Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

Sander Løken Berntsen

Industrial Control System Threat Intelligence, Process Safety and Security Oriented Ontology

Merging threat intelligence, security and safety.

Master's thesis in Master in Information Security
Supervisor: Vasileios Gkioulos
Co-supervisor: Stephen Dirk Bjørn Wolthusen
June 2022

NTNU
Norwegian University of
Science and Technology

Sander Løken Berntsen

Industrial Control System Threat Intelligence, Process Safety and Security Oriented Ontology

Merging threat intelligence, security and safety.

Master’s thesis in Master in Information Security
Supervisor: Vasileios Gkioulos
Co-supervisor: Stephen Dirk Bjørn Wolthusen
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

Industrial Control System Threat Intelligence, Process Safety and Security Oriented Ontology

Sander Løken Berntsen

Abstract

Protecting industrial control systems is a complex task where failure can result in the loss of life. This thesis takes a deep dive into ICS safety and security literature to determine which elements allow a threat actor to sabotage these systems, what system conditions enable the sabotage, and which actions taken by a threat actor will most detriment to the system safety. Threat intelligence is also incorporated into the ontology to help determine the identity of a threat actor. The author utilizes this knowledge to enable a novel ontological approach for identifying safety layers at risk, the underlying factors that determine the risk state, and how one can use data points to identify the threat actor behind the sabotage. The author tests the ontology on two real-world scenarios where it demonstrates the ability to perform ICS risk element identification without definitive threat actor identification.

Contents

Figures

Tables

Acronyms

DCS distributed control system. 1, 8, 35, 53

ICS industrial control system. iii, ix, 2–4, 7, 15, 19, 20, 25, 27–30, 35, 37–39, 48, 55, 59, 60

IT Information Technology. 4, 28, 29, 43, 47

OT Operation Technology. 4, 25, 29, 43, 47, 56

PLC Programmable Logic Controller. 53

SCADA Supervisory Control And Data Acquisition. 1, 7, 15, 35, 53

SIS Safety Instrumented System. ix, 3, 7, 27–30, 40, 41, 43, 50, 51

Chapter 1

Introduction

1.1 Industrial Control System Security

High stakes and immediate consequences are two words that describe working with information security in the context of industrial control systems. When it comes to incident handling, the need for a standard, precise, and descriptive ontology is paramount to prevent correlated incidents from getting out of control. We define ontology as "the basic terms and relations comprising the vocabulary of a topic area and the rules for combining terms and relations to define extensions to the vocabulary." Currently, there is no objective standard for incidents in the cyber security domain, let alone the niche field of ICS security. This is a typical issue in a new and rapidly developing field. Everyone wants to coin their terms, but this can lead to communication issues, especially when two or more cooperating organizations with no agreed-upon ontology try to work together to solve a problem with high-risk potential.

Concretely: let us use the construction business and the I-beam as an example. Everyone in the construction industry knows the shape of the beam, what materials it is likely to be made from, what probable dimensions in the context of the specific project, and other parts and materials that can be used in combination with the I-beam. In other words, the concept of an I-beam and all its related properties is effectively communicated essentially in just one word.

This efficiency is also ideal in incident response scenarios. Any of these scenarios in the power delivery industry will almost definitely require multiple individuals having to communicate relatively complex ideas efficiently to increase response efficiency and reduce scenario consequences.

Currently, there exist multiple ontological structures for incident response in typical network structures such as CSIHO and the Incident Management Ontology. These methodologies are suitable for network structures that contain many hosts with fancy logging and other consolidatable security features. The nature of ICS systems such as SCADA and DCS does not allow for that level of processing and network overhead as this may impact the speed and availability of the network. There is also an increasing amount of IoT devices connected to these systems as

IP is becoming the de facto standard of network communication.

1.2 Keywords

Keywords for this thesis are[1]: Control System Security, Computer Security, Cyber Warfare, Semantic Web, and Inference mechanisms.

1.3 Problem Description and Motivation

Industrial control systems are increasingly becoming a complex combination of information and operational technology devices working synergistically to control a physical process that outputs real-world value[2]. Traditionally, industrial control systems have mainly consisted of unique built subsystems working in isolation to monitor and manipulate a single process. This old-school style of ICSs is slowly becoming obsolete. Now, centralized and data-driven control systems are favorable as they require fewer human personnel to monitor and operate the system. Preferring centralized control and monitoring of large industrial control system fields now leads the charge for replacing standard ICS components with smarter, Ethernet-connected IoT devices, increasing the complexity. This increase in complexity can make it difficult to fully grasp any industrial control system's nuances, as the variance and quantity of devices and running software keeps increasing over time.

Each of the different components of an industrial control system is often placed in the system with the intent of performing one and only one specific function. This functional specialty means that these components, if put out of play from either sabotage or natural causes, can affect the security and safety of the ICS in different ways. Some of these components will only affect the value of the process by conceivably hurting the product if they fail, causing a monetary cost. However, other components are put in place specifically to prevent a tragedy where the cost will be counted in human lives.

By incorporating a substantial amount of IoT devices to perform the different operations of the industrial control system, the system owners have exposed themselves to an ever-increasing cyber-attack surface[3]. For any Internet of Things device to function correctly, there is a need for an underlying set of functions that the device must be able to perform. This effect is two-sided.

On the one hand, this makes it possible for the device to be more flexible in the performance of its auxiliary functions, such as transmitting data to the central control units, and adaptations can be made on devices in production without needing any physical interaction. It can also make it easier for different devices to intercommunicate as they now all use the same communication protocol, making proprietary devices less common.

On the other hand, this opens for more flexibility for threat actors as well[4]. The more standardized nature of these new ICS devices makes it easier for attackers to perform cyber operations as they now can have more networked devices to affect and use to amplify attacks. The increase in ethernet connectivity also allows systems that used to be isolated from the network to be compromised. Control and engineering systems such as engineering stations or operation stations are often run on a version of a commercially available operating system that can be vulnerable to the same attacks as other devices running on the same operating system. In combination with the required connectivity of these ICSs, this vulnerability allows attackers to attack devices responsible for operator intervention and monitoring and devices that serve a critical role in enabling communication across the system. If a device like this, such as a serial/ethernet converter, were to have its firmware rewritten during an attack. It would require manual intervention to work again. This intervention would cause harm in the attack and seriously disrupt the recovery work required to normalize the ICS.

In recent years there has been an increase in the number of attacks on industrial control systems[5]. Advanced Persistent Threats (APTs), such as state-sponsored threat actors and professional cybercriminals, have since 2015 noticed the value of gaining persistent access to an industrial control system performing an important function and either waiting for an opportune moment to strike or hold the entire system hostage with ransomware. Attacks such as these are rarely intentionally hazardous as they historically have focused on obstructing the system's function, not weaponizing the physical process by removing important safety layers designed to prevent harm to human life.

This attack pattern changed in 2017 when it came to light that an unknown threat actor had infiltrated a Saudi Arabian petrochemical plant and created malicious firmware designed to disable the Safety Instrumented System (SIS) in one of the ICS fields. If the attack in question were to be completed, three safety layers designed to prevent an incident from having consequences outside the ICS would not kick in. This would allow the threat actor to cause maximum real-world damage, potentially ending lives.

Generally, Intrusion Detection/Prevention Systems (IDS/IPS) focus on signature-based detection. These attacks do not lend themselves to traditional intrusion detection and Security Information and Event Management (SIEM). These attacks are dependent on certain preconditions, and the actual effect of the attack is not visible. The industrial control system cyber security space seems to be missing a technique that would take the system in question, analyze the state of its components, and infer the system's risks and why the system is facing those risks. Defenders of industrial control systems also seem to be missing a technique that would help them understand the attack's goal, giving an increased chance of performing actions that would partially or entirely prevent the threat actor from achieving their operational goal.

This thesis aims to find the relevant literature and apply it to provide an experimental solution to these problems.

1.4 Research Questions

Based on these existing problems in the Industrial Control System Security section there are four research questions that this thesis aims to answer.

1. How can ontologies be used in order to model Industrial Control Systems?
2. How can ontology be used to identify system preconditions for attacks against ICS?
3. Can ontologies further infer system states based on discovered attack preconditions?
4. How can threat intelligence be used to enhance the effectiveness of an attack precondition ontology?
5. What is the benefit of a system state-oriented ontology, compared to a network-oriented ontology?

1.5 Contributions

The main contribution of this thesis is to provide research about the possibilities of utilizing ontologies to improve the defensive work on industrial control systems and to argue the importance of understanding the value of looking at a system as a complete risk element comprised of more minor potential risk elements. The theory behind what makes an ICS vulnerable to IT and OT attacks is detailed. This paper provides a novel insight into detailing high- and low-level preconditions required for these attacks and an experimental approach to discovering these preconditions. This approach comes in the form of an experimental ontology that can model an industrial control system, infer some vulnerabilities based on the system, determine the total process risk, and finally shows how one can incorporate threat intelligence to enhance the effect of the ontological model. Finally, the thesis describes a way to test ontologies such as these by modeling two real-world scenarios based on publically available information. Creating and testing an ontology in this way will provide an excellent theoretical and practical base for someone to develop methods using ontologies and system state-based security solutions.

1.6 Thesis Outline

This thesis follows the introduction, method, results, and discussion structure (IMRaD). After this chapter, the introduction, chapter two provides theoretical insight into how ontologies work and a detailed description of how the logic in the ontology is denoted and how inference is performed. Chapter three details other scientific works that can relate to this thesis and demonstrate the novelty of this thesis. Chapter four describes the methods used to get the results from the upcoming literature review, ontology, and reference scenario chapters. Chapter five shows the results of the extensive literature review and how this control system security theory can be turned into a functional approach to create an ontology. Chapter six shows how the theory from chapter five is applied in practice to the ontology and how the ontology infers information from the model. Chapter seven uses the ontology described in chapter six in two reference scenarios and shows how these scenarios can be modeled and how the ontology would be able to detect the several risk elements that made these attacks possible. Chapter eight is the discussion where the author discusses the results and methodology, giving insight into how nuances surrounding the nature of some sources influenced the result. Finally, chapter nine summarizes the findings, concludes the thesis, and lists future work that can be done with the information in this thesis. After this chapter comes the bibliography and appendixes.

Chapter 2

Technical Background

This chapter describes the different necessary technical knowledge bases required to understand the remaining chapters of this thesis. First, a few essential concepts are explained. Then some important background on description logic and the notation will be used throughout this thesis, along with the theoretical knowledge laying the foundation on how ontologies work. Lastly, ontologies are explained in detail, along with the different descriptions and relation characteristics used in this thesis.

2.1 Definitions

Industrial Control System

"*Industrial control system (ICS) is a collective term used to describe different types of control systems and associated instrumentation, which include the devices, systems, networks, and controls used to operate and/or automate industrial processes.*"[6]

Safety Instrumented System

"*A Safety Instrumented System (SIS) is composed of sensors, logic solvers, and final control elements for the purpose of taking a process to a safe state when predetermined conditions are violated. The function of the SIS is to monitor the process for potentially dangerous conditions (process demands) and to take action when needed to protect the process.*"[7]

Supervisory Control and Data Acquisition (SCADA)

"*SCADA (supervisory control and data acquisition) is a category of software applications for controlling industrial processes, which is the gathering of data in real time from remote locations in order to control equipment and conditions.*"[8]

Distributed Control System (DCS)

"*As the name implies, the DCS is a system of sensors, controllers, and associated computers that are distributed throughout a plant. Each of these elements serves a unique purpose such as data acquisition, process control, as well as data storage and graphical display. These individual elements communicate with a centralized computer through the plant's local area network – often referred to as a control network*"[9]

Threat Intelligence

"*Threat intelligence is the process of identifying and analysing cyber threats. The term 'threat intelligence' can refer to the data collected on a potential threat or the process of gathering, processing and analysing that data to better understand threats.*"[10]

2.2 Description Logics

Description logics (DL) is an umbrella term for a branch of knowledge representation languages, and make up the theoretical basis of ontologies. Generally, DL is great for semantic modeling as it has a high degree of decidability and is efficient at performing decisions while having a reasonably high expressive power and reasoning complexity[11]. We use Description Logics instead of First Order Logics for this paper due to the overly complex nature of first-order logic. DL achieves efficiency by using simple descriptions to create more complex ones using constructors.

2.2.1 Notation

The following table (2.1) describes the notation used to describe different concepts, characteristics and relations in this thesis.

$\top$	Special concept with every individual as an instance
$\bot$	Empty set concept
$\sqcup$	Conjunction of concepts
$\sqcap$	Disjunction of concepts
$\neg$	Negation
$\forall$	Universal restriction / given all
$\exists$	There exists / existential restriction
$\sqsubseteq$	Concept inclusion / subset of
$\equiv$	Concept equivalence / equivalence
$\doteq$	Concept definition
:	Concept assertion / $x : C$ x is a C
:	Role assertion / $(x, y) : R$ x is R-related to y

Table 2.1: Description Logics notation table

2.2.2 ABoxes and TBoxes

Two essential concepts in the Description Logics domain are the TBox(Terminological box) and the ABox(assertional box). The distinction between these two concepts comes from their function. A TBox contains sentences describing the different concepts(classes) relations. The ABox contains the ground sentences describing where in the hierarchy the individual belongs. In OWL, this distinction is described as objects(TBox) and individuals(ABox).

