet:** specifies the subnet or local network where the vulnerability is located. This applies only to vulnerabilities related to network protocols.
- **l2_protocol:** the layer 2 (L2) network protocol that is vulnerable. This refers to layer 2 protocols of the OSI model (e.g., ARP, Ethernet) that operate at the local network level.
- **e2e_protocol:** the end-to-end protocol that is vulnerable. This applies only to vulnerabilities related to communication between two devices.
- **port:** the network port used by the vulnerable protocol (typically for communication protocols such as MQTT).
- **range:** indicates the scope of the attack or the distance from which the vulnerability can be exploited. For example, the vulnerability may be exploitable remotely or only by systems within the same local network or subnet.
- **consequence:** defines the consequence or impact of exploiting the vulnerability. For example, we might have: dos (Denial of Service), impersonateDst (Ability to impersonate the destination, such as intercepting traffic), eavesdropping (Ability to intercept communications), and so on.
- **data_flow:** specifies the vulnerable data flow, such as the data transferred between two devices or within a system.
- **vul:** specifies the nature of the data flow vulnerability. For example: unencrypted (data that is not encrypted and is vulnerable to interception).

Below, is an example of a diagram representing the fields necessary for understanding the network:

```
"vulnerabilities": [
    {
        "id": 1,
        "cve": "CVE-2022-25304",
```

```
            "type": "service",
            "host": "plc1",
            "service": "opcuaServer",
            "range": "remoteExploit",
            "consequence": "dos"
        },
        {
            "id": 4,
            "cve": "CVE-1999-0667",
            "type": "l2_protocol",
            "subnet": "controlSubnet",
            "l2_protocol": "arp",
            "range": "adjacent",
            "consequence": "impersonateDst"
        },
        {
            "id": 8,
            "cve": "",
            "type": "e2e_protocol",
            "src_host": "plc3",
            "dst_host": "historian",
            "e2e_protocol": "mqtt",
            "port": 1883,
            "range": "adjacent",
            "consequence": "eavesdropping"
        }
    ]
```

In this case, the example is provided in JSON format, but this does not imply that it is a requirement for the tool to function correctly, as it is merely an implementation choice.

3.1.2 Attack paths

Regarding attack graphs, a file is required to describe each attack path, starting from the source node to the final node, which may or may not be the target. Each node in the path must be described by the following fields:

- **id:** A unique identifier that characterizes a node in the network.
- **exploit:** Type of exploit or technique used in the specific step. This field describes the method through which the vulnerability is exploited.

- **vul_ids:** List of vulnerability IDs exploited in this step. The IDs refer to the specific vulnerabilities identified in the vulnerability list (as described earlier).

Below, is an example of a diagram representing the fields necessary for understanding the attack paths:


```
"path_1": [
    {
        "id": 16,
        "exploit": "execCode",
        "vul_ids": [
            9
        ]
    },
    {
        "id": 1,
        "exploit": "accessDataFlow",
        "vul_ids": [
            10,
            5,
            6
        ]
    }
],
"path_2": [
    {
        "id": 16,
        "exploit": "execCode",
        "vul_ids": [
            9
        ]
    },
    {
        "id": 32,
        "exploit": "accessDataFlow",
        "vul_ids": [
            11,
            5,
            7
```

```
                        ]
                    }
                ]
```

In this case, as well, the example is provided in JSON format, but this does not imply that it is a requirement for the tool to function correctly, as it is merely an implementation choice.

3.1.3 CVEs list

Regarding the CVEs, after conducting various tests to assign an absolute value to each weakness in the network based on predetermined parameters, it was decided to rely on an existing and standardized database. This approach ensures robustness and consistency in the initial definition phase, providing greater credibility in the final stage of the result analysis. The database used is the National Vulnerability Database (NVD), a repository of standards-based vulnerability data in JSON format. The NVD is maintained by the National Institute of Standards and Technology (NIST), a U.S. federal government agency, and is provided as a public service. Much of the data in NVD records is derived from publicly available sources, including product and manufacturer/developer information. The NVD includes databases of security checklist references, security-related software flaws, product names, and impact metrics. Specifically, there are three key values required in our model, which are:

- **cveId:** the vulnerability ID in the format CVE-AAAA-NNNN, where AAAA indicates the year of cataloging and NNNN the vulnerability number.
- **ExploitabilityScore:** value corresponding to the exploitability score of the current vulnerability.
- **ImpactScore:** value corresponding to the impact score of the current vulnerability.

These scores can be obtained through the CVSS score that characterizes each CVE. So, it is possible to find them in the *impact* field for older vulnerabilities that only include version 2 of the CVSS, for example:

```
        "cve" : {
```

```
            "data_type" : "CVE",
            "data_format" : "MITRE",
            "data_version" : "4.0",
            "CVE_data_meta" : {
                "ID" : "CVE-2003-0001",
                "ASSIGNER" : "cve@mitre.org"
        },
        ...
        ...
        "impact" : {
            "baseMetricV2" : {
                "cvssV2" : {
                    "version" : "2.0",
                    "vectorString" : AV:N/AC:L/Au:N/C:P/I:N/A:N",
                    "accessVector" : "NETWORK",
                    "accessComplexity" : "LOW",
                    "authentication" : "NONE",
                    "confidentialityImpact" : "PARTIAL",
                    "integrityImpact" : "NONE",
                    "availabilityImpact" : "NONE",
                    "baseScore" : 5.0
                },
                "severity" : "MEDIUM",
                "exploitabilityScore" : 10.0,
                "impactScore" : 2.9,
                "obtainAllPrivilege" : false,
                "obtainUserPrivilege" : false,
                "obtainOtherPrivilege" : false,
                "userInteractionRequired" : false
            }
        }
```

Alternatively, they can be calculated based on the vector string, which is a textual representation of a set of CVSS metrics, specifically:

Impact Score

$$6.42 * (1 - [(1 - Confidentiality) * (1 - Integrity) * (1 - Availability)])$$

Where *Confidentiality* evaluates how a successfully exploited vulnerability affects the

confidentiality of the information resources that a software component manages. Maintaining confidentiality means preventing unauthorized users from accessing or disclosing information, as well as restricting access and disclosure to only authorized users. *Integrity* evaluates how a successfully exploited vulnerability affects integrity. The trustworthiness and precision of information are referred to by the term integrity. *Availability* evaluates how a successfully exploited vulnerability affects the affected component's availability. This metric refers to the loss of availability of the impacted component itself, such as a networked service (e.g., web, database, email), whereas the Confidentiality and Integrity impact metrics apply to the loss of confidentiality or integrity of data (e.g., information, files) used by the impacted component. The availability of a compromised component is affected by attacks that use up disk space, CPU cycles, or network bandwidth since availability relates to the accessibility of information resources. *6.42* is a predefined constant [33].

Based on the vulnerability, a certain value is assigned to each metric, specifically:

- Confidentiality (C) / Integrity (I) / Availability (A)
  - None (N): 0
  - Low (L): 0.22
  - High (H): 0.56

Exploitability Score

$$8.22 * AV * AC * PR * UI$$

Where *Attack Vector (AV)* represents the circumstances that allow the exploitation of vulnerabilities. The more physically and logically distant an attacker can be to exploit the vulnerable component, the higher this metric number will be. It is assumed that a vulnerability that can be exploited from anywhere on the Internet gets a higher score because of more possible attackers that could exploit it than a vulnerability that requires physical access to a device. *Attack Complexity (AC)* explains the unforeseeable conditions that must exist for the attacker to take advantage of the vulnerability. Such circumstances might necessitate the gathering of additional target data, the existence of specific system configuration parameters, or computational exceptions. The least sophisticated attacks get the highest value of this statistic. *Privileges Required (PR)* explains the amount of privileges an attacker needs to have to effectively take advantage of the vulnerability. If no

privileges are needed, this metric has the greatest value. *User Interaction (UI)* highlights the need for a user, aside from the attacker, to be involved in the successful breach of the component that is at risk. This metric establishes whether the vulnerability may be exploited only by the attacker or if some kind of involvement from a different user (or user-initiated procedure) is required. When no user engagement is necessary, this metric value is at its highest. *8.22* is a predefined constant [33].

Like before, the values are:

- Access Vector (AV)
  - Network (N): 0.85
  - Adjacent (A): 0.62
  - Local (L): 0.55
  - Physical (P): 0.2
- Attack Complexity (AC)
  - Low (L): 0.77
  - High (H): 0.44
- Privileges Required (PR)
  - None (N): 0.85
  - Low (L): 0.62
  - High (H): 0.27
- User Interaction (UI)
  - None (N): 0.85
  - Required (R): 0.62

The resulting database is presented as the example below in Table 3.1 and is in CSV format:

In conclusion, the use of this database has been especially advantageous as it has allowed me to derive a starting value for both the impact score and the exploitability score for each CVE present on the nodes of the analyzed network. The decision to utilize this database was made to ensure a globally accepted and reliable assessment basis, rather

cveId	ExploitabilityScore	ImpactScore
CVE-1999-0001	10.0	2.9
CVE-1999-0001	10.0	10.0
CVE-1999-0001	10.0	10.0

Table 3.1: database example

than starting with assigned values that lacked a specific study foundation. As will be demonstrated, these values serve only as a foundation, as they will be recalculated and adjusted before assessing risk to account for dependencies among nodes that vary based on the analyzed network.

3.1.4 CWEs list

Regarding misconfigurations, since they are non-standardized issues that are not assessed using metrics such as CVSS, there was a need to utilize a globally recognized list of weaknesses to ensure data consistency. The database used was sourced from the MITRE Corporation [34]. The Common Weakness Enumeration (CWE) is a community-developed list of common software and hardware weaknesses. A "weakness" refers to a condition in software, firmware, hardware, or service components that, under certain circumstances, could lead to the introduction of vulnerabilities. The CWE List and its associated classification taxonomy identify and describe weaknesses in terms of CWEs. Understanding the weaknesses that lead to vulnerabilities allows software developers, hardware designers, and security architects to eliminate them before deployment when it is easier and cheaper to do so. The CWE List is updated three to four times a year to include new weaknesses and update existing ones.

