UNIVERSITY OF SAAD DAHLEB BLIDA

Faculty of Sciences

Departement of Computer Science

THESIS OF MASTER

In Computer Science

Option : Information System Security

THEME :

Adversarial attacks detection based on an ontology of cyber threat

Realized by

YKRELEF Imad Eddine

YAHIAOUI Abdel Madjid

Supervisor and members of the jury

Mme.Soraya CHACHOUA

Mme.Hadjer YKHLEF

Mme.Yasmina GHEBGHOUB

July 2023

Acknowledgments

First of all, we would like to thank GOD who gave us the necessary will, the strength and good health, the patience and the courage to complete this work.

We express our gratitude to our promoter and supervisor Madam the Doctor Soraya CHACHOUA who did us the honor of directing this work. We thank her for her availability and her advices which were of considerable help to us both on the scientific aspect and on the research method.

We also thank the members of the jury for their interest in our research by agreeing to examine our work and enrich it with their proposals.

Finally, we thank all those who have contributed directly or indirectly to the realization of this research work.

Abstract

Machine learning has been used in the field of cybersecurity to predict trends in cyberattacks. However, adversaries can inject malicious data into the dataset during training and testing to cause disruption and predict false narratives.
It has become difficult to analyze and predict correlations of cyberattacks due to their fuzzy nature and lack of understanding of the nature of threats.
We use our model to create a cyber threat ontology and use its rules to detect adversarial machine learning attacks.

Keywords: Cyber security, cyber attacks, cyber defense, machine learning, adversary attacks, cyber threat ontology.

Résumé

L'apprentissage automatique a été utilisé dans le domaine de la cybersécurité pour prédire les tendances des cyberattaques. Cependant, les adversaires peuvent injecter des données malveillantes dans l'ensemble de données pendant la formation et les tests pour provoquer des perturbations et prédire de faux récits.
Il est devenu difficile d'analyser et de prévoir les corrélations entre les cyberattaques en raison de leur nature floue et du manque de compréhension de la nature des menaces.
Nous utilisons notre modèle pour créer une ontologie de cybermenace et utilisons ses règles pour détecter les attaques adverses d'apprentissage automatique.

Mots clés: Cybersécurité, cyberattaques, cyberdéfense, apprentissage automatique, attaques adverses, ontologie des cybermenaces.

ملخص

تم استخدام التعلم الآلي في مجال الأمن السيبراني للتنبؤ باتجاهات الهجمات السيبرانية. ومع ذلك، يمكن للخصوم حقن بيانات ضارة في مجموعة البيانات أثناء التدريب والاختبار لإحداث اضطراب والتنبؤ بالنتائج الكاذبة.

لقد أصبح من الصعب تحليل وتوقع الارتباطات بين الهجمات السيبرانية بسبب طبيعتها الغامضة وعدم فهم طبيعة التهديدات.

نحن نستخدم نموذجنا لإنشاء أنطولوجية التهديد السيبراني واستخدام قواعده للكشف عن هجمات التعلم الآلي العدائية.

الكلمات المفتاحية :

الأمن السيبراني، الهجمات السيبرانية، الدفاع السيبراني، التعلم الآلي، الهجمات العدائية، أنطولوجية التهديد السيبراني.

Acronyms

AML Adversarial Machine Learning

ANN Artificial Neural Network

APT Advanced Persistent Threat

BIM Basic Iterative Method

CTI Cyber Threat Intelligence

CTO Cyber Threat Ontology

CVE Common Vulnerabilities and Exposures

CVSS Common Vulnerability Scoring System

CW Carlini-Wagner Attack

DDoS Distributed Denial of Service

DNN Deep Neural Network

FGSM Fast Gradient Sign Method

GAN Generative Adversarial Network

GBoost Gradient Boosting

IDSs Intrusion Detection Systems

JSMA Jacobian-Based Saliency Map Attack

KNN K-Nearest Neighbor

ML Machine Learning

OWL Web Ontology Language

RAT Remote Access Trojan

RDF Resource Description Framework

RDFS Resource Description Framework Schema

RF Random Forest

SIEM Security Information and Event Management

SPARQL SPARQL Protocol and RDF Query Language

STIX Structured Threat Information eXpression

SVM Support Vector Machine

SWRL Semantic Web Rule Language

TLO Top-Level Ontology

URIs Uniform Resource Identifiers

W3 World Wide Web

XML Extensible Markup Language

ZOO Zeroth-Order Optimization

Contents

List of Figures

List of Tables

General introduction

Machine learning (ML) approaches are changing our perceptions of the world and play an important role in the age of technology. The ML domain is a subfield of artificial intelligence that focuses on training algorithms to make predictions or decisions based on data, these algorithms can be used in cybersecurity domain for cyber threat classification accuracies, anomaly detection, analyzing network traffic and threat predictions. However, adversaries are using Adversarial Machine Learning (AML) techniques to manipulate algorithms, cause perturbations and falsify the classifications models by inserting malicious input data in datasets during training and testing phases.

Many defensive strategies are applied to detect these attacks[3][15]. Nevertheless, it is still challenging to ensure that is no false narratives predicted by the models, which is important to search and find a new methods to solve this issue. Cyber Threat Ontology (CTO) is a structured framework that organizes and categorizes information about cyber threats in a way that makes it easier to understand and analyze, it is also a system of concepts, categories, and relationships that describe different types of cyber threats. The goal of the (CTO) is to extract attack instances and information from data to ensure accuracy and consistency in security concepts and for knowledge reuse in the threat intelligence domain.

In this research, we explain how the (CTO) detects adversarial machine learning attacks by providing a structured and standardized way of categorizing and analyzing different types of cyber threats which help the (ML) model to increase the accuracy and persistent of the classification during the training and testing phases.
The rest of the manuscript is structured as follows :
In Chapter 1, We define (AML) attacks and techniques used to manipulate algorithms and predict false narratives. we also discuss the defensive strategies implemented to detect adversarial attacks. Chapter 2 describe the hole concepts in terms of classes, properties and relationships between them of the (CTO) and how they help the cybersecurity professionals to identify and respond to threats more effectively. Furthermore, we specify how (CTO) can detect adversarial attacks. In Chapter 3, we will propose our contribution to use cyber threat ontology to detect adversarial attacks and demonstrate our designed conceptual model.

In Chapter 4 we implement our model and simulating the final results.

Finally, We will conclude our manuscript with a general conclusion and some future perspectives to improve this work.