Examples of these concepts are: Every man is a human (TBox), and John is a man (ABox).

2.3 Ontologies

"*[...] an ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. The representational primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members).*"[12]

Ontologies are a way of formally describing the knowledge structure of a domain. In computer science, we use ontology to describe a specific concept within a domain, the data properties of that concept, and how this concept relates to other concepts with their data properties and relations in the same domain. This way, we can capture knowledge structured for a human brain and put it into a computer-friendly model that can answer complex questions within the specific domain.

The real benefit that ontologies have is that they can find relations across different forms of data representation (text documents, databases, spreadsheets, etc.) by capturing the data in a structured manner that is universal across the different platforms.

Ontologies consist of two main components: Classes and relationships.

2.3.1 Classes

Classes are the representation of real-world concepts. A class can be any concept, and it can be a part of a superclass, have a set of subclasses, or most likely a combination of both.

Each class has a class description that allows for a logical description of the class. The class description is comprised of eight sections:

EquivalentTo

This section shows what combination of class expressions is equated with fitting in this class. An example of this can be an apple with red color, which belongs to the class "Red Apple."

$$redApple \equiv (color.red \sqcap fruit.apple) \tag{2.1}$$

SubClassOf

This section describes what superclass this class is a subclass of. An example of this would be that an apple is a subclass of the superclass fruit.

$$\exists apple \sqsubseteq fruit \tag{2.2}$$

General Class Axioms

General class axioms function better to allow the expression of both necessary and sufficient conditions.

Instances

Here the individuals that belong to this class are displayed. Individuals are factual instances of a class. If you have the class apple, then an instance of the class apple would be an apple that you physically have in your hand.

$$instance \sqsubseteq (class_1 \sqcap ... \sqcap class_n) \tag{2.3}$$

DisjointWith

DisjointWith is a way of saying that two or more specific classes are not the same. Classes are generally considered disjoint by default unless otherwise specified or inferred. This is trivial as for natural language. Almost all nouns are disjoint from each other, with a few exceptions called synonyms. An apple is generally not in the same class as a car.

$$apple \equiv \neg car \tag{2.4}$$

Target for Key

The target for key is a way to determine key properties that are used to separate different individuals of the same class or classes. These can be data properties such as an identification number or an object property.

$$\exists keys \sqsubseteq individual \tag{2.5}$$

DisjointUnionOf

DisjointUnionOf specifies that the specific class in question is the main class in a disjoint union class axiom.

2.3.2 Relationship

Further in this thesis, relationships are referred to as object properties. This is because the primary tool used to create the ontology uses this terminology. These object properties describe how different classes and individuals interact with and relate to each other. Like classes, these object properties have descriptors, but they also have characteristics.

Description: Equivalent To

This works in the same way as for classes. The described property is equivalent to the properties listed in this descriptor.

$$generatesPower \sqsubseteq connectedToBattery \sqcap receivesEnergy \tag{2.6}$$

Description: SubProperty Of

When a property is described as a sub-property of another property, one can think of it intuitively as it implies another relationship. If a threat actor performs a DDoS against a system, it is also natural to imply that the threat actor also attacks the system in question. This makes DDoS a sub-property of Attacks.

$$\exists sabotages \sqsubseteq attacks \tag{2.7}$$

Description: Inverse Of

Inverse of is a way to declare that the selected property is an inverse of one or more properties. An example of this is that "brotherOf" can be an inverse property to "hasBrother" describing the relationship between two siblings where at least one is a male.

Description: Domains (Intersection)

This descriptor declares that the class that has this relation to another class must be in the declared class expression. In the case of parentHasSon the domain of this property would be a parent, as in order to have this relation, one needs to be a parent implicitly.

Description: Ranges (Intersection)

On the other hand, we have ranges which is more or less the opposite of domains. The class that has this relation enacted upon itself has to belong to the class expression of the range. If we look at the previous example where parentHasSon, the class receiving this property must be a son.

Description: Disjont With

This works more or less the same way as with the disjoint with of the classes. It is, however, slightly more helpful in this case as object properties do not have an implicit disjointedness. In many cases, it is essential to declare that two relations can not exist between two classes in the same direction. Things like hasParent and hasChild can not be the same as it is impossible to have your child as a parent as these two concepts can not exist simultaneously.

Description: SuperProperty Of Chain

Super properties of Chain are used to declare that a property is implied by a chain of other properties.

Characteristic: Functional

Functional properties can have at most one outgoing relationship per individual. Intuitively this means that only one of these relationships can exist per individual in the range of the property. An example of this can be that a child only has one biological father, making hasBoilogicalFather functional, while the child can have two biological parents, making hasBiologicalParent not functional.

There also exists an inverse functional object characteristic. This means that the inverse of this relationship can only have at most one individual.

Both of these characteristics can be active simultaneously, making the relationship a one-to-one relationship.

Characteristic: Transitive

The transitive characteristic of the object property means that if individual X sends to and/or receives data from an individual Y that does the same to individual Z, then X is indirectly connected to Z.

$$((x,y):R \sqcap (y,z):R) \equiv (x,z):R \tag{2.8}$$

Characteristic: Symmetric

Symmetric object properties are properties that are their inverse. Concretely this means that if a property hasSibling is symmetric, this means that if X hasSibling Y, then Y hasSibling X.

$$(x,y):R \equiv (y,x):R \tag{2.9}$$

It is also possible to declare that a property is asymmetric, making it impossible for two individuals to have the same relation to each other. This could be that X hasChild Y, where it is impossible for Y to both be X's child and have X as a child.

$$(x,y):R \equiv \neg(y,x):R \tag{2.10}$$

Characteristic: Reflexive

Reflexive relations mean that the individual has a relation to itself along the property.

$$(x, x) : R \tag{2.11}$$

Irreflexive relations must be applied from one individual to another.

$$\neg(x, x) : R \tag{2.12}$$

2.3.3 Triples

Ontologies use these combinations of classes and relationships to create triples. Semantic triples consist of three elements making up some statement. In the context of ontologies, triples consist of a subject, a predicate, and an object. The subject and the object are two ontological classes, and the predicate is an object property. Example: Computer(Subject) connects to(predicate) router(object).

$$(x, y) : R \equiv class(x) \rightarrow realtion(R) \rightarrow class(y) \tag{2.13}$$

All the triple elements are already described and characterized in the ontology before the triple is created.

2.3.4 Inference

Semantic Web inference is the practice of finding new relationships between different resources. In this context, the inference is an automatic process that creates these new relationships based on the existing data and predefined rules. The inference of ontologies follows the rules of inference[13].

These rules are applied to the defined classes and their relationships to verify ontological consistency and infer new relationships and properties that are not already stated.

Chapter 3

Related Work

This chapter examines different works related to this thesis. Firstly, it goes through works related to industrial cyber security; secondly, ontologies; and thirdly, taxonomies.

3.1 Industrial Cyber Security

There exist much work when it comes to industrial cyber security. It is a complex field with much value in circulation and many assets that need to be secure. The function that many ICSs perform can be vital to the population at large and other industries in general.

The older methodology of designing complex centralized industrial control systems, SCADA, has much work written about it[14]. These works often focus on the synergy between older legacy systems and the new, more modern data collection, communication, and system control methods. This work allows for various methodologies for detecting vulnerabilities in different ICSs, but they are strict in their form, and there are no inference mechanisms involved. This limits the methodologies' abilities to the discoveries that are already made and the expertise of the researchers. Often these methods are not easy to expand on, and they are often system design limited.

3.2 Ontologies

Some ontological models can be used to recreate an industrial control system and list the different vulnerabilities present for the individual components. These ontologies provide value as they allow defenders to have a structured overview of their component's weaknesses. However, they do little to utilize the inference and structure of the entire system to find issues that stem from the system architecture or mistakes made in configuration or operations. Many security issues in the previous section are from misconfigured firewalls or other choices that make daily operations easier. These issues are not addressed in the ontologies that the author could find.

3.2.1 Ontology Modelling of Industrial Control System Ethical Hacking

This model[15] is more or less just a slightly complicated relational database for common vulnerability enumeration.

3.2.2 Hybrid ontology for safety, security, and dependability risk assessments and Security Threat Analysis (STA) method for industrial control systems

This rather complex ontology[16] focuses primarily on describing relations between different elements of an industrial control system, how they affect safety, and it bakes in a model meant to structure security data. This last part allows for structuring an attack, either post-incident or as a pre-thought-out scenario, to give a structured view of the event.

3.3 Taxonomies

3.3.1 MIRTE ATT&CK and MITRE Engage

The MITRE ATT&CK framework is a taxonomy, or knowledge base, of tactics and techniques used by threat actors. This knowledge base is created based on observations made from real-world attacks. Attack has 14 different categories of techniques that have their sub-techniques as well. Each of the different techniques has a unique ID and metadata, procedure examples, mitigations, and ways of detecting these attacks.

3.3.2 Cyber security threat modeling based on the MITRE Enterprise ATT&CK Matrix

This paper[17] takes the information provided by Mirte ATT&CK and expands upon it by detailing system assets, attack steps, defenses, and asset associations. The threat modeling language tries to represent the information in ATT&CK in an entity-relationship model with the goal of representing information systems as a whole. This language is meant to enable attack simulations based on system models to uncover weaknesses in the system architecture.

3.4 Threat Intelligence

3.4.1 Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence

This paper[18] introduces a cyber threat intelligence model that is used to allow blue teams to increase their threat intelligence capabilities. The authors of this paper also use this model to evaluate other threat intelligence models to gauge their ability to represent important information about a threat actor.

Chapter 4

Methodology

4.1 Literature Review

A significant part of this thesis will be based on a literature review. Much work on ontologies in the Industrial Control System field needs to be reviewed. Most of the existing research papers will provide valuable insights into the ICS, power delivery, network security, and ontology fields. In addition, information from industry professionals was also taken into consideration through reviewing articles, memos, and conference material.

4.1.1 Criteria

For a piece of literature to be considered relevant for this paper, it needs to relate to one of the following fields generally: Industrial Control Systems, Power Delivery, Information Technology Security, Operational Technology Security, Ontology, or Power Distribution. Articles and books with many citations will be taken more into account. The more evidence there is of the research of a paper being used either in practice or as a basis for further research, the more it will weigh towards the conclusion of this thesis.

On the contrary, the anecdotal information provided by industry professionals will not count as much towards the conclusion of this thesis, as this information is created in situations that are not as tightly controlled as the academic work. This bias does not discount the professional experience from giving valuable insights into this subject, but there are a lot of other factors that need to be taken into consideration (See 8.5)

4.2 Ontology

Figure 4.1: Protégé logo

4.2.1 Tool Choice

Protégé is an open-source ontology tool and framework used by governments, researchers, and businesses worldwide. It was the single most mentioned tool in the literature, and other researchers have achieved good results using this tool. For this reason, Protégé is the tool of choice.

Ontology Language

This ontology will be made using Ontology Web Language (OWL), which is the standard ontology language in Protégé[19].

4.2.2 Ontology Creation Method

This ontology was made using established practices of ontology creation detailed by Noy and McGuinness [20]. They describe seven unique steps to creating an ontology from scratch.

1: Determine domain and scope

Firstly the ontology needed to be assigned a domain and scoped to a specific field within this domain. The domain chosen for this thesis was ICS security in power distribution systems, and the scope was set to focus on 132KV AC to AC transformers with supporting and security systems.

2: Consider reusing existing ontologies

Multiple different existing ontologies were taken into consideration for the problem described in 1.3. All of the different ontologies that were taken into consideration can be found in 3.2. While none of these ontologies were used in full, they still provided some useful information used in the project.

3: Enumerate important terms in the ontology

This ontology crosses over multiple technical fields with a lot of different terms specific to each field. Creating a complete list of all these terms would not be a wise use of time, so only the relevant existing terms were enumerated and used.

4: Define classes and their hierarchy

Uschold and Gruninger[21] describe three different approaches to defining classes and their hierarchy: top-down, bottom-up, and combination. The bottom-up approach made the most sense for this thesis as this ontology's scope is limited to a small and low level. The states of the individual components of the transformer will determine if it is in a dangerous state or not, which makes them the most logical place to start. On some occasions, the combination approach was used.

5: Define class slots (properties)

After all the relevant classes were defined, the different properties (or slots), both intrinsic and extrinsic, were defined in accordance with Noy and McGuinness' directions[20].

6: Define slot facets

Different facets of the slots in the ontology were defined. According to available documentation on the different sensor types and other information, the different value types were defined as the most likely type. The author acknowledges that there can be some discrepancy between individual sensor components depending on the individual transformer itself.

7: Create instances

Instances were created and tested in accordance to reference scenarios. When problems were encountered in this step, other previous relevant steps were revisited in order to modify and improve the ontology.

4.2.3 Identifying Dangerous States, Countermeasures and Attacker Influence

In order to determine what qualifies as a dangerous state, material relevant to the safety aspect of Operational Technology was reviewed. This was done by observing what process behaviors the relevant safety documentation describe as dangerous and what measures are meant to return the process to a state that is considered safe.