Starting from the original database in XML format, a new database was created in CSV format containing the following fields:

- **Name:** the name of the weakness being considered.
- **Description:** the description of the current weakness, constructed by merging two fields from the original database: 'Description' and 'Extended_Description'. This combination was made to provide a more detailed and comprehensive overall description. In this case, various tests were conducted on the construction of the 'Description' field, as it initially included an additional section of the database.

However, this section was later found to be inconsistently present across CWEs and, most of the time, worsened the quality of the descriptions, making them less relevant.

- **References:** a list of CVEs associated with the specific weakness, which will be used during execution as the basis for calculating the scores.
- **Preprocessed_description:** this field was not present in the original database but was added later. It represents the pre-processed description of the weakness, the construction of which will be analyzed later.

In summary, this input is very important because, starting from a predefined and standardized database, it allows the tool to utilize it to identify the categories of weaknesses associated with the misconfigurations present in the analyzed network. This enables the subsequent retrieval of the CVEs associated with each corresponding CWE entry. Through the CVEs, the tool can derive an initial score for the misconfigurations, for which otherwise there would have been no starting value for calculating impacts, exploitability, and risk.

3.1.5 Misconfigurations

Misconfigurations are issues that do not correspond to a specific, cataloged vulnerability but are general problems that can leave multiple attack vectors open, which can be exploited in various ways. Currently, there is no comprehensive database of known misconfigurations, which led to the need to build an entirely new database in CSV format(subject to continuous updates) to collect all identified and detectable misconfigurations in the analyzed networks. The fields that make up this database are:

- **Name:** it represents the name of the misconfiguration and must match the one found in the network description in the vulnerability section.
- **Description:** this field contains a detailed description of the risks and exploitation methods of a specific misconfiguration. This description is particularly important and must be as accurate as possible, as it will be used in the classification phase into a specific CWE group. This is necessary to identify which CVEs closely resemble this issue and to allow for the assignment of an initial impact score and exploitability score, even for these problems, which do not have a CVSS value.
- **Preprocessed_description:** as in the previous case, this field is dedicated to the pre-processed description.

- **Top_Similar_Labels:** this field contains the name of the three CWEs with the descriptions most similar to the one defined for the misconfiguration, from which the CVEs can be extracted to calculate the initial scores. Various tests were conducted using either a single CWE, the most similar one, to increase precision, or the five most similar CWEs to enhance accuracy. In the end, it was decided to use an average of the three most similar CWEs, striking a balance between precision and accuracy.

- **Top_Similarity_Scores:** For the three most similar CWEs, there are three values respectively, calculated using cosine similarity. The *higher* the value, the more similar the descriptions are.

- **Top_Euclidean_Distances:** For the three most similar CWEs, there are three values respectively, calculated using Euclidean distance. The *lower* the value, the more similar the descriptions are.

- **Top_Jaccard_Distances:** For the three most similar CWEs, there are three values respectively, calculated using Jaccard distance. The *higher* the value, the more similar the descriptions are.

- **Top_Levenshtein_Distances:** For the three most similar CWEs, there are three values respectively, calculated using Levenshtein distance. The *lower* the value, the more similar the descriptions are.

3.1.6 Asset value

The asset value is one of the metrics that make up the environmental factor, which will be analyzed later, and represents the importance level of a node or asset within the network. This value is particularly useful because it influences the results by assigning a higher risk level, and thus greater danger, to vulnerabilities present on critical assets that require priority protection. The asset value is defined on a scale from 1 to 5, where 1 represents the default minimum level assigned to each element, while 5 represents the maximum level for critical assets. This value can be manually defined by the network manager, who has an exhaustive understanding of their assets, or it can be automatically assigned during the network analysis based on the topology and protocols used. Essentially, after assigning the maximum value x to critical assets, the process can proceed recursively by assigning the value $x - 1$ to all nodes that are directly connected and present in the previous step of

the attack paths. This approach progressively decreases the risk for nodes leading to the compromise of the critical asset. The operation should continue until $x \geq 1$.

In this case, as well, it was necessary to build a new database in CSV format, which can be constructed in two different ways. The first is simply a database containing a list of IDs for the critical elements. The second is a database containing two fields: the ID of the vulnerable element and its respective asset value.

3.2 Input elaboration

The previously described inputs are not all created manually, but only in part, and therefore are not initially usable. To make these inputs available and ready for use, specific scripts have been developed. These scripts will now be described:

1. CVE database elaboration script:
    - This script was used to process and essentially filter the National Vulnerability Database (NVD), which contains hundreds of thousands of CVEs. As described earlier, for each CVE, its ID, impact score, and exploitability score were extracted or calculated to build the new database. The script takes a set of JSON files as input and is fully automatic. This means that when the original database is updated, running the script is sufficient to update the CSV database used by the developed tool.
2. CWE database processing script:
    - This script thoroughly analyzes the Common Weakness Enumeration (CWE), which is an XML database provided by MITRE, and extracts or generates the fields previously described.
3. Unified pre-processing script:
    - A script was created to pre-process the descriptions related to the CSV-formatted CWE database and the misconfigurations database through the utilization of the spacy python library. Pre-processing is necessary to clean and standardize the descriptions through a series of operations, listed below, to enable a more precise and accurate analysis of the significant words on which the future classification is based, disregarding unnecessary characters. The operations performed on the descriptions are:

- Special character removal: all special characters, punctuation, and other non-essential symbols are removed.
- Lowercasing: converts all uppercase letters to lowercase to ensure uniformity.
- Conjunction removal: conjunctions and stop words are filtered out to focus on the key components of the descriptions.
- Trimming: all spacing characters, such as tabs or line breaks (e.g., newlines), have been removed.

Once the preprocessing is complete, the script adds a new field to the database called Preprocessed_Description, where the cleaned result is stored.

4. Training and prediction:

- In this script, the pre-processed descriptions are used from the respective databases cweDB and toPredict to perform predictions. First, both files are loaded, the descriptions are tokenized and by using the Gensim library along with FastText, a vocabulary and train a model are built. This model will then be used to generate vectors corresponding to each description. For every description in toPredict, the similarity distances are performed with all the descriptions in cweDB to find the most similar ones. Four metrics were selected for this purpose:
  - Cosine Similarity (from sklearn): A measure of similarity between two non-zero vectors that calculates the cosine of the angle between them. It's commonly used in text comparison for extracting and analyzing data.
  - Euclidean Distance (from sklearn): A distance metric that measures the straight line between two points (or vectors), giving a sense of dissimilarity.
  - Jaccard Distance: A set-based similarity measure that computes the dissimilarity between two sets as the size of their intersection divided by the size of their union.
  - Levenshtein Distance: A string-based metric that calculates the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one word into another.

  At the end of the calculations, the values are normalized and the metrics are ranked according to an index derived from Cosine Similarity. The choice of cosine similarity was based on its effectiveness as a heuristic technique for measuring the similarity between two vectors, particularly in text comparison tasks such as

data extraction and textual analysis. In the final toPredict database are added five new fields:

  - The first field contains a list of the names of the three most similar CWEs.
  - The remaining four fields store the respective distance metrics for each CWE.

Now that all the inputs are correctly set up, it is possible to analyze how they will be utilized by the tool to produce the expected results.

3.2.1 Total Impact Assessment for Each Vulnerability

- **Direct Impact of X:** consider the direct or absolute impact score of vulnerability X on assets that were taken by NVD database.
- **Conditional Impact:** assess the additional impact that vulnerability Y could have on assets if X is exploited. This may include more severe damage to data or services.
- **Total Impact of X:** if there is a vulnerability Z that depends on a vulnerability Y, which in turn depends on X, it is necessary to perform a weighted sum based on the exploitability scores taken by the NVD database, so the absolute probabilities, to generate this type of value:

$$totalImpactX = impactX + pX * (impactY + pY * (impactZ + ...))$$

.

Where $impactX$ and pX are respectively the impact score and the exploitability score of a specific vulnerability.

3.2.2 Total Exploitability Probability Assessment for Each Vulnerability

- **Direct Probability of X:** consider the direct or absolute exploitability score of vulnerability X on assets that were taken by NVD database.
- **Conditional Probability:** The probability of exploiting vulnerability Y should be assessed conditionally on the exploitation of vulnerability X. If X is not exploited, the probability of exploiting Y is zero. If X is exploited, evaluate the probability of exploiting Y separately.

- **Total Exploitability of X:** Multiply the probability of exploiting vulnerability X by the probability of exploiting Y given the exploitation of X. For example, if the probability of exploiting X is 0.6 and the probability of exploiting Y given the exploitation of X is 0.8, the overall probability of exploiting Y is $0.6 * 0.8 = 0.48$ so:

$$pTotY = pX * pY$$

3.2.3 Final Risk Assessment

Each vulnerability will have an associated risk level, which will correspond to its total calculated impact score, multiplied by its exploitability score, and further multiplied by the environmental factor associated with that specific vulnerability, mentioned in the previous section and composed of various metrics:

$$riskX = impactX * pX * envFactorX$$

Analyzing this environmental factor in more detail, in addition to the asset value, which is a parameter that cannot be directly calculated by the tool and is therefore required externally, six other metrics have been selected. These metrics can be calculated through the tool and can be appropriately combined to enrich the final result by adding information about the assets themselves and the topology of the attack network under analysis. This way, low-risk vulnerabilities can be ranked higher because they are located on critical nodes that require a high level of security. Below, the various metrics will be examined from both a theoretical and mathematical perspective.

Betweenness Centrality

Betweenness centrality measures how often a node lies on the shortest path between other nodes. It highlights nodes that act as bridges or control points for information flow across the network. Nodes with high betweenness centrality can potentially control or limit the spread of vulnerabilities, making them critical for network security.

The betweenness centrality of a node v is calculated as:

$$C_B(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

Where:

- σ_{st} is the total number of shortest paths from node s to node t.
- $\sigma_{st}(v)$ is the number of those shortest paths that pass through node v.

Future development could involve identifying nodes with high betweenness and prioritizing their protection, as their compromise could severely impact the network.

Topological Importance: Root and Leaf Proximity

In the network, nodes near a root (with no incoming links) or a leaf (with no outgoing links) are topologically significant. A node closer to a root may be critical for launching attacks, while one near a leaf could be a target node in attack chains.