Chapter 1

Adversarial Machine Learning Attacks

1.1 Introduction

In the digital world, where artificial intelligence is becoming increasingly sophisticated, cybersecurity is becoming equally complex. One of the more advanced techniques in this field is Adversarial Machine Learning Attacks. This cutting-edge technology can be used for good or nefarious purposes, and has piqued the interest of security professionals, researchers, and cybercriminals alike. In this chapter, we will delve into the world of Adversarial Machine Learning Attacks and examine how they work, how they can be detected, and how they can be defended against.

1.2 Machine Learning

1.2.1 Definition

Machine Learning is a branch of artificial intelligence that enables software applications to learn from data, identify patterns, make decisions and improve their performance without being explicitly programmed. It is based on statistical and mathematical algorithms, which enable the system to automatically adjust its processes and models in response to new data, making it an essential tool for data-driven decision-making in a variety of industries. At its core, machine learning is all about finding ways to take advantage of data to solve complex problems, and its applications are far-reaching, from image and speech recognition to recommendation systems and predictive maintenance.

1.2.2 Machine Learning Categories

In general, machine learning techniques can be divided into two categories.

1.2.2.1 Supervised Machine Learning

Supervised learning depends on meaningful information in labeled data. The most common goal in supervised learning is classification. Nevertheless, manually labeling data is costly and time-intensive. As a result, the fundamental barrier to supervised learning is the lack of adequate labeled data.

1.2.2.2 Unsupervised Learning

Unsupervised learning recovers useful feature information from unlabeled data, making training material much more straightforward. On the other hand, unsupervised learning approaches often perform worse in terms of detection than supervised learning methods.

Figure 1.3 shows the most prevalent ML techniques used in the field [17].

Figure 1.1: Types of Machine Learning Models.

Here are some differences between supervised and unsupervised learning :

Supervised Learning	**Unsupervised Learning**
Input data is labeled.	Input data is unlabeled.
Data is classified based on the training dataset.	Assigns properties of given data to classify it.
Divided into Classification and Regression.	Divided into Clustering and Association.
Used for prediction.	Used for analysis.
Algorithms include : decision trees, logistic regressions, support vector machine.	Algorithms include : k-means clustering, hierarchical clustering, apriori algorithm.
Number of classes is known.	Number of classes is unknown.

Table 1.1: Supervised vs Unsupervised Learning.

1.2.3 Machine Learning techniques

Various machine learning techniques have been used in the field, the following paragraphs summarize the most commonly used techniques.

1.2.3.1 Artificial Neural Network (ANN)

An ANN is designed to function in the same way as human brains. An ANN comprises numerous hidden layers, an input layer, and an output layer. Units in neighboring strata are interconnected. Furthermore, it has an excellent fitting ability, particularly for nonlinear functions.

1.2.3.2 Deep Neural Network (DNN)

The parameters of a DNN are initially learned using unlabeled data in an unsupervised feature learning stage, and then the network is tweaked using labeled data in a supervised learning stage. The unsupervised feature learning step is mainly responsible for DNN's remarkable performance. Furthermore, DNN plays a crucial role in cybersecurity therefore DNN could understand the abstract, high-level properties of APT assaults even if they use the most complex evasion strategies.

1.2.3.3 Generative Adversarial Network (GAN)

A GAN model has two subnetworks, one for the generator and one for the discriminator. The generator's goal is to create synthetic data that looks like actual data, whereas the discriminator's goal is to find the difference between synthetic and natural data. As a result, the generator and discriminator complement each other.

1.2.3.4 Support Vector Machine (SVM))

some key components of it are:

- **Hyperplane:** A hyperplane is an n-1 dimensional subspace in an n-dimensional space. In a 2D space, a hyperplane is a plane which represents a line within a 3D space. For a linearly separable dataset, the hyperplane is the optimal separator.
- **Margin and Support Vectors** The margin is the distance between the hyperplane and the closest data points from each class. These closest points are called support vectors.
- **The Kernel and Kernel Trick:** the kernel is a function that transforms data to a higher-dimensional space, enabling SVM to find non-linear decision boundaries While The kernel trick is a mathematical technique that allows SVM to efficiently handle non-linear problems by operating in the higher-dimensional space without explicitly calculating the transformation.

The goal is to locate a hyperplane of maximum margin separation in the n-dimensional feature space. Because a small number of support vectors control the separation hyperplane, these models can produce satisfactory results even with small-scale training data. on the other hand, they are susceptible to noise around the hyperplane. SVMs excel at solving linear problems and are rife with kernel trickery.

1.2.3.5 K-Nearest Neighbor (KNN)

KNN's fundamental principle is rooted in the manifold hypothesis, which suggests that if a sample's neighbors predominantly belong to a particular class, then there is a high likelihood that the sample belongs to that class as well. As a result, KNN's classification outcomes are dependent solely on the k-nearest neighbors. The choice of k significantly impacts the performance of KNN models. A smaller k leads to a more complex model and a greater risk of overfitting, while a larger k results in a simpler model with weaker fitting capabilities.

1.3 Adversarial Machine Learning

1.3.1 Definition

Adversarial Machine Learning is a field of research that focuses on studying the security and robustness of machine learning models against adversarial attacks. In this context, an adversarial attack is a deliberate attempt to manipulate or deceive a machine learning model by introducing carefully crafted inputs that are designed to cause the model to make incorrect predictions or decisions.
Figure 1.3 explains the whole process of AML [14].

Figure 1.2: The whole process of AML.

1.3.2 Adversarial Attacks

There are several approaches for creating adversarial samples, each varying in complexity, creation speed, and performance. One crude method involves manually perturbing input data points, but this can be time-consuming and imprecise when dealing with massive datasets. A more complex approach involves automatically assessing the characteristics that best differentiate between target values and discretely disturbing these characteristics to reflect values that differ from their own. Adversaries may have full or limited understanding of the machine learning system, which can impact the effectiveness of their attacks.

1.3.2.1 White-Box Attack

In cases where the machine learning model is open source and accessible to everyone, attackers often have a complete understanding of the network architecture and the training parameters. As a result, they can launch white-box attacks that autonomously create perturbed samples. Four of the most well-known white-box attack techniques for generating such samples include:

1. **Fast Gradient Sign Method (FGSM)** It works by computing the gradient of the loss function with respect to the input data and then perturbing the input data in the direction of the gradient. The magnitude of the perturbation is controlled by a hyperparameter called the "epsilon" value. The sign of the gradient is multiplied by the epsilon value to obtain the perturbation vector, which is added to the original input data to obtain the adversarial example.