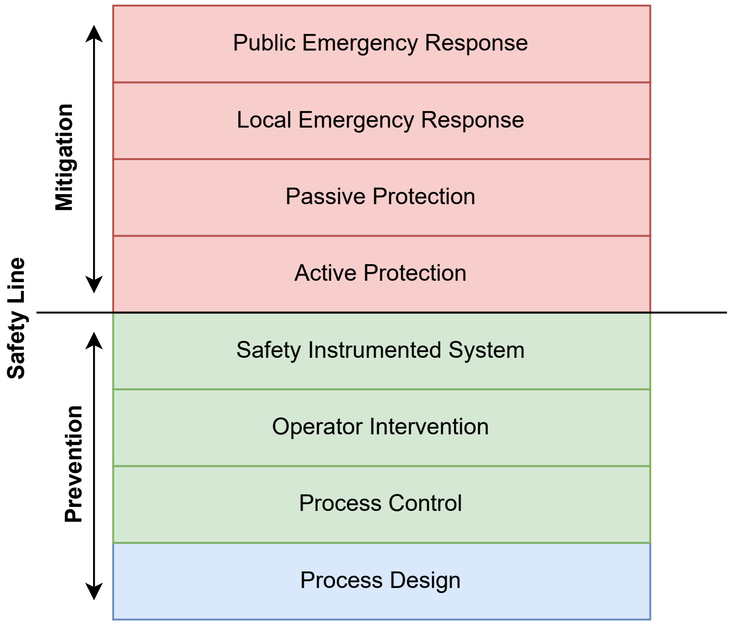


Figure 4.2: Safety Layer Protection Model

After identifying these states and their respective measures, it was necessary to understand which of these measures an adversary could influence, to what degree, and what effect this could have on the process and its safety.

4.2.4 Incorporating Threat Intelligence

In order to incorporate a threat intelligence model, we will observe the different threat intelligence models used in the literature review and determine whether to use either one of the models found or a variant. The model that we will go for is the model that most accurately reflects the information acquired in the literature review and that would be reasonably feasible to implement in the time allotted for this thesis. This model should make it possible to collectively look at a threat actor's different actions and other indicators in an attack and best attribute it to identity. These indicators should vary in granularity to be abstract enough to encompass most threat actors. At the same time, it also has the granular ability to distinguish between similar threat actors given enough information. Finally, the model chosen needs to be realistic to reproduce in the tool and language of choice. If the model expresses relations or other properties that are impossible to recreate, it will either be modified or passed for evaluation.

4.3 Evaluation

4.3.1 Reference Scenarios

In order to determine the usability of the ontology, we applied it to two different reference scenarios derived from actual events. The first one was the Russian attack on the Ukrainian power grid in 2015 (7.1) and the attack against a Saudi Arabian petrochemical plant in 2017 (7.2).

Chapter 5

Literature Review

This chapter details the results following the literature review using the methodology from chapter 4. First this chapter goes trough the different steps taken by an attacker during a cyber operation against an industrial control system. This information is then used as a foundation for the explenation of system recoverability degredation attacks that are based on specific actions that an attacker can take against an ICS.

5.1 Attack Pattern

By observing the different attack patterns[22] that were used in other attacks against industrial control systems[23] [24] one can reason a broad set of steps taken by the attackers in order to inflict harm on the ICS and its related systems and networks.

5.1.1 Reconnaissance

Like most modern cyber attacks, attacks of this kind also start with a reconnaissance phase[25]. In this stage, the attacker attempts to locate different points of entry that can be leveraged to gain access to the internal network. Due to the layered network structure of most organizations that deal with ICS[26] this should be either the DMZ or the enterprise network (see 5.1.2). Entry points that are generally considered normal in these types of attacks include internet-facing servers[27], users[28] and suppliers[29].

This is, however, not always the case, as research from Kaspersky in 2016[30] found 220 558 ICS components available to the internet via the Shodan search engine[31]. Based on the findings of the study[30] one can reasonably assume that an attack against an open ICS device from the internet would grant an attacker a fast track through the DMZ and enterprise network straight to the OT network that the exposed ICS is running on.

5.1.2 Intrusion

Breaching the network of an organization is usually done through one of the entry points mentioned in 5.1.1. More often than not, this is accomplished by leveraging phishing[32] to compromise a user device such as a laptop and then move laterally from there. An essential step in the intrusion phase is to establish the ability to reconnect to the network. Ideally, trough legitimate means[33][34]. Examples of this would be to get access to an existing user in the network and use that user to increase privilege[35], or just leveraging default accounts that exist in the network[36].

Command and Control

Another part of this stage would be to install software on the compromised machine that will aid in the further lateral movement (5.1.3).

5.1.3 Lateral Movement

For a threat actor to properly compromise an industrial control system, they would need to employ techniques that would allow the attacker to access the different networked devices that would enable the attack. This could be done by exploiting lacking security procedures for manufacturers of devices on the network and lacking patching and security auditing by using default credentials, leaked accounts, and existing unhatched vulnerabilities.

5.2 System Recoverability Degradation

For most Industry Control Systems, there are eight layers designed to provide safety surrounding the process that the ICS manipulates (see fig 4.2). Three of these: Process control, operator intervention, and Safety Instrumented Systems (SIS), are layers that can be considered digitally manipulative (fig 5.1). This means that provided the correct network access, an attacker can, either partially or entirely, hinder these safety layers by preventing the controlled process from spinning out of control and causing actual physical harm. The order of layers to attack is determined by the system impact and the chance of the attack being detected.

$$C = \{c_1, c_2, .., c_8\} \tag{5.1}$$

C is the set of safety measures (countermeasures) that the ICS has.

$$S(p) = \sum C(p) \tag{5.2}$$

The safety (S) of the process (p) is determined by the sum of the effect the set of safety measures (countermeasures) have on the process.

$$S(p) < T \implies I \tag{5.3}$$

If the safety of the process (S(p)) is lower than the given threshold (T), then this will lead to a critical incident (I)

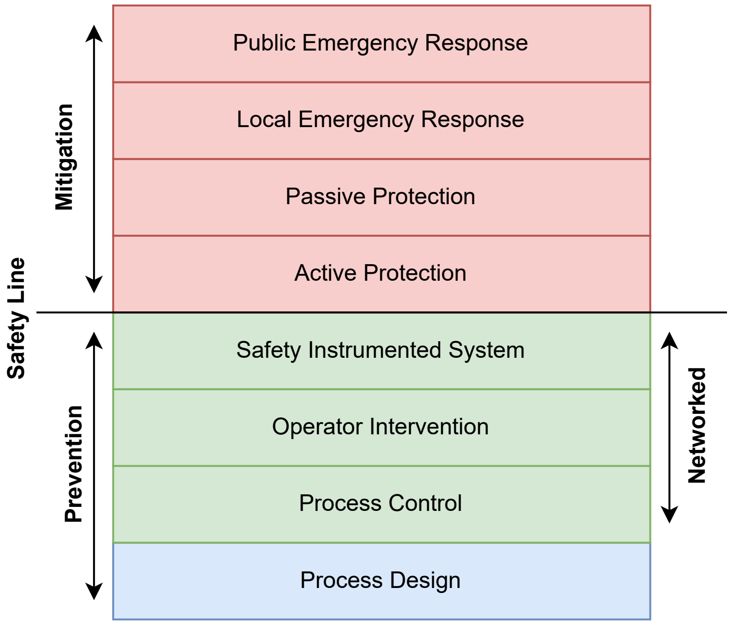


Figure 5.1: Safety layers of an ICS where the layers who are accessible to a threat actor is marked as "Networked".

5.2.1 Component Preconditions

For ICS-specific components to be compromised in such a fashion, a chain of preconditions needs to be met. One can view these similar to the protection layers of figure 4.2, but only in a network context.

The first precondition is networking. Generally, ICS components should not be reachable from any remote network. This is often done by putting a DMZ between the regular corporate network and the industrial control network, and specific components (mainly the SIS) are additionally isolated. Then comes the local access controls such as login, operation system, and networked service security. The networking of an ICS becomes an attack precondition when it is possible to reach the industrial control system from the rest of the network in an unintended way. Misconfigured DMZs are one of the possible reasons behind this precondition becoming an issue.

For attacks against Basic Process Control Systems and Safety Instrumented Systems, the (safety) engineering terminal needs to be compromised so that it can be used to load malware into the firmware of the logic[37]. Lastly, these logic controllers can come with a physical switch that is either called a "memory protection switch"[38] or a "maintenance switch" that is there to prevent accidental or malicious modifications to the firmware of the controller[39].

Only when the relevant chain of preconditions is met for the industrial control system's specific component can the system be adequately compromised.

$$\forall vulnComp \sqsubseteq ((comp, network) : connectsTo \sqcap \exists someVulnerability) \tag{5.4}$$

where for example:

$$firmwareOverwriteable.\top \sqsubseteq someVulnerability \tag{5.5}$$

5.2.2 Disabling the Safety Instrumented System

The purpose of the Safety Instrumented System (SIS) is to safely stop the process from spinning out of control after the system has determined that both the process control and operator intervention layer are not sufficient[40]. This is a last-minute resort to maintain the safety of personnel operating near the physical process, and it can often lead to the process being interrupted, potentially saving lives at the cost of the process value[41]. To summarize: the function of the SIS is to maintain safe process operation or cause the process to fail safely.

In a properly designed Industrial Control System, the SIS should be isolated from the rest of the system, at least on the network layer (5.2.1). This is because in the SIS, just like the ICS the operator uses, the system runs as an IT device. This means that the logic that decides when the system is in such a dangerous state that the SIS needs to be kicked in to prevent disaster is modifiable akin to a slightly restricted computer. In real-world scenarios where the SIS was disabled, the firmware that the device was running on was modified to provide persistent access, modify the sensor values of the SIS always to be outside of a dangerous state, and disrupt operator intervention upon attack execution. It should be noted that, due to convenience, there are many cases where the SIS and the ICS are on the same network.

A properly planned attack against Industrial Control Systems would start by attacking the SIS as this is the last resort for the system. This means that if only this system is disabled, the process would, either by incompetence or misfortune, reach a state where the process will become dangerous if the SIS is not enabled. The goal of the operation would be reached with minimal effort. Additionally, due to the nature of the Safety Instrumented System's necessity, one would usually not notice that the SIS has been tampered with. The delayed effect of an attack against the SIS makes it the ideal first target.

As an IT network device, the Safety Instrumented System can be attacked in several ways; here is an enumeration of the most likely types of attack:

1. Denial of Service:
   a. Remove the SIS from the network.
   b. Turn the SIS off.
   c. Make it unable to take input from the OT network.
2. Firmware Rewriting:
   a. Replace firmware with custom firmware and custom programming.
3. Program Manipulation
   a. Make sure that the SIS never recognises dangerous states by changing values.

5.2.3 Hinder Operator Intervention

The next step is to hinder the operator of the ICS from taking action to prevent the process from spinning out of control[42]. Generally, two different options can be taken to achieve this: Denying Operator Intervention and Control System Misinformation.

Denying Operator Intervention

One way of preventing the industrial control system operators from returning the process to a safe state is to make the terminals they use, such as a Process Control Engineering Station, inoperable. This can be done by performing some denial of service to the terminal itself, such as crashing the terminal or the system it polls data from.

Control System Misinformation

While this is similar to denying operator intervention, the critical difference lies in how operator intervention is denied. In this technique, the threat actor modifies one or more elements of the industrial control system to make it misrepresent the actual state of the system. This hurts the system's recoverability by creating a situation where the system will show that everything is normal. In contrast, the actual situation of the system would trigger an alarm that should lead to operator intervention.

5.2.4 Disrupt Process Control

The final remaining step necessary to put the industrial control system in a dangerous state is to make the process control logic unable to return the process to a safe state. In regular operation, the process control logic, similar to the SIS, will detect that some part of the process is not in an optimal state. It will correct this discrepancy by activating an actuator. An example of this would be feeding in more water to cool a nuclear process and generate more steam and, by effect, electricity.

In practice disrupting the process control will involve either performing some denial of service attack on the logic device itself or, more effectively, the logic could be altered by accessing an engineering station.

In the first case, the attacker relies on the process spinning out of control on its own when the process control logic performs no actions. From an attacker's side, the issue with this is that process design makes it so that not all processes can reach a particular harmful state on their own. Going back to the nuclear reactor example, this would be a good candidate for such an attack as the core will suffer a meltdown due to the heat[43]. Other processes, such as hydroelectric power production, will either remain in the same state or slowly reduce production until the process halts.

In the latter case, the attacker leverages the existing logic system to increase the damage potential further. This can be done by manipulating the process to increase instability, such as increasing the reactivity in a nuclear reactor simultaneously, as the effect of the actuators meant to reduce reactivity is either reduced or removed altogether.

5.3 Harm Infliction

Different actions that can put the ICS in a state where it is not able to safely recover from a process running out of control. These can be expressed as a set of rules:

1. ∀disableSIS->decrease(systemRecoverability(x))
2. ∀hinderOperator->decrease(systemRecoverability(x))
3. ∀distruptProcessControl->decrease(systemRecoverability(x))

By applying these rules in the system, one can determine if it is likely that the system is under attack and that there is a need for either investigation or preventive/preemptive measures to prevent harm.