One way to quantify this is by measuring the distance of each node from the root or the leaf:

$$\text{Root Proximity} = \frac{1}{d_{\text{root}}(v)} \quad \text{Leaf Proximity} = \frac{1}{d_{\text{leaf}}(v)}$$

Where $d_{\text{root}}(v)$ and $d_{\text{leaf}}(v)$ represent the shortest distance from the node to the root and the leaf, respectively. Nodes closer to the root might be given higher priority in attack mitigation.

PageRank

PageRank is a popular algorithm used to rank web pages but can be adapted to measure node importance in a network. It considers both the number and quality of incoming links. A node is considered more important if it is linked to other important nodes.

The PageRank score $PR(v)$ of a node v is given by:

$$PR(v) = \frac{1-d}{N} + d \sum_{u \in L(v)} \frac{PR(u)}{|L(u)|}$$

Where:

- $PR(v)$ is the PageRank of node v.
- d is the damping factor, a parameter that controls how much of the score is "transferred" to the neighbors of a node.

- N is the total number of nodes in the network.
- $L(v)$ represents the set of nodes that point to node v (its incoming neighbors).
- $|L(u)|$ is the number of outgoing links from node u.

Explanation of the Parameters:

- $(1-d)/N$: This term represents the probability that a random user "teleports" to any node in the network. In other words, there is always a small probability that the score is distributed evenly across all nodes, even those without incoming links. The value $(1-d)$ is usually very small and ensures that the model is not too dependent on the link structure, giving every node a minimum score. Typically, d is set to 0.85, meaning $(1-d)$ is 0.15, which distributes a small amount of PageRank uniformly across all nodes in the network.
- $d\sum_{u\in L(v)} \frac{PR(u)}{|L(u)|}$: This term reflects the amount of PageRank that node v receives from its neighbors u. The score received from each neighbor is proportional to its own PageRank ($PR(u)$) but is distributed across all nodes that u links to. Thus, if u has many outgoing links ($|L(u)|$ is high), its contribution to the PageRank of v will be lower.

The Role of the Damping Factor (d)

- Damping factor d, usually set to 0.85, represents the probability that a user follows the network's links rather than teleporting to a random node. In other words, d controls how much weight is given to the link structure versus uniform score distribution.
  - A value of d close to 1 means the importance of the network's links is very strong: most of a node's PageRank comes from the nodes that link to it.
  - A lower value of d (e.g., 0.5) means there is a higher probability that the score is distributed randomly to nodes, reducing the importance of links in the network.

How to Adapt PageRank in a Vulnerability Context

PageRank can be adapted to identify "central" or critical nodes in a vulnerability network that require higher protection. Some possible adaptations include:

- Modifying the damping factor: In vulnerability networks, the damping factor can be adjusted to reflect the likelihood that an attacker follows a vulnerable path. If we assume that an attacker always chooses strategic, well-connected paths, the damping factor could be increased (e.g., $d = 0.9$). On the other hand, if the attacker might choose random targets, d could be lowered.

- Directional networks: In a vulnerability network, the links could be directional, representing data flows or access paths. For example, node A might be vulnerable because it is connected to node B, which is already compromised. In this context, PageRank can be used to identify nodes that inherit vulnerabilities from compromised nodes.

- Adjusting link weights: The contribution a node v receives from its neighbors u could be weighted based on the severity of the vulnerabilities on u. This way, not all links carry the same weight, and more vulnerable or critical nodes contribute more to the PageRank of the nodes they connect to.

In conclusion, PageRank is a powerful tool that can be used not only to assess the importance of nodes in a network but also to make informed decisions on where to focus vulnerability mitigation efforts. By adjusting parameters like the damping factor or weighting the links based on vulnerability severity, the algorithm can be made even more useful in the context of cybersecurity.

Resilience

Resilience measures how well a network can maintain its structure and functionality when nodes or edges are removed. A resilient network can withstand attacks or failures without significant disruption.

This can be measured using the concept of network robustness, often based on the size of the largest connected component after node removal. The assessment of resilience in the context of our vulnerability network can be adapted through the following steps:

1. Identify Direct and Indirect Descendants: For a given node n, we first need to identify all of its direct and indirect descendants. Direct descendants are those nodes that are immediately linked to n, while indirect descendants are nodes that can be reached by following paths from n through its direct descendants. This step can be accomplished

using a breadth-first search (BFS) or depth-first search (DFS) algorithm, starting from node n.

2. Determine Accessibility: Once we have identified all the descendants, we must evaluate the accessibility of the other nodes in the network. This means checking which nodes become inaccessible when node n and all its descendants are removed from the network.

3. Calculate Resilience: The resilience of node n is defined as the total number of nodes that become inaccessible due to the removal of n and its descendants. This can be expressed as:

$$\text{Resilience}(n) = \text{Total Nodes} - \text{Accessible Nodes}$$

Where "Total Nodes" refers to the total number of nodes in the network, and "Accessible Nodes" refers to the nodes that remain reachable after the removal of n and its descendants.

Understanding the resilience of nodes in the context of vulnerabilities provides crucial insights for risk management. Nodes that, when removed, cause a large number of other nodes to become inaccessible should be prioritized for protection. Conversely, nodes with high resilience may be less critical in terms of immediate risk, allowing for more focused mitigation efforts on vulnerable nodes that impact the overall network connectivity significantly.

By assessing resilience, network administrators can make informed decisions about where to allocate resources and prioritize vulnerability remediation efforts, ultimately leading to a more secure and robust network infrastructure.

Eigenvector Centrality

Eigenvector centrality is a measure of the influence of a node in the network, considering not only the number of its connections but also the importance of its neighbors. A node with high eigenvector centrality is connected to other influential nodes.

Eigenvector centrality for a node v is calculated as:

$$EC(v) = \frac{1}{\lambda} \sum_{u \in N(v)} A_{vu} EC(u)$$

Where:

- A is the adjacency matrix of the network.
- λ is a constant (the largest eigenvalue of the adjacency matrix).
- $N(v)$ represents the neighbors of node v.

Nodes with high eigenvector centrality can be considered highly influential in spreading attacks or vulnerabilities and securing these nodes can be prioritized.

Katz Centrality

Katz centrality extends eigenvector centrality by considering all paths in the network, with more distant nodes being given less weight. It assigns importance to nodes that are connected to other important nodes, even if those connections are indirect.

The Katz centrality of a node v is given by:

$$C_K(v) = \alpha \sum_{u \in N(v)} (A_{vu} C_K(u)) + \beta$$

Where:

- α is a constant that determines the relative influence of the nodes.
- β is a constant that accounts for the initial centrality of the nodes.
- A_{vu} is an element of the adjacency matrix representing a connection between nodes v and u.

Katz centrality is useful in networks where indirect connections are important, and its development could focus on vulnerabilities that arise due to indirect influence or control over distant nodes.

By incorporating these and other metrics, we can enhance the tool's ability to calculate more accurate and context-aware risk levels. This would lead to improved prioritization, helping organizations address the most critical vulnerabilities in a timely and efficient manner.

3.3 Model Behavior

The developed tool follows a precise processing flow and operates cyclically, meaning that after displaying the results, it waits for user input before repeating all calculations and

providing new results. We can divide the flow as follows:

1. **Network construction:** after opening the files that describe the network and the attack paths, the tool proceeds iterating through each path and each node to build two parallel structures:

   - Vulnerability-based structure: this structure represents the network elements in detail, where each element contains information about the node and the vulnerabilities present on that specific node. The vulnerability details are extracted from the network description file, particularly the "vulnerabilities" section. In this structure, a node is represented multiple times, depending on the number of vulnerabilities it contains, with each occurrence being distinguished by the specific vulnerability present on the node.
   - Topology-based structure: this structure represents the entire network, offering a general overview. Each node appears only once in this structure, regardless of the number of vulnerabilities. The goal of this structure is to serve, once the iteration process is completed and the structure is fully constructed, for calculating the environmental factor metrics that rely on the network's topology.

2. **Base value calculation:** once the network structures are available, the next step is to assign the respective impact score and exploitability score to each vulnerability or misconfiguration. There are two possible scenarios during this process:

   - Vulnerabilities: if there is a vulnerability, the tool iterates through the input CVE database to find the associated values (impact score and exploitability score) for the specific vulnerability, and are directly retrieved and assigned.
   - Misconfigurations: if there is a misconfiguration, we cannot directly refer to predefined scores. Therefore, a series of steps are taken to calculate the base values:
     - First, the tool iterates through the input misconfiguration database, extracting the names of the three most similar CWEs (previously calculated).
     - The tool then iterates through the CWE database to find all the references (CVE IDs) associated with each CWE.
     - Finally, the tool iterates again through the CVE database to retrieve the impact and exploitability scores of all the referenced CVEs. These scores

are then summed and averaged to produce the final base scores for the misconfiguration.

This method ensures that even for misconfigurations, which lack direct CVE values, we can calculate starting impact and exploitability scores.

3. **Results calculation:** at this stage, the tool enters the core calculation cycle, where the actual scores for each vulnerability are computed by considering various dependencies, as explained in the previous section. Now, it can perform the calculation of the risk value like the multiplication of impact score, exploitability score, and environmental score.

4. **Output results:** at the end of the process, the tool displays an ordered ranking based on the calculated risk levels (from highest to lowest). The output is structured as follows:

   - Ranking Position: the position of the vulnerability or misconfiguration in the list.
   - Node ID: the identifier of the network node where the vulnerability or misconfiguration is present.
   - CVE ID or Misconfiguration name: the identifier of the specific CVE or the name of the misconfiguration associated with the node.
   - Risk Value: the calculated risk value for the corresponding vulnerability or misconfiguration.

5. **Recalculation:** after reviewing the output, the user will have the opportunity to select the position of a vulnerability or misconfiguration from the ranking that they wish to eliminate, for example, due to its mitigation or removal from the network. Once selected, the tool will remove it from the internal data structures that will be updated recursively, ensuring that any dependencies related to the eliminated vulnerability or misconfiguration are adjusted accordingly. The tool will then recalculate the impact scores, exploitability scores, environmental scores, and the risk values of any network elements that had dependencies on the removed vulnerability and will generate a new ranking based on the updated values and display the revised list.