2. **Jacobian-Based Saliency Map Attack (JSMA)** It works by computing the gradient of the network's output with respect to the input and identifying the most influential input features. The algorithm then iteratively modifies these features to generate the adversarial example. The process involves computing the Jacobian matrix, which captures how each output class changes with respect to each input feature. By calculating a saliency map from the Jacobian matrix, the importance of each input feature in determining the target class's probability is quantified. The features with the highest saliency values are perturbed. This iterative process continues until the desired misclassification or a perturbation budget is achieved. The approach aims to minimize noticeable changes to the input while maximizing the likelihood of deceiving the model.

3. **Carlini-Wagner Attack (CW)** It works by optimizing a set of parameters that modify the input in order to maximize the difference between the model's correct output and the desired output. This optimization is performed using a gradient descent algorithm, which iteratively adjusts the parameters to minimize the distance between the original input and the modified input, subject to the constraint that the modified input causes the model to produce the desired output.

4. **Basic Iterative Method (BIM)** This approach performs gradient calculations in small increments, building on the FGSM attack. To preserve the traffic characteristics of the input and prevent substantial changes, the perturbation is constrained.

1.3.2.2 Black-Box Attack

This attack strategy involves assuming no prior knowledge of the target system and analyzing its vulnerability using information from its settings or previous inputs. To learn more about the classification algorithm, the attacker employs two techniques.
Firstly, they may repeatedly modify malicious samples until they are misclassified, in order to identify the model's parameters for distinguishing between malicious and benign samples.
Additionally, the attacker may create a substitute model of the detection system and exploit the transferability of machine learning to generate adversarial samples that deceive both the substitute classifier and the actual detector.
black-box attacks include

1. **Zeroth-Order Optimization (ZOO)** ZOO does not directly calculate the gradient, but instead employs the symmetric difference quotient method to estimate it, which incurs a higher computational cost. This technique does not require any knowledge of the DNN network's structure for gradient estimation.

2. **OnePixel** This attack can manipulate a DNN without comprehending its network topology by altering the value of a single pixel in an otherwise unaltered image. The DNN can be susceptible to attacks of very low dimensionality with minimal information.

1.3.2.3 Gray-Box Attack

The attacker employs an iterative learning process that utilizes inference techniques to gain further understanding of the model and transition from a black box to a white box setting. As a result, the attacker may possess some partial knowledge of the model.
In situations where knowledge is limited, such as in gray-box and black-box scenarios, the attacker may use privacy attacks to acquire additional information about the targeted ML classifier.

Figure 1.3 demonstrates adversarial attacks that depend on the level of knowledge [3].

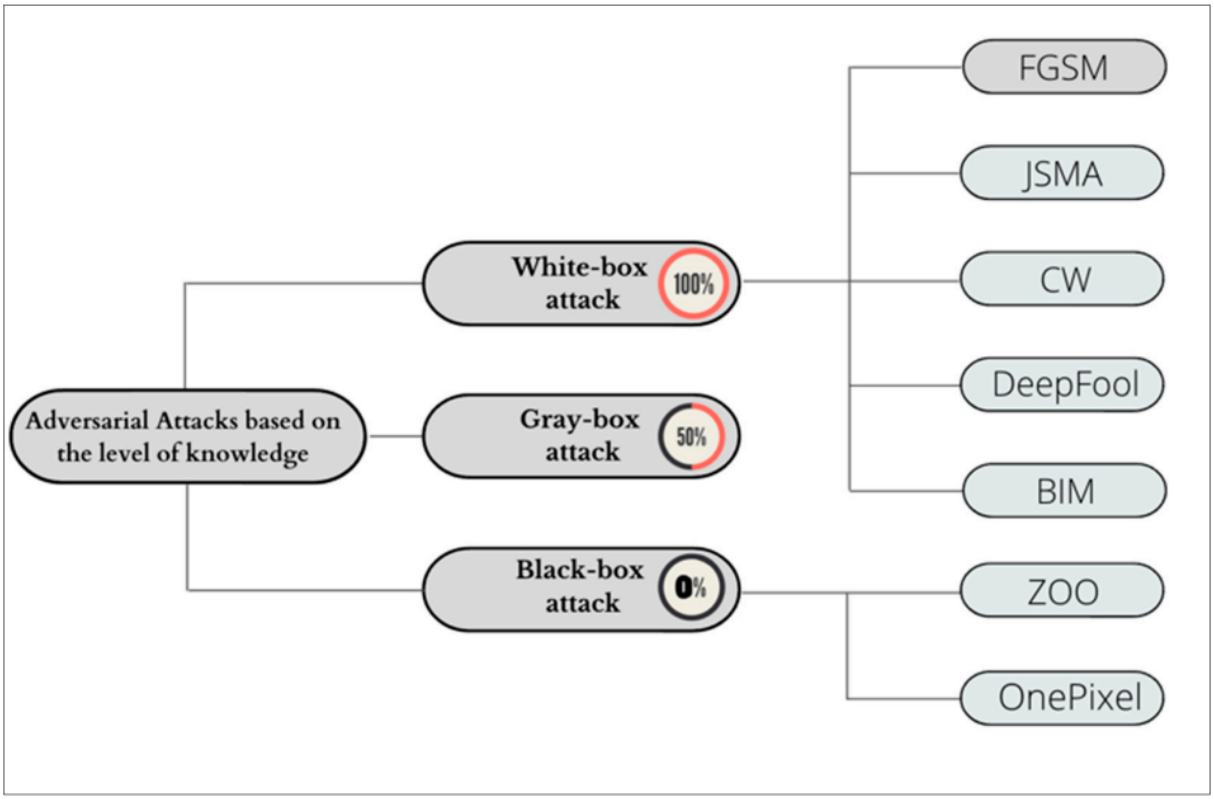

Figure 1.3: Adversarial attacks in correlation with level of knowledge

1.3.3 Adversarial Attacks Categories: Evasion vs Poisoning

1.3.3.1 Evasion Attack

Evade a system by inserting adversarial samples that does not effect the training data. The goal is to cause the model to incorrectly classify malware samples as benign during its operation.
Evasion attacks can be divided into two categories:

1. **Error-Generic Evasion Attacks** where the attacker aims to deceive the classification regardless of the predicted output class;
2. **Error-Specific Evasion Attacks** where the attacker intends to misclassify the adversarial samples as a particular class while deceiving the classification.