The real harm (H) that can be extracted from making an industrial process run out of control can be expressed in the following manner:

$$H = D(p) - \sum C(p) \tag{5.6}$$

In regular operation, the damage potential of the process (D(p)) would be lower or equal to the effect of the different safety layers ($\sum$ C(p)). When the goal of a threat actor is to cause as much harm as possible, the threat actor would need to reduce the effect of the countermeasures. As stated in section 5.2 a cyber threat actor[44] only possesses the ability to directly disrupt three of these as they are the only ones directly connected to the cyber domain.

5.4 Threat Intelligence

By observing different artifacts in the network, one can use this to determine the likelihood of an attack being performed by a particular threat actor.

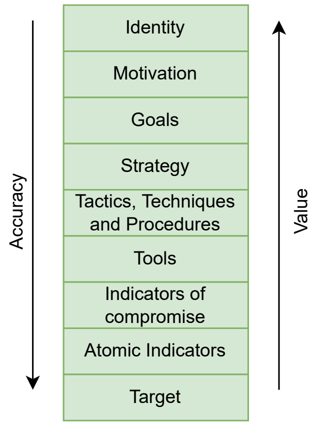


Figure 5.2: Threat Intelligence Model

5.4.1 Cyber Threat Intelligence Model

In order to do this, one needs to have a structure that can take into account different identifiers of a threat actor and put them together. By comparing different threat intelligence ontologies, Bromander[18] makes a good argument for this structure:

Identity

This is intended to be the real-world identity of the threat actor. This can often be the APT designation that a threat actor has, which can be linked to different names given by other organizations.

Motivation

This is the driving force of the threat actor. Generally, this can be described as the reason behind a threat actor trying to achieve its different goals. If you have identified a set of threat actors based on the tools used, but only one of the threat actors is likely to have a motivation that would make your organization a target, the motivation aspect of this threat model will allow resources to be allocated based on this assumption.

Goals

Bromander bases the definition of a threat actor's goals based on a paper by Fishbach and Ferguson[45]. A threat actor's goal is the desired end state of an overall objective. For a state-sponsored threat actor, a goal would be "reduce hostile states and associated entities' digital abilities" or, more relevant for this paper, "cause physical damage inside a hostile state or country". The goal can be considered a more granular or higher resolution version of the motivation.

Strategy

This is a high-level way of expressing what approach a threat actor tends to take. One can view this as a way to describe the attack in a non-technical manner.

Tactics, Techniques, and Procedures

This section describes the different technical methodologies that a threat actor will use to realize the strategy. Initially, Bromander describes three subcategories of TTP: Attack patterns, malware, and infrastructure. In this paper, we do not agree with the arguments presented by Bromander; see why in 8.2.

Tools

The model describes different tools or software that an attacker can utilize and install. There are multiple ways tool usage can be detected. This can range from installation to behavior artifacts observed on the network. Examples of network activity can be port and protocol usage, data frame artifacts, and other packet information that can, to a varying degree, determine the software in use.

Indicators of Compromise

These can be artifacts on the network and applications indicating that a system has been compromised. They will often come due to tool usage or typical attacker behavior, often described as abnormal user behavior.

Atomic Indicators

Atomic indicators are often short-lived and infrequent changes, but these can be invaluable in determining if a specific attacker is present in the system. This can be done by associating specific file hashes, domain names, and IP addresses with a threat actor.

Target

This can be what types of targets (Sector, size, nationality) an attacker is believed to target or individual entities that the threat actor is confirmed to target. Intuitively it is reasonable to assume that a state-sponsored threat actor is unlikely to target organizations within its state borders unless special conditions are met (8.4).

Chapter 6

Ontology

This chapter describes the ontology results of this thesis. First, the chapter shows how an ICS is modeled in an ontology. Then the author shows how the dangerous states are modeled and how the theory from the literature review translates into an ontology. The different preconditions are also modeled, and the author describes the thought process behind how the three different states are determined. Finally, the author shows how threat intelligence can be incorporated into the ontology to enhance its effect.

6.1 Industrial Control Network

Modeling an industrial control system is not the most straightforward task due to the varying usage of terminology and overlap between different terminologies such as DCS and SCADA. On the highest level (See figure 6.1): An industrial control system consists of a set of sensors and actuators that manipulate and monitor a process. These actuators are controlled by some logic that takes input from the sensors and makes decisions based on instructions that are programmed into the logic.

This programming is usually performed using an engineering station. Additional instructions can come from a central processing server that takes input from all logic devices and can either make decisions based on its programming or manual operator intervention is required. A human on an operating station does this manual operation. An archive server also stores all the process information for further analysis.

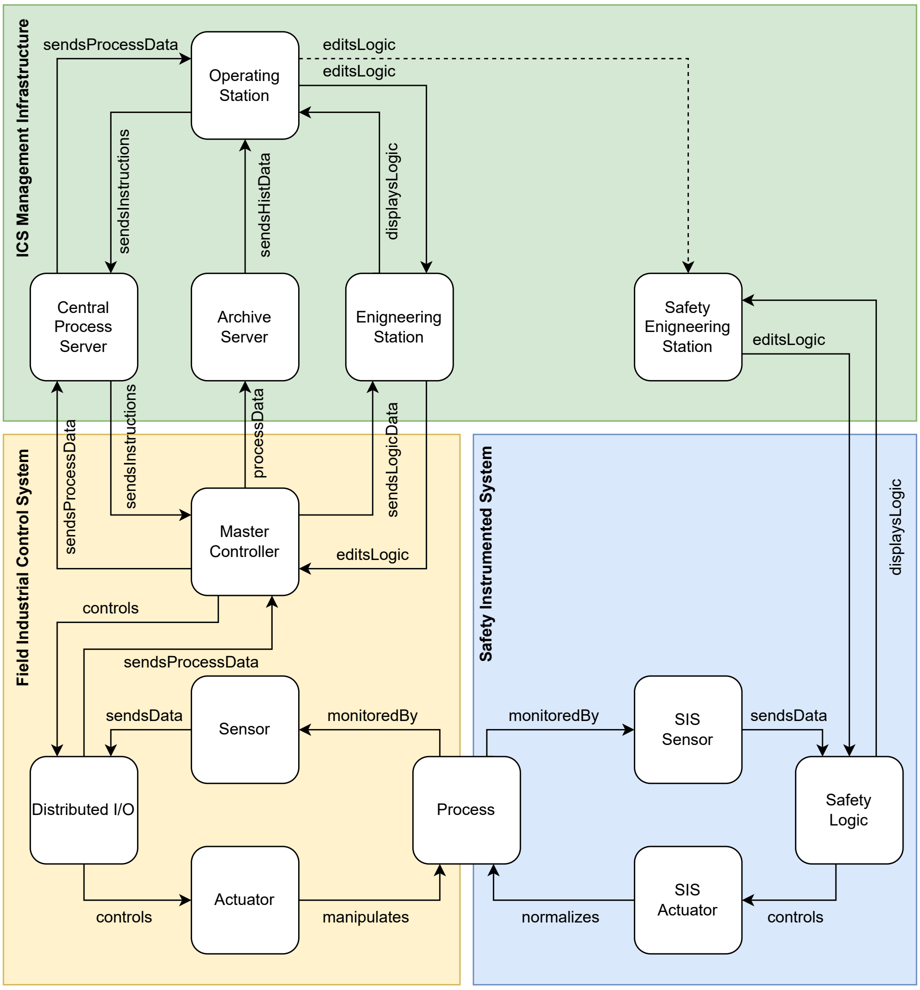


Figure 6.1: High Level Diagram of a Distributed Control System

6.1.1 Industrial Control System Classes

In order to model an industrial control system, there is a need to define the different classes of an object involved in the system. Six main classes needed to be defined for this thesis. They were mainly defined by the executive function of the class, such as management, communication, etc.

ICS Class Hierarchy

- industrialControlSystemCommunication
    - processControlNetwork
- industrialControlSystemComponent
    - actuator
        - safetyActuator
    - logic
        - masterController
        - distributedInputOutput
        - safetyLogic
    - sensor
        - safetySensor
- industrialControlSystemManagement
    - processControlEngineeringStation
        - safetyEngineeringStation
- network
    - corporateNetwork
    - processControlNetwork
- process
- processControlSystem
    - basicProcessControlSystem
    - safetyInstrumentedSystem
        - safetyActuator
        - safetyLogic
        - safetySensor

This class definition is intentionally high level as the number of variations of different components is so numerous. Instead, this level of granularity should be more than good enough to reasonably model a given industrial control system in combination with the object properties described in 6.1.2.

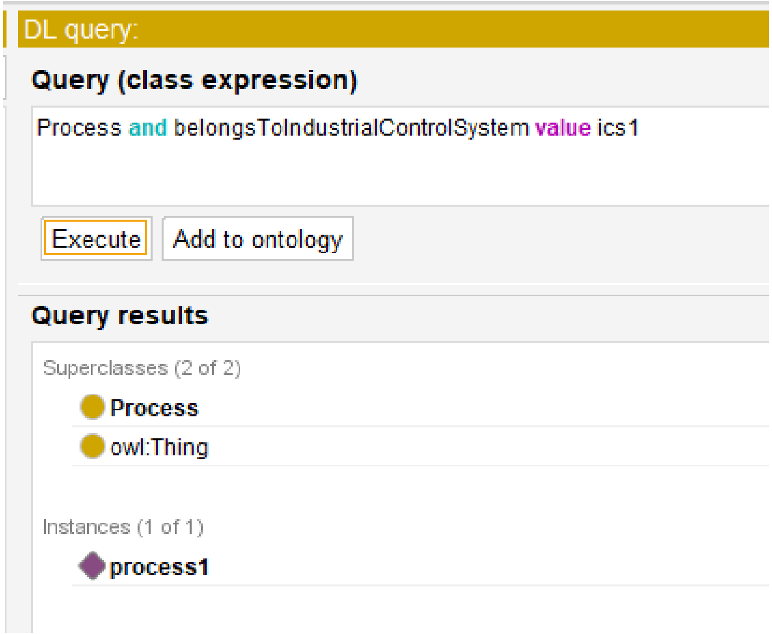
6.1.2 Industrial Control System Object Properties

Each component of the industrial control system performs only has a small number of properties due to the functional and high-speed nature of the system. The central relations of the ICS are well described in 6.1. From these relations, there are also a set of properties implied from the property's function. Examples of this can be the property "connectsTo". This property has two sub-properties that imply its existence: receivesData and sendsDataTo.

While this functionality is not directly necessary for explaining the relations of the industrial control system, it will help describe the security elements of the ICS in 6.3.

6.1.3 Modeling an ics

After the classes and object properties are put in place, all that remains is to fill the ontology up with individuals representing the industrial control system that one wants to model and describe the baseline relational object properties, such as what individuals manipulate the process and how they all connect. The rest of the inferences is done through the reasoning engine of the ontology and follows a lot along the lines of 6.1.2



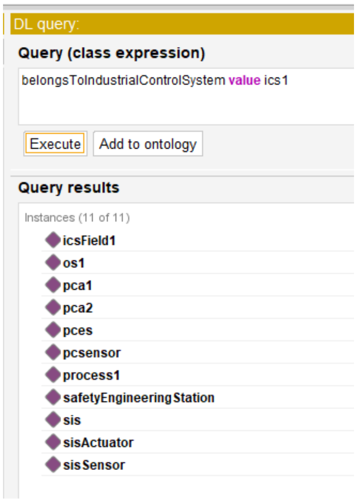
```
Process and belongsToIndustrialControlSystem value ics1
```

```
Superclasses (2 of 2)
    Process
    owl:Thing

Instances (1 of 1)
    process1
```

Figure 6.2: Queury what processes belogs to the ICS called "ics1"

The figures in this section from the DL query tool in protege show an example of a nondescript industrial control system and the components necessary for it to function.



belongsToIndustrialControlSystem value ics1

Instances (11 of 11)

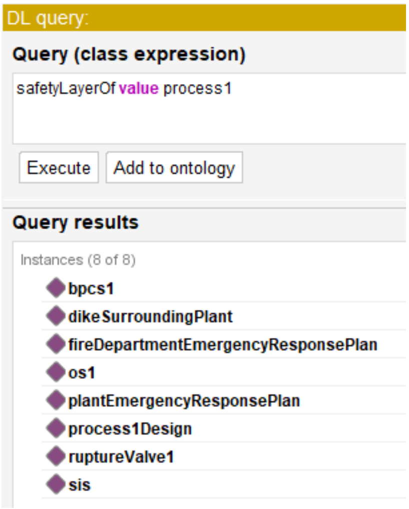
- icsField1
- os1
- pca1
- pca2
- pces
- pcsensor
- process1
- safetyEngineeringStation
- sis
- sisActuator
- sisSensor

Figure 6.3: All individuals belonging to "ics1"

6.2 Dangerous State Ontology

Determining the dangerous state of an industrial control system is described in 5.2. This theory was applied to the ontology and resulted in an ontology that was able to determine what individuals in the entire ICS field contributed to the safety of the local process and what actions, either by or taken against the ICS, would reduce the recoverability of the system and increase the potential harm.