Chapter 4

Implementation

In this chapter, the implementation of the scripts and the main tool will be described. Each script has a specific purpose and is crucial to the overall process, as they are used to generate the inputs that the main tool will process to produce results. Both the scripts and the main tool were developed using the Python programming language, which provides a wide range of libraries for data management and processing.

4.1 Pre-processing details

In this section, the implementation of the four developed scripts will be described. Each script has a specific function and plays a critical role in preparing the necessary inputs for the main tool.

1. **Filtering and construction of the CVEs database**

   - Purpose: the script is used to process the data from the original database to extract information regarding the CVE ID, exploitability score, and impact score for each vulnerability.
   - Files used:

```
NVDPATH = "../ vulnerability / nvd_json_files"
FILEOUT = "../ vulnerability / vulDB .csv "
```

where *NVDPATH* is the path of the folder containing the download NVD JSON files. *FILEOUT* is the path for the output CSV file where the results are stored.

- Implementation: after opening the new CSV database for writing, there is an iteration over the files as shown below.

```
Data: DEFAULT_VALUE = 0, NVDPATH, FILEOUT

begin
    nvd_files <- list all files in NVDPATH
    sort nvd_files

    open FILEOUT as output_file in write mode
    write header to output_file

    for each filename in nvd_files do
        open NVDPATH/filename as f
        json_object <- load JSON content from f

        for each item in json_object do
            cve_id <- cve_item['ID']
            ExpScore <- DEFAULT_VALUE
            ImpScore <- DEFAULT_VALUE

            if 'MetricV3' exists in cve_item then
                ExpScore <- cve_item['expScoreV3']
                ImpScore <- cve_item['impScoreV3']

            else if 'MetricV2' exists in cve_item then
                ExpScore <- cve_item['expScoreV2']
                ImpScore <- cve_item['impScoreV2']

            if ExpScore = DEFAULT_VALUE and ImpScore =
   DEFAULT_VALUE then
                continue

            write to output_file
        end
    end

```

```
    close output_file
end
```

this is a pseudocode representing a key excerpt from the script.

2. **Filtering and construction of the CWEs database**

- Purpose: this script reads an XML file, extracts the useful information (including names, descriptions, and references) for our purpose, and stores the results in a CSV file.
- File used:

```
tree = ET.parse('../weaknesses/cwes.xml')
root = tree.getroot()
output_file = '../weaknesses/cweDB.csv'
```

where *tree* is the entire tree of the parsed XML file, *root* is the root of the tree just defined and *output_file* is the path of the new CSV database.

- Implementation: after opening the files, there is an iteration on the 'Weakness' fields of the XML database to extract the name of the weakness, the description and the references as a list of CVEs.

```
Data: INPUT_XML, OUTPUT_CSV, NAMESPACE

begin
    tree <- parse XML file INPUT_XML
    root <- get root of XML tree

    w <- find all 'Weakness' elements using NAMESPACE
    items_data <- empty list

    for each item in weaknesses do
        name <- get 'Name' attribute from item

        if name does not start with 'DEPRECATED' then
            des <- get 'Description' element text
            extDes <- ""
```

```
            refs <- empty list

            extDesElem <- find 'ExtDes' element in item
            if extDesElem exists then
                extDes <- concatenate all from extDesElem

            obsEx <- find all 'ObsEx' elements in item
            for each example in obsEx do
                refElem <- find 'Ref' element in example
                if refElem exists then
                    refs.append(trimmed reference text)

            des <- des + " " + extDes
            des <- replace '\n' and '\t' with spaces
            des <- replace multiple spaces with a space
            des <- trim des

            if ref not empty then
                item_data <- {
                    'Name': name,
                    'Des': des,
                    'Refs': refs
                }

                items_data.append(item_data)

    open OUTPUT_CSV in write mode as file
    writer <- create CSV writer
    writer.writeheader()
    writer.write rows from items_data
end
```

this is a pseudocode representing a key excerpt from the script.

3. **Text pre-processing**

- Purpose: this script reads a CSV file containing descriptions, preprocesses the text in the "Description" column and saves the processed text in a new CSV file. The preprocessing involves lemmatization, stopword removal, and text cleaning.
- File used:

```
file_path = '../weaknesses/cweDB.csv'
```

where *file_path* represents the path of the CSV file from which a description is read, preprocessed, and added to a new dedicated column. In the example, the file cweDB.csv is used, but any CSV file can be used as long as it contains a 'Description' field.

- Implementation: after opening the file and reading the description, the text is converted to lowercase, all unnecessary spaces, punctuation marks, and special characters are removed, and a list of meaningful words separated by spaces is reconstructed. Finally, the pre-processed text is saved.

```
Data: INPUT_CSV, OUTPUT_CSV, SPACY_MODEL

begin
    nlp <- load SpaCy model SPACY_MODEL

    function preprocess_text_spacy_english(text):
        doc <- process text.lower() with nlp

        procToken <- reducing words to the base form
        preprocText <- join procTokens into a string
        preprocText <- replace '\n' and tab characters with
    spaces
        preprocText <- replace spaces with a space
        preprocText <- trim spaces from preprocText

        return preprocText
    end function

    df <- read CSV from INPUT_CSV
    df['PreprocDes'] <- apply the func to df['Des']
    write df to OUTPUT_CSV without index
end
```

this is a pseudocode representing a key excerpt from the script.

4. **Similarity calculation for misconfigurations**

- Purpose: this script is used to classify the misconfigurations by calculating 4 appropriately selected metrics, allowing the tool to derive the relevant references needed to compute the two base scores.
- File used:

```
cweDB = pd.read_csv('../weaknesses/cweDB.csv')
toPredict = pd.read_csv('../weaknesses/toPredict.csv')
```

  where *cweDB* is the CWEs database in CSV format from which retrieve a pre-processed description of a specific CWE and *toPredict* is the database of the misconfigurations from which retrieve the pre-processed description useful for the classification.

- Implementation: after opening the files, the descriptions are tokenized and then combined to create a model using FastText. Next, an iteration is performed over the pre-processed misconfiguration descriptions to create a vector, followed by another iteration over the CWE descriptions during which another vector is created and the four metrics are computed. Finally, the predicted CWEs have been ordered, and all calculated values are saved in the misconfiguration database.

```
Data: CWE_CSV, PREDICT_CSV, OUTPUT_CSV, FT_MODEL_CONFIG, EPOCHS
    <- 10

begin
    cweDB <- read CSV from CWE_CSV
    tp <- read CSV from PREDICT_CSV

    cweDBToken <- split 'PreprocDes' from cweDB
    tpToken <- split 'PreprocDes' from tp

    combinedDes <- concatenate cweDBToken and tpToken into a
    list

    model <- initialize FastText with FT_MODEL_CONFIG
    model.build_vocab(combinedDes)
```

```
    model.train(combinedDes, epochs=EPOCHS)

    function calcSimilarity(tpVec, cweDes):
        cweVec <- average vector of words from cweDes
        s <- cosine similarity between tpVec and cweVec
        d <- euclidean distance between tpVec and cweVec
        return s, d
    end function

    function jaccardDistance(set1, set2):
        i <- size of set1 intersection set2
        u <- size of set1 union set2
        jaccard <- 1 - (i / u) if u != 0 else 0
        return jaccard
    end function

    function levenshteinDistance(s1, s2):
        m <- initialize empty matrix
        for i in range(0, len(s1)+1) do
            matrix[i][0] <- i
        for j in range(0, len(s2)+1) do
            matrix[0][j] <- j
        for i in range(1, len(s1)+1) do
            for j in range(1, len(s2)+1) do
                cost <- 0 if s1[i-1] == s2[j-1] else 1
                m[i][j] <- min(calculations + cost)
        return m[len(s1)][len(s2)]
    end function

    function normLevenshteinDistance(s1, s2):
        d <- levenshteinDistance(s1, s2)
        max_length <- max(len(s1), len(s2))
        normDistance <- d / max_length
        return normDistance
    end function

    topLabels <- empty list
    topSimilarities <- empty list
    topDistances <- empty list
    topJaccards <- empty list
    topLevenshtein <- empty list
```

```
55
56      for each row in toPredict do
57          tpDes <- row['PreprocDes']
58          tpVec <- average vector of words from tpDes
59          similarities <- empty list
60
61          for each cweRow in cweDB do
62              cweDes <- cweRow['PreprocDes']
63              s, d <- calculateSimilarity(tpVec, cweDes)
64              j <- jaccardDistance(set(tpDes), set(cweDes))
65              l <- normLevenshteinDistance(tpDes, cweDes)
66              similarities.append(name + metrics)
67
68          scaler <- initialize MinMaxScaler
69          dNormalized <- scaler.fit_transform
70          jNormalized <- scaler.fit_transform
71          lNormalized <- scaler.fit_transform
72
73          for i in range(len(similarities)) do
74              similarities[i] <- update similarity with normalized
        distances
75
76          top_similarities <- sort similarities by s in descending
        order, select top 3
77          top_labels, top_similarity_scores, top_distances,
       top_jaccards, top_levenshtein <- extract relevant info from
       top_similarities
78
79          topLabels.append(top_labels)
80          topSimilarities.append(top_similarity_scores)
81          topDistances.append(top_distances)
82          topJaccards.append(top_jaccards)
83          topLevenshtein.append(top_levenshtein)
84
85      tp['Top_Similar_Labels'] <- all_top_labels
86      tp['Top_Similarity_Scores'] <- all_top_similarities
87      tp['Top_Euclidean_Distances'] <- all_top_distances
88      tp['Top_Jaccard_Distances'] <- all_top_jaccards
89      tp['Top_Levenshtein_Distances'] <- all_top_levenshtein
90
91      write tp to OUTPUT_CSV
```

```
92 end
```

this is a pseudocode representing a key excerpt from the script.

4.2 Algorithm

This section is intended to define the procedure of the implemented tool, which consists of two main classes and various methods that allow the reconstruction of the network under analysis, calculation of the necessary metrics, and assessment of the total risk level for each vulnerability. This ensures a consistent output ranking of vulnerabilities ordered by decreasing risk levels. As for the scripts, the tool has been implemented using the Python programming language. After an initial explanation of the two classes used, the procedure will be described step by step and documented through pseudocode.