1.3.3.2 Poisoning Attack

Is an attempt by an adversary to pollute the training data by introducing carefully planned samples, which can compromise the learning process.
Poisoning attacks can be categorized into two types:

1. **Error-Generic Poisoning Attacks** where the attacker aims to cause a denial of service by inducing as many classification errors as possible;
2. **Error-Specific Poisoning Attacks** where the attacker's goal is to produce specific misclassifications to further their objective.

Figure 1.4 showcases these two attacks and their effect on machine learning [6].

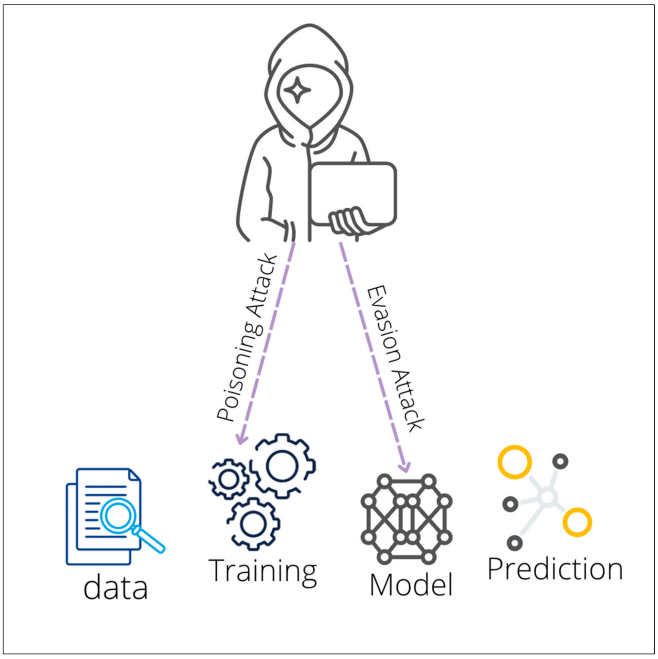

Figure 1.4: Evasion vs Poisoning attack

Here are some differences between these two types of attacks :

	Evasion Attacks	**Poisoning Attacks**
Objective	Manipulate input data to mislead the model's prediction.	Inject malicious data to compromise model performance.
Attack Stage	At inference time.	During the training phase.
Target	Exploits vulnerabilities in deployed models.	Targets the training process and model parameters.
Intent	Evade detection or misclassify specific instances.	Introduce backdoors or degrade model performance.
Techniques	Adversarial examples, input perturbations.	Injecting poisoned data, data poisoning algorithms.
Impact on Mode	May reduce model accuracy or bypass security measures.	Alters model behavior, compromises generalization.
Detection and Mitigation	Adversarial defense mechanisms, robust models.	Data sanitization techniques, robust training methods.
Required Knowledge	Knowledge of the model architecture and attack methods.	Knowledge of the training process and data manipulation.

Table 1.2: Evasion attacks vs Poisoning attacks.

1.4 Related Work

Table 1.3 provides an overview of the existing literature and related works in adversarial machine learning.

Ref	**Year**	**Highlights**
[4]	2023	discusses the use of machine learning in intrusion detection systems (IDSs) and the threat of adversarial machine learning (AML) in compromising their effectiveness. It provides an overview of AML attacks and defense strategies to mitigate their impact, with suggestions for future research.
[24]	2015	Evaluates the security of Support Vector Machines (SVMs) against adversarial label noise attacks and reports an extensive experimental analysis on their effectiveness. The authors argue that their approach can provide useful insights for developing more secure SVM learning algorithms and novel techniques in related research areas.
[15]	2019	Discusses the threat of adversarial attacks faced by machine learning applications in network security and analyzes their defenses. It introduces a classification of machine learning in network security applications and examines various adversarial attacks against them
[17]	2019	Introduces common machine learning algorithms used in IDSs, explains how they can solve key issues, and discusses challenges and future developments in the field.

Table 1.3: adversarial ML related works.

Recent advancements in adversarial machine learning have brought about groundbreaking progress in understanding, defending against, and mitigating adversarial attacks. researchers are actively developing robust solutions to safeguard machine learning systems from adversarial manipulation. These achievements highlight the ever-evolving nature of the field :

1. Generative Adversarial Networks (GANs): Proposed in 2014 GANs consist of a generator and a discriminator network that compete against each other. GANs have been leveraged to generate realistic and targeted adversarial examples, enabling more sophisticated and challenging attacks[11].

2. Adversarial Examples in the Physical World: Researchers in 2017 demonstrated that adversarial examples can be created in the physical world by adding imperceptible perturbations to objects or images. These physical adversarial attacks pose security concerns in real-world scenarios[9].

3. Certified Defenses: The development of certified defenses, which provide mathematical guarantees on the model's robustness, gained traction in recent years. These defenses offer formal bounds on the model's performance and provide reliable protection against adversarial attacks[23].

4. Adversarial Transfer Learning: Understanding the transferability of adversarial examples between different models, tasks, or domains has become a significant research focus. This exploration aims to develop more robust defenses that can withstand adversarial attacks in various scenarios[13].

5. Privacy-Preserving Adversarial Learning: The intersection of adversarial machine learning and privacy protection has gained importance. Researchers are developing techniques that protect the privacy of individual data samples or model parameters during the training process, ensuring robustness against adversarial attacks[1].

1.5 Conclusion

In conclusion, the rise of adversarial machine learning attacks is a cause of concern for the robustness and reliability of machine learning models. Attackers can exploit the vulnerabilities of these models to manipulate the outcomes in malicious ways.
In this Chapter, we have defined the machine learning field, categories, techniques and adversarial machine learning attacks. In addition, we have defined the various AML attacks types and the difference between them. At the end, we explain the different related works which they discuss about the impact of these attacks on machine learning models.
There are various techniques developed to detect and mitigate these attacks, such as adversarial training, input sanitization, and model diversification. However, these techniques are not foolproof, and there is still room for further research in this area.

Chapter 2

Cyber Threat Ontology

2.1 Introduction

In the ever-evolving landscape of cybersecurity, Cyber Threat Ontology (CTO) is one of the most promising new approaches to understanding and managing risks posed by online threats such as malware, phishing, DDoS attacks, and other forms. This groundbreaking method involves the use of an organized and standardized set of concepts, relationships, and definitions to categorize and analyze cyber threats to develop effective strategies and mitigate the risks of these attacks. It is an essential tool for cybersecurity professionals to identify and respond to threats in a timely and effective manner.