6.2.1 Modelling Safety Layers



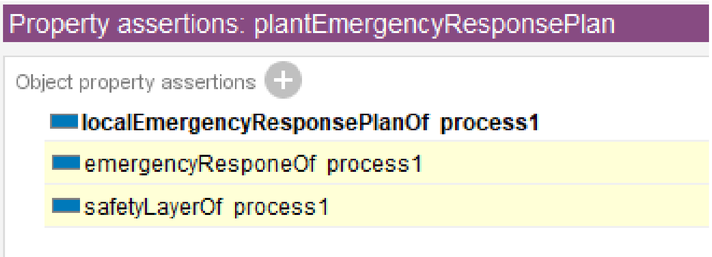
safetyLayerOf value process1

Instances (8 of 8)

- bpcs1
- dikeSurroundingPlant
- fireDepartmentEmergencyResponsePlan
- os1
- plantEmergencyResponsePlan
- process1Design
- ruptureValve1
- sis

Figure 6.4: Safety Layer Query

The ontology determines what constitutes a safety layer of the process by determining the safety function of the class. This is done through the defined reasoning of the ontology, where specific object properties imply that they increase the system's safety. This also implies what kind of safety layer this object property belongs to.



Property assertions: plantEmergencyResponsePlan

Object property assertions

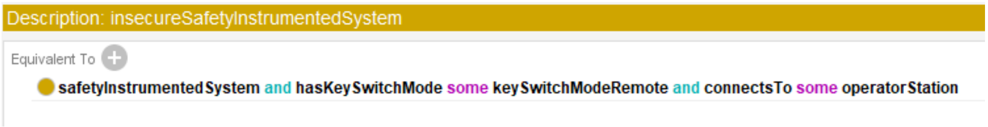
- localEmergencyResponsePlanOf process1
- emergencyResponeOf process1
- safetyLayerOf process1

Figure 6.5: Safety implications of the plant emergency plan of "process1"

As seen in figure 6.5 creating an emergency response plan for the plant in question that encompasses "process1" one can see that the ontology infers that the process now has at least one emergency plan that acts as a safety layer for the process.

6.2.2 Identifying Component Preconditions for ics Attacks

The ontology can utilize reflexive relations in order to express the different component preconditions required for an attack component to exist (5.2.1). Figure 6.6 shows one of the ontology's object properties that are required for a Safety Instrumented System to be considered insecure. This insecure state is required for the SIS to become the class "nonFunctionalSafetyInstrumentedSystem" and for the threat actor to be able to enact the object property "disableSafetyInstrumentedSystem".



Description: insecureSafetyInstrumentedSystem

Equivalent To

safetyInstrumentedSystem and hasKeySwitchMode some keySwitchModeRemote and connectsTo some operatorStation

Figure 6.6: Expression of condition required for firmware overwrite of SIS

As a proof of concept, the safetyInstrumentedSystem safety layer has an insecure state constructor based on three conditions: It is a safety instrumented system, it can connect to an operator station, and the firmware is overwriteable. These are all points of information that could be observed on the network, given that one of the conditions for this insecure state is that the safety engineering station is connected to the network. When all of these conditions are met, the safety instrumented system moves from a functional to an insecure state, which is part of the constructor for a nonFunctional SIS. The insecure state of the SIS makes it possible for an attacker to compromise it and then attack the safety layer of the associated process. If an attacker were to attempt to sabotage the SIS without the preconditions for the insecure state present, the SIS would remain in the functional model. If the SIS is in the aforementioned insecure state and is attacked (sabotaged) by a threat actor, it will now move into the nonFunctional state.

$$nonFunctionalSIS \equiv (insecureSIS \sqcap (TA, insecureSIS) : sabotage) \tag{6.1}$$

Where TA = Threat Actor and

$$insecureSIS \equiv (SIS \sqcap memoryOverwriteable \sqcap (\top, OS) : connectsTo) \tag{6.2}$$

OS = Operation Station.

The case for determining if the operator intervention is in an insecure state is based on the operation intervention being reliant on either one or more components for performing communication to the BPCS. This ontology has only considered one scenario where a vulnerable communication component could stop operation intervention. The reason behind this is mostly time and that there is a real world example where a threat actor exploited this vulnerable communication component element to prevent operator intervention (7.1).

$$unavailOperInter \equiv vulnOperInter \sqcap (TA, vulnOperInter) : sabotage \tag{6.3}$$

Where TA = Threat Actor and

$$vulnOperInter \equiv operInter \sqcap (operInter, vulnComponent) : reliesOn \tag{6.4}$$

Where

$$vulnCommComponent \equiv commComponent \sqcap memoryOverwriteable \tag{6.5}$$

In order to determine if the operator intervention is in one of the unwanted states, one needs to understand if for this experimental ontology, the operation intervention is reliant on a communication component to work and if it is possible to attack this communication component. If one again refers to 7.1 it is clear that is it possible to obtain both the information that operator intervention is reliant on one kind of serial to ethernet converter that can have its firmware rewritten remotely.

Finally, we must determine the preconditions for the process control layer to be put out of play by a threat actor. This also follows along similar lines to the previous constructors in this section. Usually, the process control relies on the local logic to perform its functions. In the example model in this ontology, the firmware of this logic is remotely rewriteable in a similar fashion to the SIS modeling.

$$nonOperationalPC \equiv vulnBPCS \sqcap (TA, vulnBPCS) : sabotage \tag{6.6}$$

Where TA = Threat Actor

$$vulnBPCS \equiv BPCS \sqcap (BPCS, vulnLogic) : reliesOn \tag{6.7}$$

Where

$$vulnLogic \equiv logic \sqcap memoryOverwriteable \sqcap (logic, OS) connectsTo \tag{6.8}$$

OS = Operation Station

6.2.3 Determining What Relations Contribute to the Dangerous States

This paper mainly focused on the safety layers that are mentioned in 5.2 and the different object properties that could affect them. Different classes of this ontology can affect the different safety layers differently. Mainly this ontology focuses on the different actions that a threat actor can take on the industrial control system. Actions such as disabling operation stations. This action implies multiple things for the ontology, such as a denial of service taking place that can be considered sabotage. Additionally, this implies that the recoverability of the system is degraded and that the system can now be considered unstable, depending on the affected safety layer.

6.2.4 Determining Dangerous State

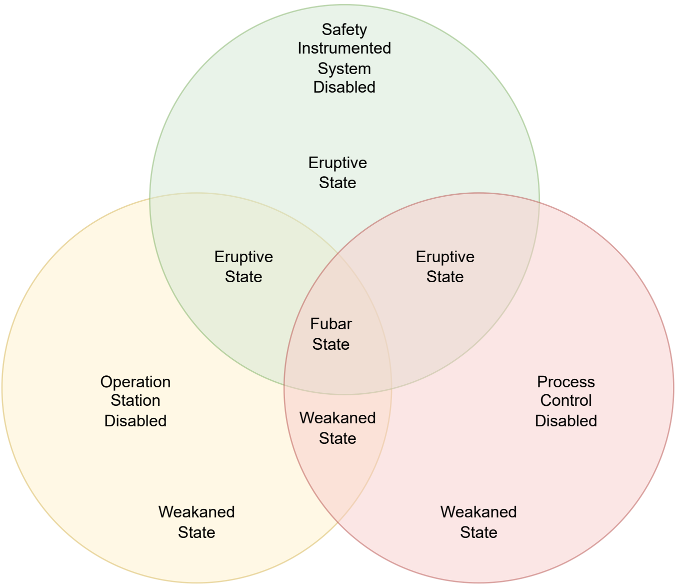
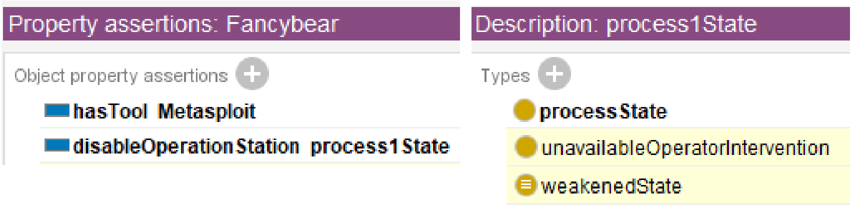


Figure 6.7: Venn diagram of safety layer disruption and process state

By determining which safety layers of the industrial control system are disabled, the ontology can determine what dangerous state the process is in by leveraging class expressions. Figure 6.7 shows the different overlaps of safety layer compromise that would result in the different identified process states. If the operation station or process control is disabled, the process is in a weakened state when it comes to safety, given that the safety instrumented system works as intended[41]. As the SIS is the last IT/OT safety layer, the system should not spin so much out of control that the remaining safety layers (fig. 4.2) need to be activated.

However, if the SIS is disabled, the process would be in an eruptive state. This is because just process design, control, and operator intervention may not be enough in some instances to prevent a safety incident. The process state remains eruptive as long as one of fewer of the other networked safety layers(fig. 5.1) are compromised.

If all three networked safety layers are compromised, then the process has its most unsafe state; fubarState. This state is the "ideal" state for a threat actor to reach as it will do the most amount of damage, and all that is left for the attacker to do is to "push the metaphorical button" by forcing the process to spin out of control. By increasing the process reactivity, for example.



Property assertions: Fancybear

Object property assertions

hasTool Metasploit

disableOperationStation process1State

Description: process1State

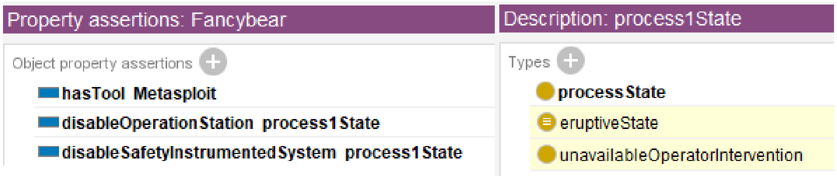
Types

processState

unavailableOperatorIntervention

weakenedState

Figure 6.8: Threat Actor "Fancybear" disables the Operation Station, resulting in the process state to become the class "weakenedState"



Property assertions: Fancybear

Object property assertions

hasTool Metasploit

disableOperationStation process1State

disableSafetyInstrumentedSystem process1State

Description: process1State

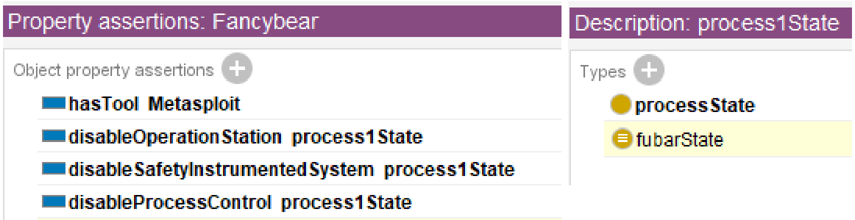
Types

processState

eruptiveState

unavailableOperatorIntervention

Figure 6.9: Threat Actor "Fancybear" disabling the SIS causing the process state to become the class "eruptiveState"



Property assertions: Fancybear

Object property assertions

hasTool Metasploit

disableOperationStation process1State

disableSafetyInstrumentedSystem process1State

disableProcessControl process1State

Description: process1State

Types

processState

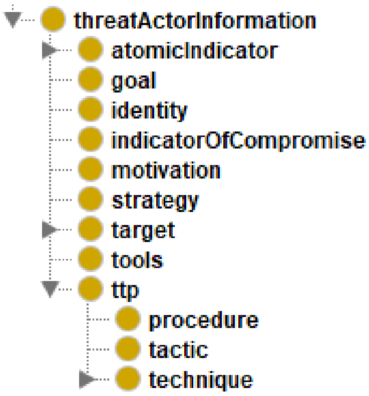
fubarState

Figure 6.10: Threat Actor "Fancybear" disables all three safety layers, resulting in the process state to become the class "fubarState"

6.3 Threat Intelligence

6.3.1 Threat Actor Identifiers Class Structure

To determine the identity of a threat actor we took the structure from Bromander's [18] Cyber Threat Intelligence Model (5.4.1 & 8.2) and incorporated it into the ontology. This gives the ontology 9 or 11 (depending on the interpretation of TTP) different classes of Threat Actor Information that can identify or exclude different threats depending on the indicators found in a given attack.



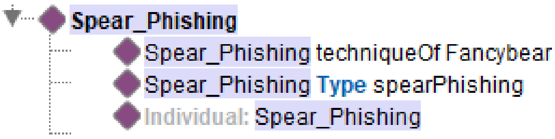
- threatActorInformation
  - atomicIndicator
  - goal
  - identity
  - indicatorOfCompromise
  - motivation
  - strategy
  - target
  - tools
  - ttp
    - procedure
    - tactic
    - technique

Figure 6.11: Class Structure of Threat Actor Information

The class structure used in this thesis is not complete as that would mean defining thousands of different classes and subclasses of Threat Actor Information. It is, however, defined enough to prove the concept of utilizing this ontology to identify threat actors.

6.3.2 Applying Threat Identifiers

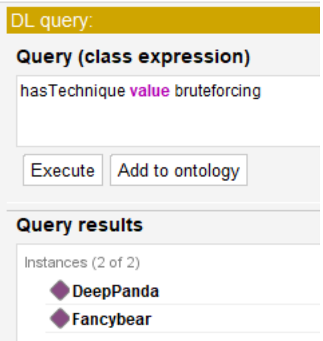
In order to associate specific threat actors with the different identifiers, we created one individual for each identifier. By associating these unique techniques with at least one threat actor, one can relate different identifiers to a specific attack and query the ontology for the threat actor(s) that have a matching set of identifiers to the attack.