4.2.1 Class Definition

Vulnerability

This class was used to represent a vulnerability present in our network, which can refer to a CVE or a misconfiguration. The Vulnerability class consists of the following fields:

- **node:** this field consists of the node ID followed by an underscore and a list of vulnerability IDs separated by commas. An example would be "12_5,6", where 12 indicates the node with ID 12, which is affected by vulnerabilities with IDs 5 and 6.
- **cve:** this field contains the CVEs or the names of the misconfigurations present on this element. For example, with "12_5,6", this field will contain the CVE or the name of the misconfiguration associated with ID 5 and ID 6. While the previous field is a string, the cve field is a list of strings.
- **expAbsScore:** this field contains the absolute value of the exploitability score obtained from the referenced database.
- **exploitabilityScore:** this field contains the final exploitability score, which will be updated in the last phase of the process and not entered beforehand. The calculation will be performed according to the formulas defined in the previous chapter.

- **impAbsScore:** this field contains the absolute value of the exploitability score obtained from the referenced database.
- **impactScore:** this field contains the final impact score, which will be updated in the final phase of the process and not entered beforehand. The calculation will be performed according to the formulas defined in the previous chapter.
- **assetScore:** this field contains information about the importance level of the node, which will be used in the calculation of the environmental score. The value can be obtained from the dedicated database containing these values for each node, or if only the nodes with the highest value are available, the process can be automated with a small script to assign the correct value to each node.
- **riskScore:** this score represents the risk level associated with a vulnerability and will be used to build the final ranking.
- **environmental:** this value encompasses a set of metrics that make up the environmental factor, which will be used during the risk calculation to tailor the result to the type of network being analyzed. The number and type of metrics to include in this factor are at the developer's discretion, but we will dive into this further in the next chapter.
- **fathers:** a list of Vulnerability objects through which this node is reached.
- **children:** a list of nodes that can be reached by exploiting a specific vulnerability present on this node.

Objects of this class, potentially exploited through multiple ways, can appear in several attack paths that exploit different weaknesses. Therefore, a single node may appear in our list of vulnerabilities as many times as there are different types of attacks that can be executed from the node in question.

Node

This class is used to represent the individual nodes that will be used to construct the network, providing information about the topology and allowing us to perform evaluations during the calculations for the environmental score. It consists of the following fields:

- **node:** This field indicates the ID of the node being referenced.

- **fathers:** It is a list of nodes that can be reached, exploiting certain vulnerabilities, to the current node.
- **children:** It is a list of nodes that we can access using the appropriate vulnerabilities.

4.2.2 Procedure

The execution of the program begins with the opening of the two main files: *system* and *attack_paths*.

```
atk_path = "../n_description/attack_paths.json"
sys_path = "../n_description/system.json"
```

Starting with the second file, an iteration is performed over each attack path, and temporary lists are built to store the node IDs that appear and the vulnerability IDs present on each node. From the first file, in the 'vulnerabilities' section, using the IDs found in the paths, it is possible to retrieve the specific CVE code or the name of the misconfiguration. After performing the necessary checks, the tool can link the various objects created in the temporary lists to establish dependencies between the initial node, where the first vulnerability is exploited, and the final node. At the end of each path, the objects are saved in the original lists and the temporary lists are reset.

```
Data: node_list, vul_list

begin
    for z, atk in atk_data.items() do
        tmp_list <- empty list for vulnerabilities
        ntmp_list <- empty list for nodes

        for i, x in enumerate(atk) do
            update of the two temporary lists
            with the node just read

            for vuln in sys_data['vulnerabilities'] do
                for v in vuls_id do
                    if vuln['id'] = v then
```

```
                    if vuln['cve'] != "" then
                        vuls_cve <- vuln['cve']
                        break
                    else
                        vuls_cve <- vuln['consequence']
                        break
                end
            end
        end

        y <- Vulnerability(combined_string, vuls_cve)
        tmp_list <- y
        n <- Node(id)
        ntmp_list <- n
    end

    #iteration to update the node_list
    for i from 1 to len(ntmp_list) - 1 do
        if ntmp_list[i-1] not in node_list then
            if ntmp_list[i] not in node_list then
                update the dependencies father-children
            else
                f <- findVulnerability(ntmp_list[i])
                update the dependencies father-children
        else
            p <- findVulnerability(ntmp_list[i-1])
            if ntmp_list[i] not in node_list then
                update the dependencies father-children
            else
                f <- findVulnerability(ntmp_list[i])
                if f not in p.children then
                    update the children list
                end
                if p not in f.fathers then
                    update the father list
                end
    end

    #check and update the original node_list
    for x in ntmp_list do
        if x not in node_list then
```

```
                node_list.append(x)
            else
                for y in node_list do
                    if x.node = y.node then
                        y <- x
                        break
                    end
                end
        end

        #iteration to update the vul_list
        In this case, the iteration is omitted because it is
        identical to the previous one, with checks performed on
        the other temporary list.

        check the original vul_list and append
        the new inserted vulnerability
    end
```

The use of these temporary lists is essential because, as mentioned earlier, both vulnerabilities and nodes may repeat across different attack paths. Therefore, it's necessary to perform checks to reduce redundancy in the connections and properly manage the removal of nodes or vulnerabilities during tool execution. These checks are handled separately for the node list and the vulnerability list due to differences in their management. For example, if three vulnerabilities can be exploited on a single node, leading to three different outgoing connections, the node will appear three times in the vulnerability list (each associated with a different vulnerability). However, it will appear only once in the node list, as this list provides a global view of the attack network.

Next, the tool will open the three files required to gather the necessary input: *toPredict*, *vulDB*, and *cweDB* (for simplicity it is possible to assume that the asset value is assigned statically otherwise it is needed to open the assets database *assetValueDB*).

```
toPredict <- pandad.read_csv('../n_description/toPredict.csv')
vulDB <- pandas.read_csv('../n_description/vulDB.csv')
cweDB <- pandas.read_csv('../n_description/cweDB.csv')
```

Now that the complete list of nodes is available, the tool can already compute the

environmental factor using the various metrics described in the model definition. Currently, it is possible to use only two of these metrics to calculate this factor: asset value and centrality.

```
for each vul in vul_list do
    for each node in node_list do
        if node.node equals id then
            vul.env <- environmental calculation
            break
        end
    end
end
```

The details of the calculation will be discussed later. At this point, the tool iterates over the list of vulnerabilities. For each vulnerability, it must check the 'cve' field. If a CVE is present, the vulDB database is queried to find the corresponding CVE and retrieve the impact and exploitability values. If, instead, the 'cve' field contains a misconfiguration, the assignment becomes more complex. In this case, the tool accesses the toPredict file and once the corresponding row for the misconfiguration is found, it can retrieve the 'Top_Similar_Labels' field, which contains the top three predicted CWEs. For each CWE, through the cweDB file, which contains all the weaknesses, once on the correct row, the 'References' field, which holds the list of CVEs associated with that specific CWE, is taken. After gathering all the CVEs referenced by the three CWEs for a given misconfiguration, the tool performs the sum of all impact and exploitability scores and then divides the total by the number of scores collected, effectively calculating an average score.

```
for each vul in vul_list do
  tmpc <- ""
  numberOfC <- -1

  for each c in vul.cve do
    numberOfC <- numberOfC + 1

    if c starts with 'CVE' then
      for each row in vulDB do
        if row['cveId'] equals c then
```

```
11          vul.expAbsScore <- row['ExpScore'] / 10
12          vul.impAbsScore <- row['ImpScore']
13          tmpc <- c
14          break
15        end
16      end
17
18    else
19      for each row in toPredict do
20        if row['Name'] equals c then
21          labels <- evaluate row['TopSimilarLabels']
22          tmp_exp_score <- 0
23          tmp_imp_score <- 0
24          tmp_count <- 0
25
26          for each label in labels do
27            for each cwe in cweDB do
28              if cwe['Name'] equals label then
29                references <- evaluate cwe['Refs']
30
31                for each ref in references do
32                  for each row in vulDB do
33                    if row['cveId'] equals ref then
34                      tmp_exp_score <- sum the ExpScore
35                      tmp_imp_score <- sum the ImpScore
36                      tmp_count <- tmp_count + 1
37                      break
38                    end if
39                  end for
40                end for
41              end if
42            end for
43          end for
44
45          if tmp_count != 0 then
46            vul.expAbsScore <- (tmp_exp_score / tmp_count) / 10
47            vul.impAbsScore <- (tmp_imp_score / tmp_count)
48            tmpc <- c
49          end if
50        end if
51      end for
```

```
  end for
end for
```

Now, the tool has all the necessary data to compute the results. At this point, a loop is present where, to begin, the calculation of various values is performed using a wrapper function, as shown.

```
function calculateExploitabilityScore(vul):
    if vul has no fathers:
        return vul.expAbsScore
    else if vul.exploitabilityScore != 0:
        return vul.expScore
    else:
        scores <- empty dictionary
        mul <- vul.expAbsScore

        for each father f in vul.fathers do:
            scores[f] <- mul * calculateExploitabilityScore(f)
        end for

        return the maximum value in scores
    end if
end function

function calculateImpactScore(vul):
    if vul has no children:
        return vul.impAbsScore
    else if vul.impactScore != 0:
        return vul.impactScore
    else:
        scores <- empty dictionary
        sum <- vul.impAbsScore
        mul <- vul.expAbsScore

        for each child c in vul.children do:
            scores[c] <- mul * calculateImpactScore(c)
        end for

        for each score s in scores do:
```

```
            sum <- sum + s
        end for

        return sum
    end if
end function

function elaboration(list_vul):
    for each vul in list_vul do:
        if vul.exploitabilityScore = 0 then:
            vul.expScore <- calculateExploitabilityScore(vul)
        end if
    end for

    for each vul in list_vul do:
        if vul.impScore = 0 then:
            vul.impScore <- calculateImpactScore(vul)
        end if
    end for

    for each vul in list_vul do:
        vul.riskScore <- vul.expScore*vul.impScore*vul.envScore
    end for
end function
```