2.2 Ontologies

2.2.1 Definition

Ontology is a formal, structured representation of knowledge that describes the concepts and categories within a particular domain, as well as the relationships between them.
It is a powerful tool for organizing and managing information, and is widely used in many different fields including cybersecurity.
It provides a way to represent knowledge in a machine-readable format, making it possible to use automated reasoning and analysis to draw conclusions.

2.2.2 Types of Ontologies

There are several types of ontologies, the most important in our context are defined and illustrated in Figure 2.1.

2.2.2.1 Top-Level Ontology

A top-level ontology (TLO) is a type of ontology that defines a set of general, high-level concepts and relationships that are common across different domains. It provides a framework for organizing and categorizing other ontologies, and serves as a foundation for developing more specific ontologies. It contains objects and not structures and cannot be instantiated and specializes. The most common examples are **DOLCE** [5], **SUMO** [2], etc.

2.2.2.2 Core Ontology

It is typically more specific than a top-level ontology, but more general than a domain-specific ontology, this type of ontology provides structuring concepts of the domain and the relationships between these concepts like **Mikrokosmos** [22]. For example, a core level ontology for the healthcare domain might include concepts such as patient, diagnosis, treatment, and medication.

2.2.2.3 Domain Ontology

Domain ontology called also Operational is designed to capture the key concepts and relationships that are specific to a particular domain or subject area, which are handled by domain professionals, such as descriptive ontology which is semantically rich such as **TOVE** [10] ontology.

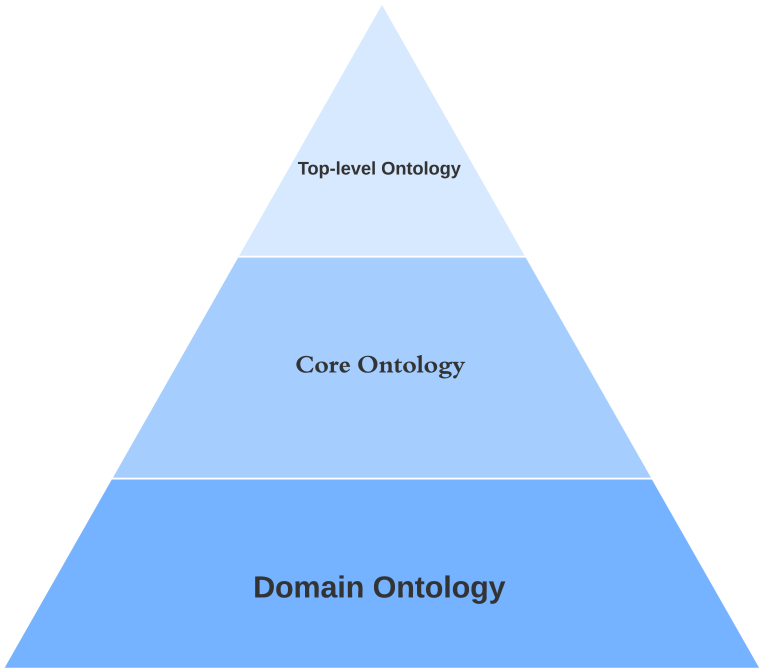

Figure 2.1: Types of Ontologies.

2.2.3 Structure of Ontology

The structure of the ontology typically consists of three mains components:

1. **Classes or concepts:** These are categories or types of things that the ontology defines. For example, "person", "place", "event", "object", etc.
2. **Properties or attributes:** These are characteristics of the classes or concepts. For example, for the class "person", attributes could include "name", "age", "gender", "occupation", etc.
3. **Relationships or links:** These are the connections between the classes or concepts. For example, the relationship between a "person" and a "place" could be that the person "lives in" the place.
4. **Instances:** Instances are specific examples of classes within the domain. For example, "diabetes" would be an instance of the class "disease" in a medical ontology.
5. **Axioms:** Axioms are statements that define the logical relationships between classes and properties. For example, an axiom may state that if a medication "treats" a disease, then the medication should not cause the disease.
6. **Constraints:** Constraints are rules that restrict how classes and properties can be used within the ontology. For example, a constraint may limit the number of instances that can be associated with a particular property.

Overall, the structure of the ontology aims to provide a clear and organized representation of knowledge in a specific domain, allowing for better understanding, organization, and sharing of information.

2.2.4 Ontology and Semantic Web

Ontology and Semantic Web are closely related concepts that are often used together to enable the sharing and interoperability of data and knowledge on the web.
The Semantic Web is a vision of the future of the World Wide Web (W3), in which information is structured in a way that is machine-readable. The idea behind the Semantic Web is to enable machines to understand the meaning and context of information on the web, allowing them to perform more sophisticated tasks and processes. Semantic Web technologies include **RDF, OWL, and SPARQL**, among others which we will discuss in the next section.
This enables the development of intelligent applications that can understand and process information on the web in a more automated and intelligent way, leading to more efficient and effective use of data and knowledge.

2.2.5 Ontology Representation Languages

2.2.5.1 RDF

RDF (Resource Description Framework) extends the linking structure of the Web to use URIs (Uniform Resource Identifiers) to name the relationship between things by using a simple triple

format **(subject, predicate, object)**, which is also known as a "RDF triple".

2.2.5.2 RDFS

RDFS (RDF Schema) is a lightweight ontology language that provides a basic vocabulary for describing classes, properties, and relationships within an RDF graph. RDFS provides basic elements for the vocabularies and definition of ontologies intended to structure RDF resources.

2.2.5.3 OWL

OWL (Web Ontology Language) is a semantic markup language used to represent knowledge on the web. OWL is based on the XML syntax and is designed to support complex modeling of knowledge, with features such as classes, properties, individuals, and rules. It is widely used in various domains, including bioinformatics, e-commerce, and intelligent systems. OWL is an important tool for managing, sharing, and reasoning about large-scale knowledge bases.