- Spear_Phishing
  - Spear_Phishing techniqueOf Fancybear
  - Spear_Phishing Type spearPhishing
  - Individual: Spear_Phishing

Figure 6.12: Example of Technique Identifying "Fancybear" as a Potential Threat Actor

One can utilize the different layers of granularity of the ontology to better distinguish between threat actors in the following way:

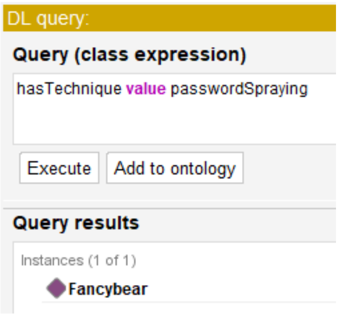


hasTechnique value bruteforcing

Instances (2 of 2)
- DeepPanda
- Fancybear

Figure 6.13: Two threat actors with the same technique

However, only looking at brute-forcing as the technique class is not enough to determine which threat actor is currently behind the attack. Instead we can take it further by specifying the subclass of bruteforcing that is taking place:

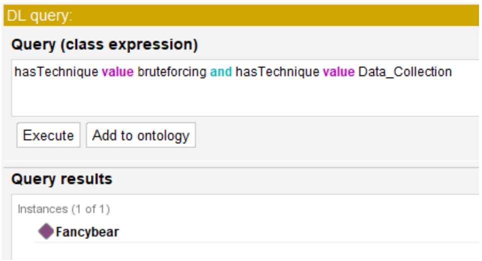


hasTechnique value passwordSpraying

Instances (1 of 1)
- Fancybear

Figure 6.14: Threat actor with unique attack technique

Or we can add in an additional identifier that also has been observed in the current attack:



hasTechnique value bruteforcing and hasTechnique value Data_Collection

Instances (1 of 1)
- Fancybear

Figure 6.15: Threat actor with a unique set of attack techniques

Chapter 7

Reference Scenarios

This chapter goes through two different scenarios and applies the ontology to see the how it measures up to real world scenarios.

7.1 Ukraine Power Grid Attack 2015

7.1.1 Attack Description

In 2015 the Ukrainian power delivery network was attacked by a foreign threat actor. The attacker sent malware disguised as an Excel attachment in an email opened by an employee of the power delivery company. After the attachment was opened and the employee's laptop was compromised, the attackers moved laterally through the network. They found an open connection between an IT and OT system[46] giving the attacker access to a supervisory system of the power grid.

From here, the threat actor exploited two sets of vulnerable states. One of these states was that the serial to ether converters could have their firmware overwritten and that there was no other technical function to prevent illegal firmware overwriting. The second state was that it was possible to remotely reconfigure the UPSs so that they would not kick in the event of a power loss.

For the first vulnerable state, it would be possible to attack the second safety layer (fig. 4.2) by not allowing the local process control to execute commands. The second vulnerability attacked the third safety layer by preventing the operators of the power delivery network from performing manual operator intervention to keep the process running.

This attack resulted in power outages in Ukraine but no loss of life. The lack of direct physical harm results from the nature of the process in question, as it can not be forced into a state that would be hazardous. This is also why these kinds of industrial control systems do not have safety instrumented systems in place.

7.1.2 Modeling the Attack With the Ontology

The ontology is slightly over-dimensioned for detecting vulnerable states in this kind of industrial control system as it focuses on the hazard elements of attacks against ICSs. However, this does not mean that it is impossible to model this attack in the ontology.

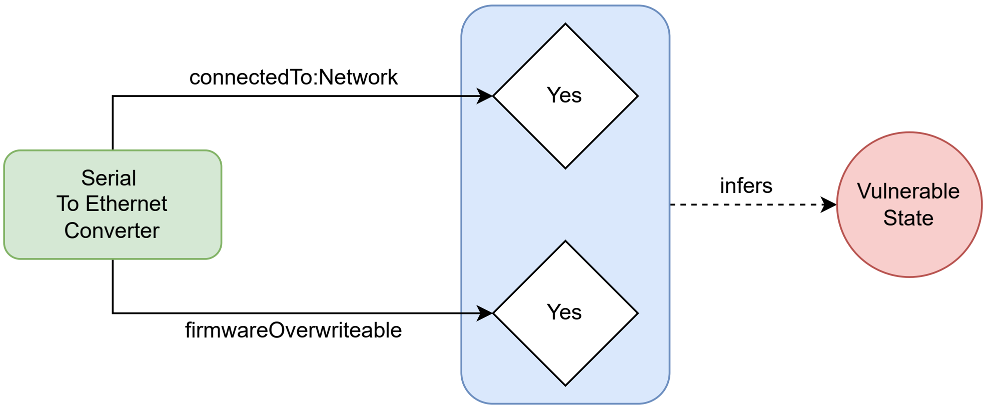


Figure 7.1: Vulnerable State Inference of Serial to Ethernet Converter

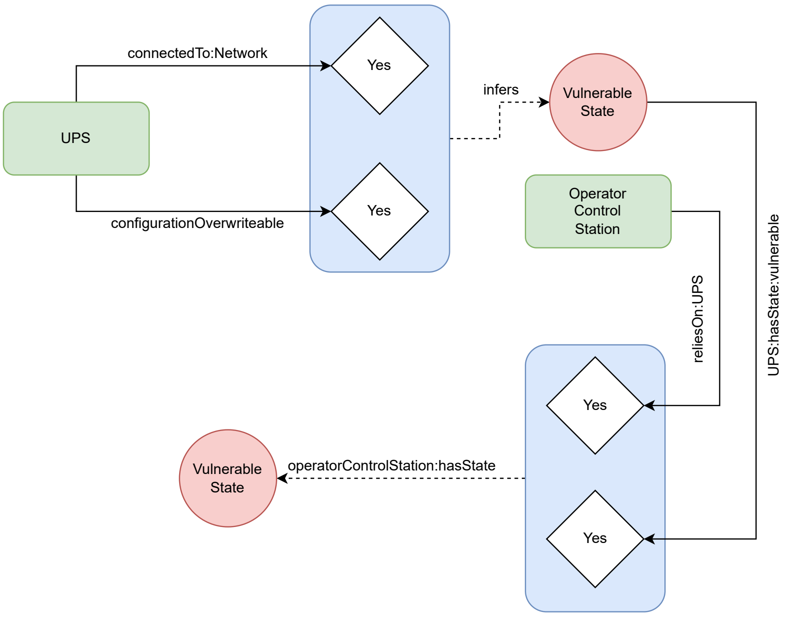


Figure 7.2: Vulnerable Sate Inference of Operator Control Station and UPS

By modeling the scenario, the ontology infers two different vulnerable states that are caused by the configurations of each of the devices, their connection status, and what other components rely on them to work (figure 7.1 & 7.2.)

This is then put in relation with the rest of the ontology, resulting in the Power

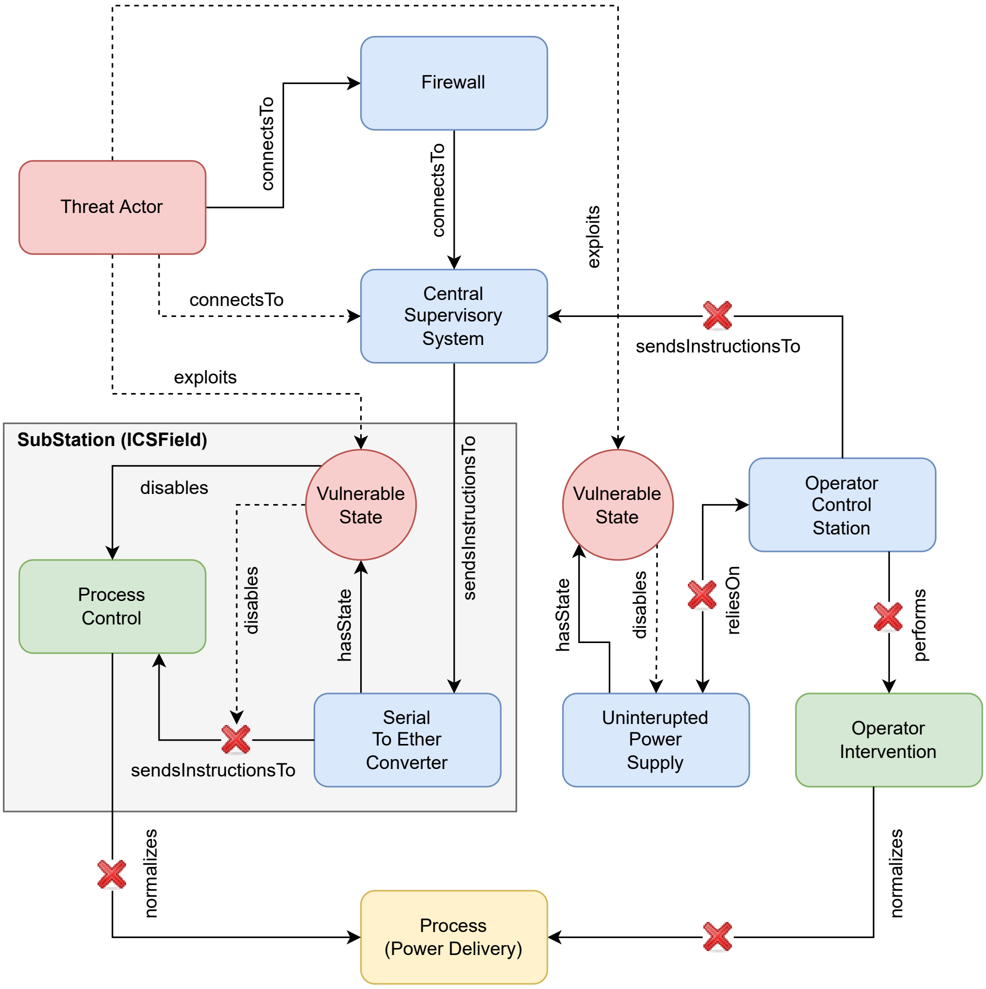


Figure 7.3: Model of the Ukraine Power Grid Attack

Delivery Process obtaining a "Weakened State" as it has two safety layers compromised in accordance with figure 6.7.

7.1.3 Threat Intelligence

The ontology can pick up on multiple indicators of an attacker in this scenario. It can attribute TTP, tool usage, and target. This makes it able to make a reasonable inference about the attackers' identity. While it is not definitive enough to identify the attackers, it does reduce the number of potential threat actors. This attack has already been attributed to a specific threat actor, and the ontology (if populated enough) would have agreed that this threat actor is a likely candidate.

7.2 Attack on Saudi Arabian Petrochemical Plant 2017

7.2.1 Attack Description

In 2017 a Saudi Arabian petroleum plant was attacked by an unknown cyber threat actor. There is speculation about how the attackers made their way into the system, but one source claims that they gained access to the network already in 2014 due to a poorly configured firewall[47]. The threat actor then gained access to the safety engineering station of the process plant, either by using a vulnerability in the safety engineering station or legitimate credentials and uploaded firmware that would enable the attacker to disable the safety instrumented system and establish a backdoor into the network. The attacker showed a good understanding of the working components of the industrial control system and a tremendous ability to utilize resources in order to be well prepared for such an advanced attack[48]. In order for the attacker to perform the aforementioned firmware upload, the SIS control module needed to be in an operating mode that allowed for the firmware to be overwritten (i.e., not in read-only mode[39]).

7.2.2 Modeling the Attack With the Ontology

First, we look at the components in play for this scenario. From the information available to the public, there is only one component of the industrial control system mentioned directly, and that is the Safety Instrumented System and the four components that make up the entire SIS logic. These are a Triconex 7400027-100 Rack, a Triconex 8310 Power Module, a Triconex NCM 4329 Communication Module, and a Triconex 3008 Logic Module[49]. These components put together will fall under the category of "safetyLogic".

The process and associated Distributed Control System are also in place, but they are not mentioned in any particular detail.

For the ontology to determine if the SIS is vulnerable or not, it needs to check for specific conditions. The first part that needs to be checked is how the SIS is networked. As stated in 5.2.2, for it to be possible to attack the safety instrumented system practically, it needs to be connected to the network. Ideally, SISs should be isolated, but this is often perceived as impractical. This was also the case in this scenario.

Additionally the SIS was not in a read-only memory mode. Instead the keyswitch[39] on the SIS rack was set to "remote" instead of the ideal "run" (Run = read-only) mode, allowing the program logic to be overwritten.

As described in fig 7.4 these conditions made it possible for the safety instrumented system to be put in a state that allowed for it to be compromised and disabled. Operating under the assumption that these conditions are observable, the ontology recognizes this as a vulnerable state for the SIS. The system can be subject to further attacks. This information could already, at this stage, be used by the petrochemical operations personnel to either put the SIS module in "run"

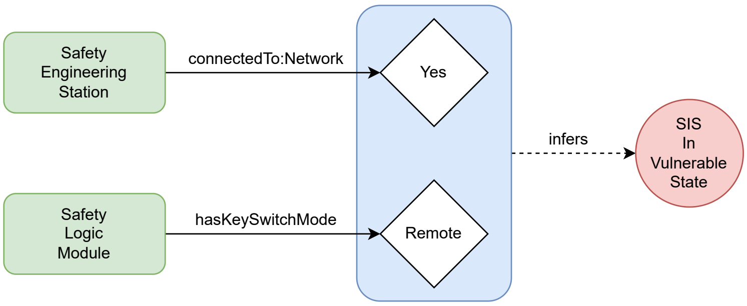


Figure 7.4: SIS Vulnerable State Inference by the Ontology

mode, isolate the safety engineering station from the network, or take other preventive steps to ensure that the SIS can not be disabled.