Once the final risk is calculated, the tool proceeds to print the output, which includes the ranking position, the node where the vulnerability is located, the CVE or the misconfiguration name, and the associated risk level. When the user inserts the number of the weakness to remove, the tool will remove it, potentially also removing the corresponding node if necessary. The updated ranking will then be recalculated and displayed, and the tool will wait again for another user input.

```
#Used for the node removal
function remove_node(list_node, list_vul, vul):
    listaFigli <- empty list

    for each v in list_vul do:
        if v.node equals vul.node then:
            for each child c in v.children do:
```

```
                listaFigli <- c
            end
        end
    end

    tmpf <- empty list

    for each n in list_node do:
        if n.node equals vul.node then:
            for each child c in n.children do:
                flag <- None

                for each lf in listaFigli do:
                    if c.node equals lf.node then:
                        flag <- True
                        break
                    end
                end

                if flag equals None then:
                    tmpf <- c
                end
            end

            for each c in tmpf do:
                if c.fathers equals 1 then:
                    remove_node_recursive(c, list_node)
                end
                remove n from c fathers
                remove c from n children
            end

            if length(n.children) equals 0 then:
                for each f in n.fathers do:
                    remove n from each of f father
                end
                remove the node n
            end
        end
    end

```

```
49      return 1
50  end function
51
52  function remove_node_recursive(node, list_node):
53      if node.children equals 0 and node.fathers equals 0 then:
54          remove node from list_node
55          return 1
56      end if
57
58      for each child c in node.children do:
59          remove node from c fathers
60          if c.fathers equals 0 then:
61              remove_node_recursive(c, list_node)
62          end if
63      end for
64
65      remove node from list_node
66      return 1
67  end function
68
69  #Used for vulnerability removal
70  function remove_vulnerability(list_vul, to_remove):
71      toSearch <- id to remove
72
73      for each vul in list_vul do:
74          if vul.node equals to_remove.node then:
75              numbers <- ids of vulnerabilities
76              if toSearch is in numbers then:
77                  for each father f in vul.fathers do:
78                      remove vul from f children
79                      f.impactScore <- 0
80                      f.exploitabilityScore <- 0
81                      reset_fathers_values(f)
82                  end for
83
84                  remove_recursive(vul, list_vul)
85                  break
86              end if
87          end if
88      end for
89
```

```
90      return 1
91  end function
92
93  function remove_recursive(vul, list_vul):
94      if vul.children equals 0 and vul.fathers equals 0 then:
95          remove vul from list_vul
96          return 1
97      end if
98
99      for each child c in vul.children do:
100         remove vul from c fathers
101         c.impactScore <- 0
102         c.exploitabilityScore <- 0
103         if c.fathers equals 0 then:
104             remove_recursive(c, list_vul)
105         end if
106     end for
107
108     remove vul from list_vul
109     return 1
110 end function
111
112 function reset_fathers_values(vul):
113     if vul.fathers equals 0 then:
114         return 1
115     end if
116
117     for each father f in vul.fathers do:
118         f.impactScore <- 0
119         f.exploitabilityScore <- 0
120         reset_fathers_values(f)
121     end for
122
123     return 1
124 end function
```

Chapter 5

Experimental evaluation

In this chapter, we will analyze the results produced by the tool for two example networks. We will start by evaluating the base version of the tool, where the vulnerability ranking is calculated purely based on impact and exploitability values without considering any environmental or external factors. Afterward, we will examine how the incorporation of the asset value influences the associated risk levels, leading to changes in the vulnerability rankings as different vulnerabilities are assigned new levels of priority. Next, we will assess the results taking into account the centrality of a node within the network. This will show how the importance of a node in the network structure affects the overall risk calculation. Finally, we will look at the results when both metrics, asset value, and centrality are combined.

5.1 Networks structures

Below we can observe the structure of my networks, which consist of all the attack paths. For my project, I worked with two networks where the node IDs are indicated inside the ovals, while the vulnerability IDs associated with those nodes are shown in square brackets:

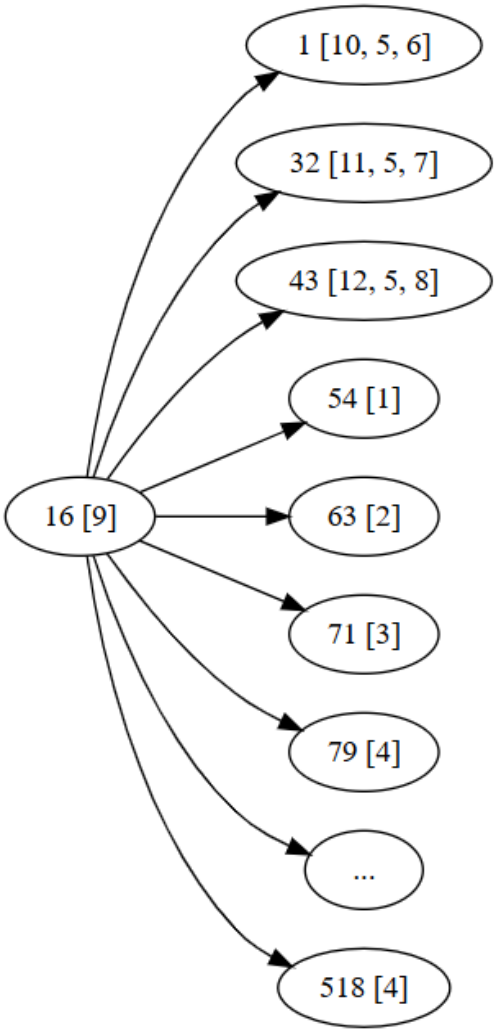

Figure 5.1: First network

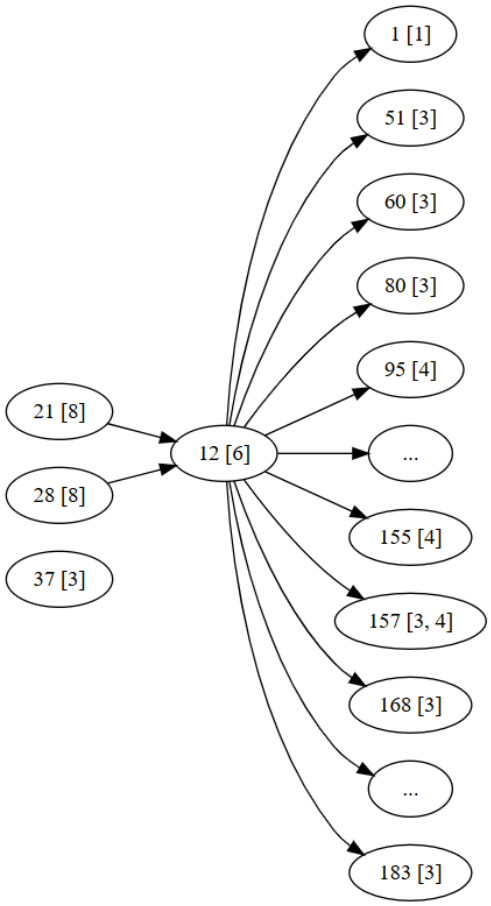

Figure 5.2: Second network

Each path in these networks, starting from a node without any incoming connections (i.e., a root node) and ending at a node without any outgoing connections (i.e., a leaf node), is considered a single attack path. This network structure allows us to model the various possible attack routes, from the initial access points to the final points of compromise. By analyzing these paths, we can evaluate how vulnerabilities propagate through the network and prioritize them based on the risk they pose.

5.2 Baseline results

In this section, I will present the output generated by my tool, which consists of a ranking of 22 vulnerabilities, along with the corresponding node IDs and the calculated risk level. These results are based solely on the basic values of impact and exploitability, without considering any environmental factors. For completeness, I will display the tool's results applied to the second network presented, as it contains a multi-layered tree structure with more connections, providing a more comprehensive example. By focusing on the second network, we can see how vulnerabilities propagate across different levels of the network and how their prioritization changes based solely on inherent risk factors. This initial ranking will serve as a baseline, allowing us to later observe how the inclusion of environmental factors such as asset value and node centrality impacts the final ranking of vulnerabilities and shifts their priority.

```
Vulnerabilities sorted by risk score
1) 12 - CVE-2015-0991 - 67.56
2) 21 - accessControlBypass - 27.19
3) 28 - accessControlBypass - 27.19
4) 37 - CVE-1999-0667 - 10.0
5) 51 - CVE-1999-0667 - 3.83
6) 60 - CVE-1999-0667 - 3.83
7) 80 - CVE-1999-0667 - 3.83
8) 95 - CVE-1999-0667 - 3.83
9) 106 - CVE-1999-0667 - 3.83
10) 121 - CVE-1999-0667 - 3.83
11) 132 - CVE-1999-0667 - 3.83
12) 134 - CVE-1999-0667 - 3.83
13) 140 - CVE-1999-0667 - 3.83
14) 151 - CVE-1999-0667 - 3.83
15) 153 - CVE-1999-0667 - 3.83
16) 155 - CVE-1999-0667 - 3.83
17) 157 - CVE-1999-0667 - 3.83
18) 168 - CVE-1999-0667 - 3.83
19) 179 - CVE-1999-0667 - 3.83
20) 181 - CVE-1999-0667 - 3.83
```

```
21) 183 - CVE-1999-0667 - 3.83
22) 1 - CVE-2022-25304 - 0.54
Enter the number of the vulnerability to remove(x to exit):
```

At this stage, the tool will await user interaction regarding a vulnerability that, for any reason, such as its resolution, needs to be removed from the ranking. Upon removal, the tool will recalculate all relevant risk values. The removal or addition of elements to the ranking can potentially be automated in future development phases, where the ranking could be auto-updated multiple times a day. However, for now, without a physical network available and without an already automated network analysis, user interaction is essential for producing results. Moving forward, let's assume we decide to remove the vulnerability ranked at position 9, the user must write the number 9. In this case, the tool will recognize that the user is referring to the vulnerability ranked in the 9th position, which is CVE-1999-0667 on the node with ID 106 and a risk factor of 3.83.

```
9) 106 - CVE-1999-0667 - 3.83
```