2.3 Cyber Threats

A cyber threat refers to a potential danger or risk that arises from the use of digital technology and computer networks. It includes any attempt to harm or damage computer systems, networks, or devices, steal or compromise sensitive information, distribute malware or viruses, and disrupt businesses or critical infrastructure. Some common types of cyber threats include:

1. **Malware:** Malicious software that is designed to harm or disrupt computer systems.
2. **Ransomware:** A type of malware that encrypts files on a victim's computer and demands payment in exchange for the decryption key.
3. **Phishing:** Social engineering attacks that trick users into divulging sensitive information such as passwords, credit card numbers, or social security numbers.
4. **Man-in-the-middle attacks:** Intercepting communications between two parties to steal or modify information.
5. **DDoS attacks:** Distributed Denial of Service attacks that flood a target website or network with traffic to disrupt its normal functioning.

Protecting against cyber threats is important in order to prevent data breaches, financial losses, or reputational damage. It is important to implement cybersecurity measures such as firewalls, antivirus software, strong passwords, and regular system updates.

2.4 Ontology in Cyberseucrity

A cyber threat ontology is a framework that is designed to categorize, model and represent knowledge about cyber threats. It provides a structured way of organizing information related to cyber threats, including different types of attacks, their methodologies, motivations, and consequences. The aim of a cyber threat ontology is to bring clarity to the complex world of cybersecurity, making it easier for security analysts to identify and respond to threats.

Creating a cyber threat ontology (CTO) can be a complex process that requires a good understanding of cybersecurity, threat intelligence, and ontology modeling. Here are some general steps to follow when creating a cyber threat ontology :

1. **Identify the scope and purpose of the ontology:** Determine the specific use case for the ontology, and identify the broad categories of threats that it will cover.

2. **Gather information about threats:** Collect data from various sources, such as threat intelligence feeds, incident reports, and security research publications. Analyze the data to identify common patterns and characteristics of different types of threats.

3. **Define the categories and relationships:** Use the information gathered in step 2 to define a hierarchy of categories and subcategories for the ontology. Determine the relationships between the categories, such as which threats are commonly associated with specific attack methods, vulnerabilities or targets.

4. **Define the properties:** Define the attributes of each category and subcategory of the ontology. These can include properties such as the tools, techniques, and procedures used by attackers, their motivations, and potential impact, among others.

5. **Develop the ontology using an ontology language:** Use an ontology modeling language such as OWL or RDF to represent the ontology as a machine-readable semantic network.

6. **Evaluate and refine the ontology:** Test the ontology using various use cases and refine it based on feedback from domain experts, software tools, and real-world events.

7. **Maintain and update the ontology:** Keep the ontology up to date by continuously updating it with new information, integrating new sources of threat intelligence, and adapting it to changes in the threat landscape.

2.5 Related works

This overview summarizes the current body of research and relevant studies in the field of Cyber Threat Ontology, as presented in the existing literature of Table 2.1.

Ref	Year	Highlights
[19]	2018	provide a unified technical framework to monitor business processes and technology assets using an ontology and knowledge reasoning for the IoT cybersecurity domain.
[16]	2018	build an ontology for cybersecurity that is based on vulnerability, and puts forward a method to build a cybersecurity knowledge base.
[8]	2019	propose an ontology of metrics for cyber security assessment, it is based on determining the concepts and relations between primary features of initial security data and forming a set of hierarchically interconnected security metrics.
[18]	2021	propose an ontology using CTI for risk monitoring, which improves an existing ontology, originally proposed to be used within a SIEM (Security Information Event Management), by extending it and aligning it with the STIX concepts.
[12]	2022	define an extended cybersecurity ontology, which may be used to assist in targeted information gathering and risk assessment procedures applied on complex cyber-physical systems.
[21]	2023	present an ontology to describe different types of anomalies, merged with previously developed models for Cyber-Threat Intelligence.

Table 2.1: CTO related works.

2.6 Adversarial attacks and Cyber threat ontology

The field of adversarial attacks and defense is constantly evolving, with new attack techniques and defense strategies being developed all the time. As a CTO, it is important to stay up-to-date with the latest research and developments in this field in order to effectively protect organizations from potential attacks.
As of the date of making this research not much work has been done combining this two. one of the researches we found [25] considered CTO concepts for knowledge representation and the AML attack approach to predict false narrative. Therefore, the main rationale for using this method is:

1. Firstly, they modeled CTO for advanced persistent threat (APT) attacks for threat mapping and the properties to determine the causal relationships for knowledge representation.

2. Secondly, they applied AML attack on the dataset through malicious insertion to predict false narratives cyberattacks.

Starting with their CTO model implementation it considers a conceptual model to identify and map the concepts that drive the required entities, properties relationships and rules sets for the cyberattack domain.
The concepts include infiltration, manipulation, exfiltration, and obfuscation as well as the properties that provide the conceptual reasoning, relational knowledge and understanding of cyber threat intelligence required as shown in Figure 2.2.