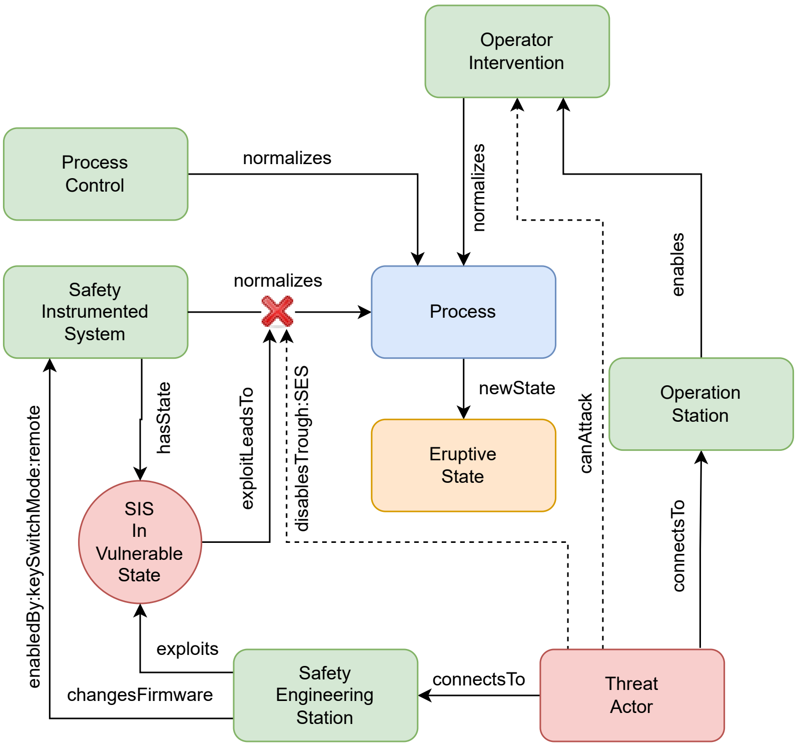


Figure 7.5: Ontological Model of Triton Attack

The vulnerable state is shown in fig. 7.4 remained, and the threat actor behind the Triton attack chose to execute the firmware overwrite. During an actual attack, this will make the SIS unable to prevent the process from running out of control, removing one of the safety layers of the process in question as shown in figure 7.5. These object relations make the ontology infer that the process is in an eruptive state based on figure 6.7.

The threat actor would have access to the operator stations or other components critical to enabling operator intervention, based on the layered network model that industrial control system networks follow[41]. The ontology recognizes these relations and infers that the threat actor has the object property "canAttack" the operation intervention safety layer.

7.2.3 Threat Intelligence

Sadly there is little threat intelligence information related to this attack. Assuming that the threat actor behind this attack is a known threat actor that could be put in the threat intelligence model utilized by this ontology (6.3) and then the actions taken by the threat actor could be mapped to the model, and one or more likely existing candidates could be attributed with performing the attack.

Chapter 8

Discussion

8.1 Lack of Terminology Consistency

A major issue that made it hard to be effective when writing this thesis was the inconsistency of terminologies in the industrial control system space. This is best summarised by the similarities between SCADA and DCS systems. These systems are functionally similar, but DCS is mainly used in connected systems not too remote in geography. SCADA is more data collection-oriented, while DCS is more control-oriented.

Another example of this overlap in terminology is Programmable Logic Controllers and Remote Terminal Units. From the author's understanding, it seems as if either RTU is a class of PLC or that they are two different types of logic controllers. Functionally they seem to do the same thing.

8.2 Cyber Threat Intelligence Model

In Bromander's paper regarding the cyber threat intelligence model, Bromander puts all elements of the Tactics, Techniques, and Procedures as one single class[18]. After performing the literature review for this paper, the author is under the impression that TTP is a more nuanced identifier of threat actors, especially if they are similar in their approach. The definitions of TTP as used in this paper (2.1) state that each of the elements describes the behavior of a threat actor in an increasingly granular manner. We chose not to put these increasingly granular behavioral descriptions in a hierarchical structure, as this does not seem like a natural choice. It is more important than these are all grouped as a sub-class of TTP to structure the information in the ontology more efficiently. It is worth mentioning that each part of the TTP naturally has many sub-classes that can be described, but this was not expansively done for this thesis to fit the scope and time allotted for this thesis.

8.3 Advantages of Threat Intelligence

In the author's opinion, one of the greatest advantages of utilizing an ontology that incorporates threat intelligence with Bromander's threat intelligence model is that by narrowing down the potential acting threat actor, one can better predict its next move. This suddenly shifts the blue teamwork from reactive to proactive.

Take any given threat actor "TA". This TA is an advanced threat actor that works for a government hostile towards the government of a petrol plant "P". Using a threat intelligence knowledge base that is structured like the structure in this paper, the blue team of P has much information on different identifies for many threat actors, including TA.

TA initiates an attack against P and breaches some devices on the network. A client in the corporate network employs a phishing attack and loads their preferred tools for further exploitation of the client. P's blue team's IDS notices the attack, technique used, and files loaded on the device. They now know that there is an attack present, the attacker targets petroleum plants in this country, there are some tools present, and some of them are custom. This should give the blue team enough information to determine who the attacker is and infer the likely goal of the threat actor.

By knowing the goals of the threat actor, the blue team can start working towards securing the assets that are under threat from TA parallel to the work required to isolate the already compromised hosts and prevent the attack from spreading throughout the network. Suppose TA is a threat actor with the goal of obtaining information from businesses and other entities. In that case, the blue team can prioritize securing information sources, like databases, and prioritize by the value of the information stored and how "far" away it is in-network terms (Different VLAN, behind DMZ, etc.).

8.4 States attacking their own organizations

The threat intelligence part of this thesis and ontology does not consider if a state-sponsored actor were to attack an organization within the state's borders. This could be an exciting avenue to explore, but the author did not find time to prioritize this.

8.5 Experience Based Literature

An issue with the results gained from chapter 5 and especially chapter 7 is that there is both anecdotal, potentially politically biased, and incomplete information when it comes to detailing attacks against industrial control systems.

The report gathered detailing attacks and other threat information is released by private companies. Some of these companies are considered potential security threats by other countries. This can, knowingly or unknowingly, contribute to the company not realizing full transparency.

The potentially embarrassing nature of the post-incident report, detailing the mistakes made by the affected entity, can also lead to certain key elements of information being let out of the report. This can be more true when performing a research project such as this thesis, as a lot of "special interest" information had to be gathered that might not be interesting for the authors of the varying reports.

Lastly, some of the information used in this thesis, mainly regarding chapter 7, is anecdotal and therefore not as reliable as one would like when writing a master's thesis. These accounts of the different events during an incident could be subject to wrongful recollection, an intention to preserve the self-image, and other biases.

The author of this thesis has taken all of these elements of bias and misinformation into account when selecting sources.

8.6 Ethical Considerations

One of the issues with this paper is the nature of its content and what the answers to the research questions would entail. Essentially this paper could be considered a recipe for creating the most amount of physical harm. It is entirely possible to utilize the information in this paper to create dangerous situations in industrial control systems by following the steps laid out in 5.2.

However, three items make the publication of these findings ethically defensible.

Firstly the information in this paper is abstract and does not describe how to do the maximum amount of damage to a specific industrial control system. Neither the literature review nor the ontology describes a specific system ICS, its associated safety layers, or the multitude of nuances that will play if this theory was to be put into practice.

Secondly, this thesis is based on information that already exists, and it is not particularly challenging to come by. All sources utilized in this research are listed in the bibliography.

Thirdly the inspiration behind this paper is the Triton incident in 2017. From that incident, it is apparent that the threat actors performing the attack already knew the order of operations necessary to make a digital attack against the ICS as hazardous as possible. This makes it a reasonable assumption that any threat actor with the competence and resources required to perform such an attack is already aware of the information in 5.2 and that this paper stands to benefit owners and operators of industrial control systems as they may not have this information.

8.7 Lack of Domain Knowledge

The author of this paper does not have a high enough level of knowledge in the industrial control system field to model and utilize an industrial control system ontology without fault. Instead, the ontology is intentionally created abstractly as this still makes it able to cover the main aspects of the thesis while simultaneously reducing the number of mistakes and misconceptions.

8.8 Advantages of Ontologies in Cyber Security Operations

By writing this thesis on industrial control system cyber security and ontology, it is clear how these concepts can be used in conjunction to enhance security. The traditional security information and event management (SIEM) systems will work fine on their own, and an ontology is not required for them to function. However, as discussed in 8.3 and further in this section, ontology can be used to enhance these SIEM systems.

Understanding that modern-day systems, both industrial control systems and non-OT systems, have different subsystems that work synergetically and have their function that all will affect other subsystems. This is even more true with the popularization of microservices and a more component-based approach to delivering services. Ontologies will help the defenders of a particular system abstractly look at it and make it easier to identify the ramifications of different security events. A properly designed, populated, and implemented ontology will understand certain relations that might not be intuitively clear to a human. Ontologies can, in this respect, be used to better identify vulnerable parts of the system that will have a more significant impact on the value generated by the system if they were compromised and what elements of the system can be used to create this compromise or what elements will contribute to the effect of the compromise.

It is also important to mention that ontologies are helpful from a cooperative aspect. Suppose a new vulnerability for a particular device or some ontology classes is added. This new information can be seamlessly incorporated into the ontology, and new inferences can be made. This is also important when it comes to threat intelligence cooperation. The field is ever-evolving, and it can be critical to have the most recent information about threat actors and their operations.

8.9 Disadvantages of Ontologies in Cyber Security Operations

There are several drawbacks to using ontologies in cyber security, especially in industrial control system security. The main drawback of ontologies is that they do not account for time in any meaningful manner. When defending an industrial control system, how fast certain events occur can be valuable for the conduction of the defensive work.

Additionally, it is essential to mention that ontologies can be quite labor and process-intensive, especially in larger environments. The design of an ontology can be time-consuming for a person to perform, as it is necessary to include a massive amount of information about each of the individual components if one wants an appropriately effective ontology. As discussed in 8.7 the person or group developing an ontology for any practical purposes needs to have both good theoretical and practical knowledge. It can be worthwhile to use the ontology more abstractly, as is done in this thesis, to reduce the granularity of the ontology to only the information needed to express the relation between the different systems and components in play. This is because one can run the risk of having a lot of different classes on a model type level that does not contribute to any meaningful difference in the ontology but only increases the complexity and size of the ontology.

8.9.1 This Ontology Specifically

This ontology, in particular, is quite limited as it has been created over only one semester and by someone who is not a domain expert. Consequently, the ontology is far from exhaustive as there are many variations of industrial control systems, special cases, and cases that are not necessary for proving the concept of this ontology but can have practical significance. The ontology is also not particularly nuanced or granular as this is a deliberate choice to spend time effectively, but it would undoubtedly yield valuable results.

It is difficult to quantify the completeness of the ontology as one could spend much time filling in different models and their respective functions. However, the ontology is complete enough to prove that it is possible to determine the state and safety of a process based on threat actor actions and other system parameters such as the vulnerability of specific vital components.

Chapter 9

Conclusion

9.1 Summary of findings

9.1.1 How can ontologies be used to model Industrial Control Systems?

Ontologies are well suited to model industrial control systems on an abstract level by modeling the different classes of elements in an ICS, their hierarchies, and how they all relate. This thesis demonstrated an experimental approach for modeling ICSs. The author used this model as a foundation for all the work performed in this project, and it served its function perfectly.

9.1.2 How can ontology be used to identify system preconditions for attacks against ICS?

In this paper, the author has demonstrated that ontologies can determine if a component of an industrial control system is vulnerable given a set of states either coming from the component itself or other connected components. This thesis's sets of identifiable preconditions are limited and can be expanded upon to increase ontological coverage.

9.1.3 Can ontologies further infer system states based on discovered attack preconditions?

This thesis demonstrates how it is possible to determine the safety state of an industrial control system. One determines the safety state of an ICS based on which and how many of the safety layers can normalize the system's process. The author has identified four states that describe the safety of an ICS: Safe, weakened, eruptive, and fubar. The Safe state is the ideal state where nothing is wrong. The Weakened state occurs when either the process control, operator intervention, or both safety layers are disabled. The Eruptive state is declared when the safety instrumented system is no longer functioning, and without efficient operator intervention, the process will erupt, causing physical harm. The last state, Fubar, is when all of these safety layers are not functioning, and there are no digital safety elements to stop the process from running haywire.

9.1.4 How can threat intelligence enhance the effectiveness of an attack?

precondition ontology? In this thesis, the author demonstrates that it is possible to map security events on the network and other essential security data to a threat intelligence model that can determine important information about a threat actor. This information can be: inferred tactical or strategic goal or, ideally, the attacker's identity.