Once this vulnerability is removed, the tool will recalculate the rankings and risk factors accordingly, accounting for the updated attack paths and connections. This recalculation will be particularly important, as removing a vulnerability may impact multiple nodes and affect both the exploitability and impact scores for other vulnerabilities in the network. After this recalculation, the tool will display the updated ranking, allowing us to observe how the absence of a single vulnerability affects the overall network risk and vulnerability prioritization. The result is:

```
Vulnerabilities sorted by risk score
1) 12 - CVE-2015-0991 - 63.73
2) 21 - accessControlBypass - 25.73
3) 28 - accessControlBypass - 25.73
4) 37 - CVE-1999-0667 - 10.0
5) 51 - CVE-1999-0667 - 3.83
6) 60 - CVE-1999-0667 - 3.83
7) 80 - CVE-1999-0667 - 3.83
8) 95 - CVE-1999-0667 - 3.83
9) 121 - CVE-1999-0667 - 3.83
10) 132 - CVE-1999-0667 - 3.83
```

```
11) 134 - CVE-1999-0667 - 3.83
12) 140 - CVE-1999-0667 - 3.83
13) 151 - CVE-1999-0667 - 3.83
14) 153 - CVE-1999-0667 - 3.83
15) 155 - CVE-1999-0667 - 3.83
16) 157 - CVE-1999-0667 - 3.83
17) 168 - CVE-1999-0667 - 3.83
18) 179 - CVE-1999-0667 - 3.83
19) 181 - CVE-1999-0667 - 3.83
20) 183 - CVE-1999-0667 - 3.83
21) 1 - CVE-2022-25304 - 0.54
Enter the number of the vulnerability to remove(x to exit):
```

As we can see, the vulnerability on the node with ID 12, which was directly affected by the previously removed vulnerability, has seen its risk score decrease from 67.56 to 63.73. This demonstrates the cascading effect that removing a vulnerability can have on other connected vulnerabilities. Similarly, we observe that the risk scores for vulnerabilities on nodes 21 and 28 have also changed accordingly. The removal of a vulnerability can impact the overall risk assessment of related vulnerabilities, leading to a recalibration of their risk scores based on the updated network conditions. This is crucial for understanding how vulnerabilities interact within a network and emphasizes the importance of continuously monitoring and updating risk assessments as vulnerabilities are added or removed. If we decide to remove another vulnerability, for example, let's say

```
1) 12 - CVE-2015-0991 - 63.73
```

by inserting the number 1, the resulting output will reflect the updated vulnerability ranking, now without CVE-2015-0991, and the recalculated risk factors will likely show changes across other vulnerabilities due to the interconnected nature of nodes and their vulnerabilities. The new output might look like this:

```
Vulnerabilities sorted by risk score
1) 37 - CVE-1999-0667 - 10.0
2) 21 - accessControlBypass - 1.34
3) 28 - accessControlBypass - 1.34
Enter the number of the vulnerability to remove(x to exit):
```

Now, we can see that all the vulnerabilities, starting from the one present on node 12, as well as those that were directly and uniquely reachable from it, have been eliminated and are no longer accessible. Given the basic functionality of the tool, the next step will be to introduce the environmental factor into the risk calculation. This factor takes into account the specific characteristics of the network, such as asset value and centrality of the nodes, which directly influence the priority of the vulnerabilities in the ranking.

5.3 Asset value consideration

Now, after assigning different asset values to six nodes in our network, we expect these changes to influence the final ranking of vulnerabilities. Here's the breakdown of the asset scores assigned:

- Node "51_3" has an asset score of 5.
- Node "155_4" has an asset score of 4.
- Node "1_1" has an asset score of 3.
- Node "80_3" has an asset score of 2.
- Node "183_3" has an asset score of 4.
- Node "95_4" has an asset score of 3.

The higher the asset score, the more important the node is for the network. As a result, vulnerabilities on nodes with higher asset values will receive a higher risk score, moving up in the ranking, while those on less critical nodes might move down. Let's see how the final ranking will change after considering these modifications and integrating the asset values into the risk calculation:

```
Vulnerabilities sorted by risk score
1) 12 - CVE-2015-0991 - 67.56
2) 21 - accessControlBypass - 27.19
3) 28 - accessControlBypass - 27.19
4) 51 - CVE-1999-0667 - 19.14
5) 155 - CVE-1999-0667 - 15.31
```

```
6) 183 - CVE-1999-0667 - 15.31
7) 95 - CVE-1999-0667 - 11.49
8) 37 - CVE-1999-0667 - 10.0
9) 80 - CVE-1999-0667 - 7.66
10) 60 - CVE-1999-0667 - 3.83
11) 106 - CVE-1999-0667 - 3.83
12) 121 - CVE-1999-0667 - 3.83
13) 132 - CVE-1999-0667 - 3.83
14) 134 - CVE-1999-0667 - 3.83
15) 140 - CVE-1999-0667 - 3.83
16) 151 - CVE-1999-0667 - 3.83
17) 153 - CVE-1999-0667 - 3.83
18) 157 - CVE-1999-0667 - 3.83
19) 168 - CVE-1999-0667 - 3.83
20) 179 - CVE-1999-0667 - 3.83
21) 181 - CVE-1999-0667 - 3.83
22) 1 - CVE-2022-25304 - 1.62
Enter the number of the vulnerability to remove(x to exit):
```

By comparing the updated ranking with the one obtained using only the base values, we can observe significant differences that will influence our approach in the mitigation phase based on these results. Now, vulnerabilities present on nodes with a higher asset value are prioritized over others. This prioritization is crucial for effective risk management, as it directs our focus and resources toward the most critical areas of the network that require enhanced protection.

The implications of this analysis are twofold.

- **Targeted Mitigation:** By identifying vulnerabilities on high-value assets, we can implement more targeted and effective mitigation strategies. This may involve applying patches, enhancing monitoring, or even redesigning parts of the network architecture to minimize exposure.

- **Resource Allocation:** Understanding which vulnerabilities pose the highest risk allows for better allocation of resources. Organizations can focus their efforts on addressing the vulnerabilities that, if exploited, could lead to the most significant

impact on their operations.

In conclusion, the incorporation of asset value into the risk assessment process not only provides a more nuanced understanding of vulnerabilities but also facilitates informed decision-making regarding risk mitigation strategies.

5.4 Node centrality consideration

In this section, we will analyze the ranking by considering the centrality of each node within the network:

```
Vulnerabilities sorted by risk score
1) 12 - CVE-2015-0991 - 61.42
2) 37 - CVE-1999-0667 - 10.0
3) 21 - accessControlBypass - 1.24
4) 28 - accessControlBypass - 1.24
5) 51 - CVE-1999-0667 - 0.18
6) 60 - CVE-1999-0667 - 0.18
7) 80 - CVE-1999-0667 - 0.18
8) 95 - CVE-1999-0667 - 0.18
9) 106 - CVE-1999-0667 - 0.18
10) 121 - CVE-1999-0667 - 0.18
11) 132 - CVE-1999-0667 - 0.18
12) 134 - CVE-1999-0667 - 0.18
13) 140 - CVE-1999-0667 - 0.18
14) 151 - CVE-1999-0667 - 0.18
15) 153 - CVE-1999-0667 - 0.18
16) 155 - CVE-1999-0667 - 0.18
17) 157 - CVE-1999-0667 - 0.18
18) 168 - CVE-1999-0667 - 0.18
19) 179 - CVE-1999-0667 - 0.18
20) 181 - CVE-1999-0667 - 0.18
21) 183 - CVE-1999-0667 - 0.18
22) 1 - CVE-2022-25304 - 0.03
Enter the number of the vulnerability to remove(x to exit):
```

Central nodes often serve as critical points in the network, meaning that vulnerabilities associated with them can have a broader impact. A vulnerability in a central node may allow for greater access to other parts of the network compared to a vulnerability in a leaf node. Since we are focusing solely on pure centrality, the calculated factor yields a value between 0 and 1. In our case, with a relatively small network featuring a single central node and many leaf nodes, we can observe a significant delta in the associated risk values. This disparity underscores the importance of centrality in assessing vulnerabilities. However, it is crucial to note that the final tool will not rely solely on this single factor. Instead, it will integrate multiple factors, which will then be normalized and averaged to provide a more comprehensive assessment. The results we obtain here serve primarily to illustrate how the ranking can change when considering various intrinsic characteristics of the network.

5.5 Results with combined metrics

Once we have analyzed the various factors individually, we will combine them using the same values applied previously. To achieve this, I have constructed an environmental factor using the following formula:

$$\text{Env} = \text{assetScore} + \left(\text{assetScore} \times \frac{\text{nrIngoingLinks} + \text{nrOutgoingLinks}}{\text{totalNumberOfNodes}}\right)$$

where:

- **Asset Score (`assetScore`):** this represents the importance or value of a specific node in the network. Nodes with higher asset scores are considered more critical and therefore pose a higher risk if compromised.
- **Number of Ingoing Links (`nrIngoingLinks`):** this is the count of connections that lead into the node from other nodes in the network. It indicates how many other nodes are connected to this node, potentially increasing its importance.
- **Number of Outgoing Links (`nrOutgoingLinks`):** this is the count of connections that lead out from the node to other nodes. It shows how many pathways can be exploited from this node.
- **Total Number of Nodes (`totalNumberOfNodes`):** this is the total count of nodes in the entire network. It serves as a normalizing factor to assess the node's connectivity relative to the entire network size.

- **Centrality:** the term

$$\text{Centrality} = \frac{\text{numberIngoingLinks} + \text{numberOutgoingLinks}}{\text{totalNumberOfNodes}}$$

  represents the centrality of the node, combining both incoming and outgoing connections. This provides an indication of the node's connectivity and influence within the network. A node with higher centrality is likely more significant because it has multiple paths connecting it to other nodes.