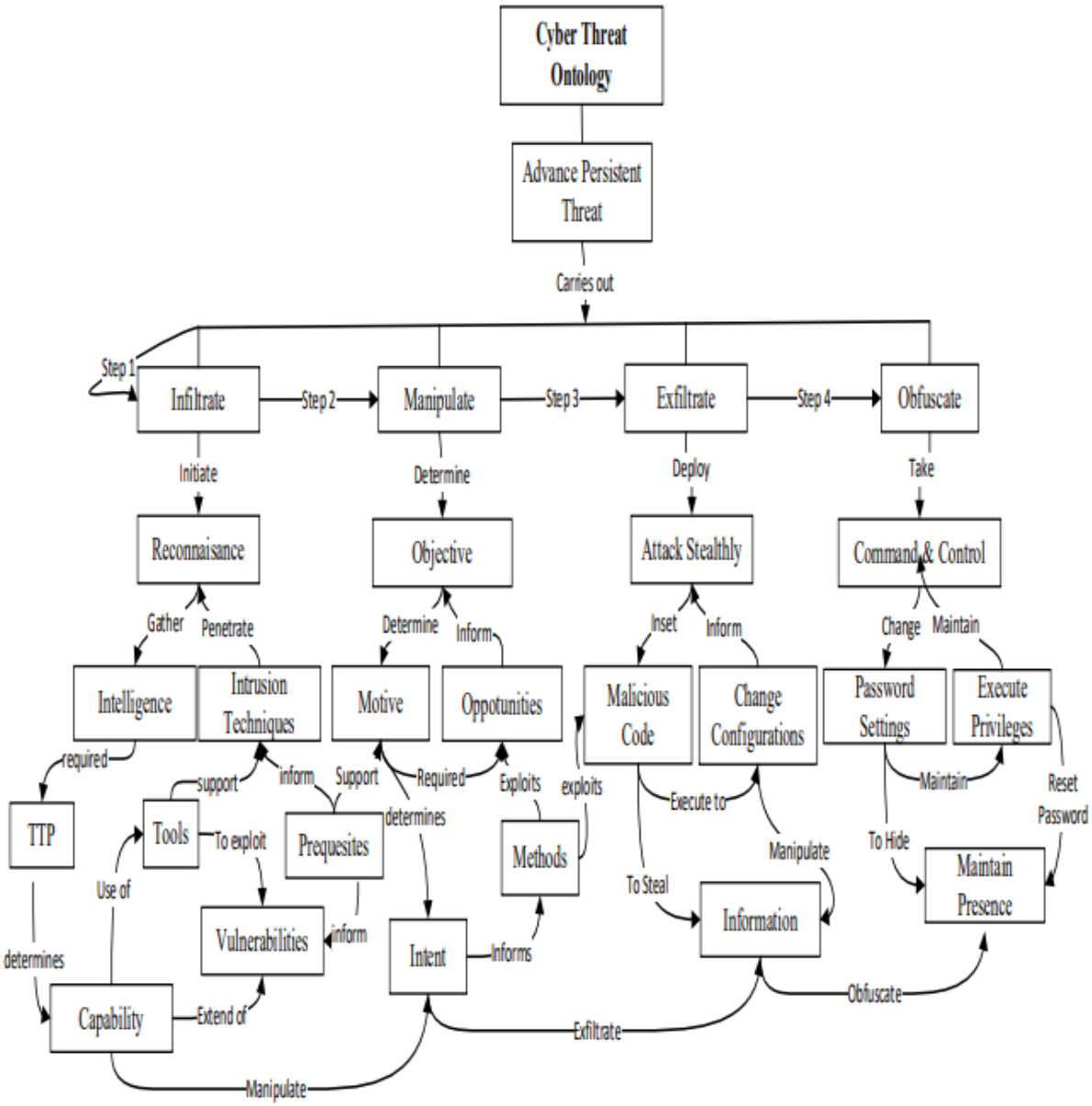

Figure 2.2: CTO Conceptual Model

As for The adversarial attack model implementation it considered a penetrated network using remote access trojan (RAT) attack. The adversarial attacker was able to misclassify the prediction using advanced persistent threats (APT), command and control capabilities. It measured the effectiveness of malicious attacks on the feature selection criteria on the dataset during the retraining and retesting time to determine the degree of the poisoned rates on the ransomware, malware, spyware, and RAT attacks.

Figure 2.3 shows the results on the adversarial attacks scenarios using various ML algorithms presenting the performance accuracies of the different threats

ACCURACY	RF 72%			GBOOST 79%		
ATTACKS	P	R	F	P	R	F
Ransomware	0.73	0.71	0.72	0.78	0.76	0.77
RAT	0.69	0.66	0.67	0.72	0.70	0.71
Malware	0.72	0.69	0.72	0.79	0.76	0.77
Spyware	0.73	0.70	0.71	0.78	0.75	0.76

(a) before the attack

ACCURACY	RF 22%			GBOOST 25%		
ATTACKS	P	R	F	P	R	F
Ransomware	0.24	0.20	0.22	0.26	0.24	0.25
RAT	0.22	0.69	0.21	0.25	0.23	0.24
Malware	0.24	0.21	0.23	0.27	0.24	0.25
Spyware	0.23	0.21	0.22	0.26	0.24	0.25

(b) after the attack

Figure 2.3: Predictions Using RF and GBoost Classifiers after malicious adversarial attack

The Goal of analysing the CTO for the APT attack was to extract relevant attack instances from the intelligence for accuracy in security concepts and knowledge reuse in the threat intelligence domain. The CTO acted as a link that connects the concepts with the threat information required for predicting future trends. Therefore, it's not involved in mitigating the threat directly.

2.7 Conclusion

In this chapter, we have defined the ontology and its types. We have also describe the structure of the ontology and its relationship with the semantic web by specify the ontology representation languages.

We have defined the Cyber threat ontology related works and how it is a valuable tool for improving our understanding of the complex and evolving landscape of cyber threats. Its development and use can help to enhance our ability to protect against cyber attacks, and ultimately, to safeguard the security and privacy of individuals and organizations in the digital age.

Finally, the detection of adversarial attacks is a critical problem in machine learning, and by using CTO concepts to analyse the perturbance data can be a inventive solution to ensure the integrity of our training and testing data. This is what we are going to prove by creating a cyber threat ontology and use it to analyse and conclude the true outcome for the machine learning model.

Chapter 3

Conception

3.1 Conceptual Model

In order to get the best outcomes and increase the performance of the model we have to ensure the integrity of data. For this reason, we create a Cyber Threat Ontology (CTO) that describes different concepts and relationships which provide a standardized way of representing and sharing information related to cyber threats including vulnerabilities, softwares, adversary groups, tactics and techniques used in cybersecurity field.

In addition, we use the ontology concepts and rules to analyse our cybersecurity dataset and compare the model's outcomes with the ontology reasoning. This will indicate whether an adversarial attack occurred or not.

For instance, our conceptual model illustrated in the figure 3.1 describes the following parts:

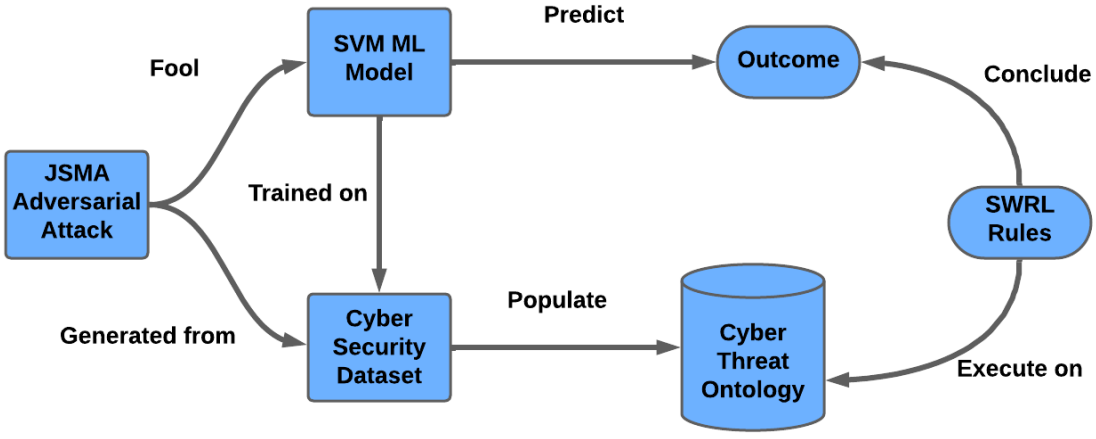

Figure 3.1: Conceptual Model.