9.1.5 What is the benefit of a system state-oriented ontology?

The author demonstrates throughout this paper how modeling the state of the system allows for a much better understanding of the different risks that the ICS faces. System vulnerabilities are put into context, and the model infers their effect on system safety. This effect inference will allow system defenders to prioritize security work better. Combining this with a threat intelligence model allows attackers to have a high degree of situational awareness in any scenario.

9.2 Future Work

9.2.1 Increase Ontology Population

For the ontology to better perform its function, it is crucial to add more classes, object properties, and individuals. There are multiple elements of the ontology that this applies. Firstly, the ontological part regarding industrial control systems could be fleshed out by adding more specific classes and components. In this paper, the classes of industrial control system components that were added were only the ones necessary to either: prove a point when it came to the function of the ontology, or show the abstract function of the ontology.

9.2.2 Further Work on Threat Intelligence Aspect

The Limited scope of this thesis made it so that the threat intelligence part of this ontology was not as well defined as it could be. Threat intelligence works better the more information is available. This thesis's methodology was only tested on a limited experimental level. This would mean that to determine the actual efficiency of such an ontology, one would need to enter more information and test it more against existing scenarios. Ideally, these would be scenarios where the threat actor behind the operation is known, or there exist a proper amount of information that one can reasonably assume what threat actor is behind the attack.

It should also be entirely possible to make the ontology infer the threat actor behind an attack automatically based on attack information. The beauty of the ontology is that one does not necessarily need to observe every attack element to infer the threat actor behind an attack.

It could also be interesting to see if it is possible to infer attack type (disjoint from threat actor) based on network information and artifacts.

Bibliography

[1] IEEE, *Ieee taxonomy - created by the institute of electrical and electronics ...* 2022. [Online]. Available: https://www.ieee.org/content/dam/ieee-org/ieee/web/org/pubs/ieee-taxonomy.pdf.

[2] T. Sauter, S. Soucek, W. Kastner and D. Dietrich, 'The evolution of factory and building automation,' *IEEE Industrial Electronics Magazine*, vol. 5, no. 3, pp. 35–48, 2011. DOI: 10.1109/MIE.2011.942175.

[3] P. Gonzalez-Gil, J. A. Martinez and A. F. Skarmeta, 'Lightweight data-security ontology for iot,' eng, *Sensors (Basel, Switzerland)*, vol. 20, no. 3, p. 801, 2020, ISSN: 1424-8220.

[4] C.-K. Wu, 'On the it security and the ot security in iot,' eng, in *Internet of Things Security*, ser. Advances in Computer Science and Technology, Singapore: Springer Singapore, 2021, pp. 171–198, ISBN: 9811613710.

[5] 'Sans/nozomi: Ot/ics cyber security,' eng, *Computer fraud & security*, vol. 2021, no. 9, pp. 4–4, 2021, ISSN: 1361-3723.

[6] TrendMicro, 2022. [Online]. Available: https://www.trendmicro.com/vinfo/us/security/definition/industrial-control-system.

[7] Safeopedia, *What are safety instrumented systems (sis)? - definition from safeopedia*, Oct. 2017. [Online]. Available: https://www.safeopedia.com/definition/5011/safety-instrumented-systems-sis.

[8] P. Loshin, *What is scada (supervisory control and data acquisition)?* Dec. 2021. [Online]. Available: https://www.techtarget.com/whatis/definition/SCADA-supervisory-control-and-data-acquisition.

[9] C. Station, *What is a distributed control system?* Nov. 2021. [Online]. Available: https://controlstation.com/blog/what-is-a-distributed-control-system/.

[10] Kaspersky, *What is threat intelligence? definition and explanation*, Apr. 2022. [Online]. Available: https://www.kaspersky.com/resource-center/definitions/threat-intelligence.

[11] L. F. Sikos, *Description logics in multimedia reasoning*. Springer International Publishing, 2018.

[12] T. Gruber, *Ontology*, 2008. [Online]. Available: http://web.dfc.unibo.it/buzzetti/IUcorso2007-08/mdidattici/ontology-definition-2007.htm#:~:text=In%5C%20the%5C%20context%5C%20of%5C%20computer,or%5C%20relations%5C%20among%5C%20class%5C%20members)..

[13] K. H. Rosen, *Discrete mathematics and its applications*, eng, New York, 2019.

[14] A. Creery and E. Byres, 'Industrial cybersecurity for power system and scada networks,' in *Record of Conference Papers Industry Applications Society 52nd Annual Petroleum and Chemical Industry Conference*, 2005, pp. 303–309. DOI: 10.1109/PCICON.2005.1524567.

[15] T. Heverin, A. Chandnani, C. Lopex and N. Brahmhatt, *Ontology modelling of industrial control system ethical hacking*, English, Copyright - Copyright Academic Conferences International Limited Feb 2021; Last updated - 2021-03-27, Feb. 2021. [Online]. Available: https://www.proquest.com/conference-papers-proceedings/ontology-modelling-industrial-control-system/docview/2505729706/se-2?accountid=12870.

[16] J. Alanen, J. Linnosmaa, T. Malm, N. Papakonstantinou, T. Ahonen, E. Heikkilä and R. Tiusanen, 'Hybrid ontology for safety, security, and dependability risk assessments and security threat analysis (sta) method for industrial control systems,' *Reliability Engineering & System Safety*, vol. 220, p. 108 270, 2022, ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2021.108270. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832021007444.

[17] W. Xiong, E. Legrand, O. Åberg and R. Lagerström, 'Cyber security threat modeling based on the mitre enterprise att&ck matrix,' eng, vol. 21, no. 1, pp. 157–177, 2021, ISSN: 1619-1366.

[18] V. Mavroeidis and S. Bromander, 'Cyber threat intelligence model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence,' in *2017 European Intelligence and Security Informatics Conference (EISIC)*, 2017, pp. 91–98. DOI: 10.1109/EISIC.2017.20.

[19] *Protegedesktopuserdocs*, Dec. 2021. [Online]. Available: https://protegewiki.stanford.edu/wiki/ProtegeDesktopUserDocs#Editor_features.

[20] N. F. Noy and D. L. McGuinness, 'Ontology development 101: A guide to creating your first ontology,' Tech. Rep., Mar. 2001. [Online]. Available: http://www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness-abstract.html.

[21] M. Uschold and M. Grüninger, 'Ontologies: Principles, methods and applications,' *The Knowledge Engineering Review*, vol. 11, Jan. 1996.

[22] M. J. Assante and R. M. Lee, 'The industrial control system cyber kill chain,' *SANS Institute InfoSec Reading Room*, vol. 1, 2015.

[23] A. L. Johnson, *Endpoint protection*, Jun. 2014. [Online]. Available: https://community.broadcom.com/symantecenterprise/communities/community-home/librarydocuments/viewdocument?DocumentKey=7382dce7-0260-4782-84cc-890971ed3f17%5C&amp;CommunityKey=1ecf5f55-9545-44d6-b0f4-4e4a7f5f5e68%5C&amp;tab=librarydocuments.

[24] Kaspersky, *Crouching yeti (energetic bear) malware*, Jan. 2021. [Online]. Available: https://www.kaspersky.com/resource-center/threats/crouching-yeti-energetic-bear-malware-threat.

[25] E. M. Hutchins, M. J. Cloppert, R. M. Amin *et al.*, 'Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains,' *Leading Issues in Information Warfare & Security Research*, vol. 1, no. 1, p. 80, 2011.

[26] P. De Vaere, C. Hähni, F. Monti and A. Perrig, 'Tableau: Future-proof zoning for ot networks,' in *Critical Information Infrastructures Security*, D. Percia David, A. Mermoud and T. Maillart, Eds., Cham: Springer International Publishing, 2021, pp. 207–227, ISBN: 978-3-030-93200-8.

[27] K. Scarfone and P. Hoffman, Sep. 2009. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-41r1.pdf.

[28] *Ics alert (ics-alert-14-281-01e)*, Dec. 2014. [Online]. Available: https://www.cisa.gov/uscert/ics/alerts/ICS-ALERT-14-281-01B.

[29] C. T. U. R. Team, *Resurgent iron liberty targeting energy sector*, Jun. 2019. [Online]. Available: https://www.secureworks.com/research/resurgent-iron-liberty-targeting-energy-sector.

[30] O. Andreeva, S. Gordeychik, G. Gritsai, O. Kochetova, E. Potseluevskaya, S. I. Sidorov and A. A. Timorin, *Industrial control systems and their online availability*, Jul. 2016. [Online]. Available: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2016/07/07190427/KL_REPORT_ICS_Availability_Statistics.pdf.

[31] T. M. Fernández-Caramés and P. Fraga-Lamas, 'Teaching and learning iot cybersecurity and vulnerability assessment with shodan through practical use cases,' eng, *Sensors (Basel, Switzerland)*, vol. 20, no. 11, p. 3048, 2020, ISSN: 1424-8220.

[32] Ibm, *Ibm security x-force threat intelligence index*, Feb. 2022. [Online]. Available: https://www.ibm.com/security/data-breach/threat-intelligence/.

[33] I. Foulds, *Attractive accounts for credential theft*. [Online]. Available: https://docs.microsoft.com/nb-no/windows-server/identity/ad-ds/plan/security-best-practices/attractive-accounts-for-credential-theft.

[34] D. Simpson, *Active directory accounts (windows 10) - windows security*, Mar. 2021. [Online]. Available: https://docs.microsoft.com/en-us/windows/security/identity-protection/access-control/active-directory-accounts.

[35] *Advisory: Apt29 targets covid-19 vaccine development - ncsc.gov.uk*, Jul. 2020. [Online]. Available: https://www.ncsc.gov.uk/files/Advisory-APT29-targets-COVID-19-vaccine-development.pdf.

[36] N. Falliere, *W32.stuxnet dossier - wired*, Nov. 2010. [Online]. Available: https://www.wired.com/images_blogs/threatlevel/2010/11/w32_stuxnet_dossier.pdf.

[37] A. Kling, *One year after triton: Building ongoing, industry-wide cyber resilience*, Jul. 2020. [Online]. Available: https://blog.se.com/machine-and-process-management/2018/08/07/one-year-after-triton-building-ongoing-industry-wide-cyber-resilience/.

[38] A. Kling and P. Forney, *Triton - schneider electric analysis and disclosure*, Jan. 2018. [Online]. Available: https://www.youtube.com/watch?v=f09E75bWvkk%5C&amp;list=PL8OWO1qWXF4qYG19p7An4Vw3N2YZ86aRS%5C&amp;index=4.

[39] Ivensys, *Technical product guide tricon systems*, Aug. 2006. [Online]. Available: https://www.nrc.gov/docs/ML0932/ML093290424.pdf.

[40] D. J. Smith and K. G. Simpson, *The Safety Critical Systems Handbook: A Straightforward Guide to Functional Safety: IEC 61508 (2010 Edition), IEC 61511 (2015 Edition) and Related Guidance*. Butterworth-Heinemann, 2020.

[41] Y. Redutskiy, 'Modelling and design of safety instrumented systems for upstream processes of petroleum sector,' *Procedia Engineering*, vol. 182, pp. 611–618, 2017, 7th International Conference on Engineering, Project, and Production Management, ISSN: 1877-7058. DOI: https://doi.org/10.1016/j.proeng.2017.03.165. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877705817313012.

[42] T. Macaulay, G. Brown and P. Schneck, *RIoT Control: Understanding and Managing Risks and the Internet of Things*, eng. San Francisco: Elsevier Science & Technology, 2016, pp. 249–254, ISBN: 9780124199712.

[43] S. Wheatley, B. K. Sovacool and D. Sornette, 'Reassessing the safety of nuclear power,' *Energy Research & Social Science*, vol. 15, pp. 96–100, 2016, ISSN: 2214-6296. DOI: https://doi.org/10.1016/j.erss.2015.12.026. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214629615301067.

[44] *Cyber threat source descriptions*. [Online]. Available: https://www.cisa.gov/uscert/ics/content/cyber-threat-source-descriptions.

[45] A. Fishbach and M. J. Ferguson, 'The goal construct in social psychology.,' 2007.

[46] J. Don, *Lessons learned from a forensic analysis of the ukrainian power grid cyberattack*. [Online]. Available: `https://blog.isa.org/lessons-learned-forensic-analysis-ukrainian-power-grid-cyberattack-malware`.

[47] M. Giles, *Triton is the world's most murderous malware, and it's spreading*, Apr. 2020. [Online]. Available: `https://www.technologyreview.com/2019/03/05/103328/cybersecurity-critical-infrastructure-triton-malware/`.

[48] B. Johnson, *Attackers deploy new ics attack framework "triton" and cause operational disruption to critical infrastructure*, Sep. 2022. [Online]. Available: `https://www.mandiant.com/resources/attackers-deploy-new-ics-attack-framework-triton`.

[49] BlackHatOfficialYT, *How triton disrupted safety systems &amp; changed the threat landscape of industrial control systems*, Jan. 2020. [Online]. Available: `https://www.youtube.com/watch?v=Hw2HclZV2Kw%5C&amp;t=555s`.

NTNU

Norwegian University of
Science and Technology