Here is the final ranking of vulnerabilities after considering both the asset value and the centrality of the nodes:

```
Vulnerabilities sorted by risk score
1) 12 - CVE-2015-0991 - 128.97
2) 21 - accessControlBypass - 28.43
3) 28 - accessControlBypass - 28.43
4) 51 - CVE-1999-0667 - 20.01
5) 155 - CVE-1999-0667 - 16.01
6) 183 - CVE-1999-0667 - 16.01
7) 95 - CVE-1999-0667 - 12.01
8) 37 - CVE-1999-0667 - 10.0
9) 80 - CVE-1999-0667 - 8.01
10) 60 - CVE-1999-0667 - 4.01
11) 106 - CVE-1999-0667 - 4.01
12) 121 - CVE-1999-0667 - 4.01
13) 132 - CVE-1999-0667 - 4.01
14) 134 - CVE-1999-0667 - 4.01
15) 140 - CVE-1999-0667 - 4.01
16) 151 - CVE-1999-0667 - 4.01
17) 153 - CVE-1999-0667 - 4.01
18) 157 - CVE-1999-0667 - 4.01
19) 168 - CVE-1999-0667 - 4.01
20) 179 - CVE-1999-0667 - 4.01
21) 181 - CVE-1999-0667 - 4.01
22) 1 - CVE-2022-25304 - 1.69
Enter the number of the vulnerability to remove(x to exit):
```

This chapter illustrates how the inclusion of environmental factors transforms the basic risk calculation process, allowing a more context-sensitive prioritization of vulnerabilities. With the potential to integrate additional metrics in future iterations, the tool can become a more comprehensive solution for network security analysis.

Chapter 6

Conclusions

The work carried out in this thesis has provided a comprehensive analysis of cyber risk management in industrial systems, demonstrating the importance of a proactive and continuous approach. By using automated reasoning techniques and attack graph representations, it has been possible not only to identify existing vulnerabilities but also to predict possible attack paths, thus improving the overall resilience of industrial networks.

A key achievement of this research is developing a continuous monitoring tool that allows real-time network risk assessment. Although still in its preliminary phase, the tool shows significant potential for future applications. It could be particularly valuable in critical infrastructure security and automated vulnerability management, where timely detection and response to threats are paramount. As the tool evolves, it could become an essential resource for real-time cyberattack detection and prevention.

The results also highlight how the convergence of IT and OT introduces new security challenges while simultaneously offering opportunities to strengthen defenses. Integrating traditional technologies with advanced risk assessment tools can enhance protection against emerging threats, ensuring greater operational continuity and reducing potential damage.

In conclusion, this thesis presents a balanced assessment of the developed tool's current capabilities and potential future impact. While still in its early stages, the tool lays a solid foundation for future developments, including the improvement of misconfiguration descriptions to allow a better classification, expanding the available database, or incorporating advanced metrics and further optimizations in risk analysis processes. These improvements would enhance the ability to prevent cyberattacks in critical environments, solidifying the tool's role as a proactive defense mechanism in industrial cybersecurity.

Bibliography

[1] *7 Reasons to Make Continuous Risk Assessment a Standard Practice - Abel Solutions*. Section: Abel Insights. June 27, 2022. URL: https://www.abelsolutions.com/7-reasons-to-make-continuous-risk-assessment-a-standard-practice/.

[2] *My Top 5 Reasons for Conducting Ongoing, Continuous Risk Analyses*. URL: https://405d.hhs.gov/post/detail/28ae7294-a5c6-4bb9-ac81-9757a4b33cd0.

[3] Joseph McCafferty. *The Power of Rolling Risk Assessments*. Internal Audit 360. June 8, 2023. URL: https://internalaudit360.com/the-power-of-rolling-risk-assessments/.

[4] *What is OT Security? An Operational Technology Security Primer*. Fortinet. URL: https://www.fortinet.com/solutions/industries/scada-industrial-control-systems/what-is-ot-security.

[5] Keith Stouffer et al. *Guide to Operational Technology (OT) security*. NIST SP 800-82r3. Gaithersburg, MD: National Institute of Standards and Technology (U.S.), Sept. 28, 2023, NIST SP 800–82r3. DOI: 10.6028/NIST.SP.800-82r3. URL: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r3.pdf.

[6] *How Is OT Different From IT? OT vs. IT*. Cisco. URL: https://www.cisco.com/c/en/us/solutions/internet-of-things/what-is-ot-vs-it.html.

[7] *Dozens of insecure-by-design flaws found in OT products | CSO Online*. URL: https://www.csoonline.com/article/573035/dozens-of-insecure-by-design-flaws-found-in-ot-products.html.

[8] *OT:ICEFALL*. Forescout. URL: https://www.forescout.com/research-labs/ot-icefall/.

[9] *What is a SCADA System and How Does It Work?* URL: https://www.onlogic.com/blog/what-is-a-scada-system-and-how-does-it-work/.

[10] PECB. *What Is SCADA and How Does It Work?* URL: https://pecb.com/article/what-is-scada-and-how-does-it-work.

[11] *Intelligent SCADA solutions.* SCADA International. URL: https://scada-international.com/.

[12] *PLC vulnerabilities can enable deep lateral movement inside OT networks.* CSO Online. URL: https://www.csoonline.com/article/574547/plc-vulnerabilities-can-enable-deep-lateral-movement-inside-ot-networks.html.

[13] admin. *Alla scoperta del P.L.C.* Foral Formazione. Nov. 9, 2018. URL: https://www.foral.org/2018/11/09/alla-scoperta-del-p-l-c/.

[14] *Forescout – Manage cyber risk and mitigate threats, continuously.* Forescout. URL: https://www.forescout.com/.

[15] *What Is ICS (Industrial Control System) Security?* Fortinet. URL: https://www.fortinet.com/it/resources/cyberglossary/ics-security.html.

[16] *Industrial Control System - Definition | Trend Micro (US).* URL: https://www.trendmicro.com/vinfo/us/security/definition/industrial-control-system.

[17] *What is Industrial Control Systems Security?* Section: Manufacturing & Industrial. June 7, 2023. URL: https://internationalsecurityjournal.com/industrial-control-systems/.

[18] *What is a control loop? | Watlow.* URL: https://www.watlow.com/blog/posts/setting-up-a-control-loop.

[19] *HMI: Human-Machine Interface.* Inductive Automation. URL: http://inductiveautomation.com/resources/article/what-is-hmi.

[20] *What Is Intelligent Electronic Device? A Complete Guide 2023.* Section: Devices. Apr. 18, 2023. URL: https://techjournal.org/what-is-intelligent-electronic-device.

[21] *What Are Data Historians and Do You Need One?* URL: https://www.clarify.io/learn/data-historian.

[22] *DNP3 (Distributed Network Protocol) e IEC 61850.* URL: https://www.copadata.com/it/industrie/energia-infrastrutture/energy-insights/dnp3-e-iec61850/.

[23] *Cos'è un Modbus e come funziona? | Schneider Electric Italy.* www.se.com. June 22, 2023. URL: https://www.se.com/it/it/faqs/FAQ000258445/.

[24] *Protocollo Modbus TCP: protocollo di comunicazione universale*. Section: Guide. Jan. 12, 2021. URL: https://moxa.distry.shop/protocollo-modbus-tcp-tutto-quello-che-ce-da-sapere/.

[25] *What is OPC?* OPC Foundation. URL: https://opcfoundation.org/about/what-is-opc/.

[26] *What is the BACnet Protocol and How is it Used in Building Automation Systems to Control Data Exchange? - Technical Articles*. URL: https://control.com/technical-articles/what-is-the-bacnet-protocol/.

[27] *Common Industrial Protocol (CIP™) | ODVA Technologies*. ODVA. URL: https://www.odva.org/technology-standards/key-technologies/common-industrial-protocol-cip/.

[28] *What is the EtherCAT Communication Protocol - acontis technologies*. URL: https://www.acontis.com/en/what-is-ethercat-communication-protocol.html.

[29] *Why Do Attackers Target Industrial Control Systems? | Trend Micro (US)*. URL: https://www.trendmicro.com/vinfo/us/security/news/cyber-attacks/why-do-attackers-target-industrial-control-systems.

[30] *Defensive Strategies for Industrial Control Systems | Trend Micro (US)*. URL: https://www.trendmicro.com/vinfo/us/security/news/cyber-attacks/defensive-strategies-for-industrial-control-systems.

[31] *Network security assessment using a semantic reasoning and graph based approach - ScienceDirect*. URL: https://www.sciencedirect.com/science/article/abs/pii/S0045790617302409.

[32] Ian Roberts. *Contextual Analytics: The Reason Behind Reasoning*. CyberTheory. June 5, 2022. URL: https://cybertheory.io/contextual-analytics-the-reason-behind-reasoning/.

[33] *CVSS v3.0 Specification Document*. FIRST — Forum of Incident Response and Security Teams. URL: https://www.first.org/cvss/v3.0/specification-document.

[34] *CWE - Common Weakness Enumeration*. URL: https://cwe.mitre.org/index.html.

[35] *Practical Risk Management: The 5 "W's" for the Win!* MBP. Oct. 14, 2020. URL: https://www.mbpce.com/risk-management/practical-risk-management-the-5-ws-for-the-win/.

[36] *Flaws in industrial wireless IoT solutions can give attackers deep access into OT networks | CSO Online*. URL: https://www.csoonline.com/article/574531/flaws-in-industrial-wireless-iot-solutions-can-give-attackers-deep-access-into-ot-networks.html.

[37] *What is a remote terminal unit (RTU)? | Definition from TechTarget*. WhatIs. URL: https://www.techtarget.com/whatis/definition/remote-terminal-unit.

[38] *What are Human-machine interfaces (HMI)?* PubNub. URL: https://www.pubnub.com/learn/glossary/human-machine-interface/.

[39] *What is Remote Monitoring and Diagnostics?* Click Maint CMMS. URL: https://www.clickmaint.com/glossary/remote-monitoring-and-diagnostics.

[40] *Master Terminal Units MTU in SCADA systems ?* URL: https://engineeringslab.com/all_interview_questions/master-terminal-units-mtu-in-scada-systems--3635.htm.

[41] *ICS: Cos'è l'Industrial Control System?* UNILAB - Heat Transfer Software. June 11, 2021. URL: https://www.unilab.eu/it/articoli/coffee-break-it/ics/.

[42] *Cos'è PROFIBUS DP?* Procentec. URL: https://procentec.it/contenuto/cos%25C3%25A8-profibus-dp/.

[43] *Cos'è PROFIBUS PA?* Procentec. URL: https://procentec.it/contenuto/cos%25C3%25A8-profibus-pa/.

[44] *MITRE ATT&CK®*. URL: https://attack.mitre.org/.

[45] *Sistemi di controllo industriale e SCADA*. Check Point Software. URL: https://www.checkpoint.com/it/quantum/next-generation-firewall/industrial-control-systems-appliances/.