- **Adversarial Attack:** We use a JSMA attack which is generated from a dataset. The latter, represents the fool model to predict false narratives;

- **ML Model:** We use SVM model to train and test the cybersecurity dataset and make an accurate predictions about the data;
- **Cybersecurity Dataset:** We create a CVSS dataset to collect a set of vulnerabilities and their scores;
- **Outcome:** Which stands for the prediction result of our model after training and testing the dataset;
- **Cyber Threat Ontology:** We create a cyber threat ontology and use its rules to analyse data integrity and check the outcome prediction.
- **SWRL Rules:** A set of rules which we apply it on the CTO after inserting our dataset to conclude the outcomes.

3.2 Common Vulnerability Scoring System (CVSS) Dataset:

There are several cybersecurity datasets available that can be used for research, training, and testing of machine learning algorithms including the NSL-KDD [7] and CICIDS2017 [20]. However, creating a specific dataset can ensure that the model works with convidence data and gives a better results.
Therefore, we create our dataset using these two components:

1. **CVE:** defined as a Common Vulnerabilities and Exposures. It is a system used to identify and track known cybersecurity vulnerabilities in software and hardware systems. we use the **CVE ID** to assign each vulnerability to its identifier number;
2. **CVSS** is a Common Vulnerability Scoring System. It is a framework used to assess the severity of cybersecurity vulnerabilities and provides a standardized method for evaluating the impact of vulnerabilites. In our dataset, we use the following metrics :
    - **Attack Vector:** Describes how the vulnerability can be exploited and the level of access required to exploit it. In addition, it scores range from Local to Network;
    - **Attack Complexity:** Describes how difficult it is for an attacker to exploit the vulnerability and it scores range from Low to High;
    - **Privileges Required:** This metric describes the level of privileges needed to exploit the vulnerabilities and it scores range from None to High;
    - **User Interaction:** Describes if an attacker requires user interaction or not to exploit the vulnerabilities and it scores range from None to Required;

- **Scope:** Describes the extent of the impact of the vulnerability. Scores range from Unchanged to Changed;
- **Impact Metrics:** These metrics assess the impact of the vulnerability on the confidentiality, integrity, and availability of the affected system. Scores range from None to High.

3. **NVD:** Stands for National Vulnerability Database. It is a project of the National Institute of Standards and Technology (NIST) in the United States. The NVD maintains a comprehensive database of information about software vulnerabilities, including CVE identifiers, vulnerability descriptions, and links to security advisories and patches;

4. **CVSS Calculator:** Is a tool used to determine the severity of software vulnerabilities based on a standardized scoring system.
   This calculator takes all the CVSS metrics and produces a numerical score between 0.0 and 10.0, with higher scores indicating more severe vulnerabilities. The score is accompanied by a textual severity rating, which can be low, medium, high, or critical, depending on the score range as it is shown in the Figures 3.2 and 3.3.

Id	Attack_Vector	Attack_Complexity	Privileges_Required	User_Interaction
20130375	Network	Low	Low	None
20143566	Network	High	None	Required
20121516	Network	Low	Low	None
20090783	Local	Low	High	None
20120384	Network	Low	High	None

Figure 3.2: Example from the CVSS Dataset.

Scope	Confidentiality_Imp...	Integrity_Impact	Availability_Impa...	Score
Changed	Low	Low	None	6.4
Unchanged	Low	None	None	3.1
Changed	High	High	High	9.9
Unchanged	Low	Low	Low	4.2
Unchanged	High	High	High	7.2

Figure 3.3: Example of CVSS Dataset scores.

3.3 Development softwares:

- **Python:** Python is a popular high-level, interpreted programming language that is known for its simplicity, ease of use, and readability. It was created by Guido van Rossum in the late 1980s and was first released in 1991. Python's syntax is easy to understand and its dynamic typing allows for flexibility in programming. It supports many programming paradigms including object-oriented, functional, and procedural programming.
  Why Python ?
  Python is widely used in a variety of fields including web development, scientific computing, data analysis, and machine learning due to its large standard library and many third-party modules and packages. It is also cross-platform, meaning it can run on various operating systems including Windows, macOS, and Linux.

- **Google Colaboratory:** also known as Google Colab, is a cloud-based development environment for creating and running Jupyter Notebook documents. It provides access to a variety of powerful computing resources, including GPUs which can be used to train machine learning models and run computationally intensive tasks. It also includes many popular Python libraries and tools, such as TensorFlow, PyTorch, and scikit-learn, making it a convenient and powerful platform for data science and machine learning projects.

- **Pandas:** Pandas is an open-source data analysis and manipulation library for Python that provides data structures and tools for working with labeled and indexed data. Pandas is widely used in data preprocessing, cleaning, exploration which provides a wide range of functions for data manipulation, including merging, grouping, filtering, and reshaping data.

- **Numpy:** NumPy is an open-source numerical computing library for Python that provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on them. This library is widely used in scientific computing, data analysis, and machine learning applications.

3.4 Support Vector Machine (SVM) Model:

Among the various machine learning models we mentioned in the first chapter, the SVM model was the one picked to demonstrate the perturbance. mainly for the following reasons :

- **Popularity and Historical Significance:** SVMs are widely used in the field of machine learning which can be tracked back to the 1990s. Many researchers has been interested in evaluating the robustness of SVMs against adversarial attacks.
- **Robustness:** this model is known to be robust to overfitting and generalizing errors. This makes them a good choice for evaluating adversarial attacks because they can handle perturbations in the input data without being affected too much.
- **Simplicity:** SVMs are a relatively simple and interpretable machine learning model, this makes it easier to understand how an adversarial attack affects the model's decision-making process. In contrast, other models such as deep neural networks can be very complex and difficult to interpret.

In order to interpret the data with this model we utilize many well known libraries in python such as:

- **Scikit-learn:** also known as sklearn is a widely-used Python library for machine learning that provides a range of tools for building and analyzing machine learning models. Scikit-learn is built on top of the NumPy, SciPy, and Matplotlib libraries, and provides a user-friendly interface for performing common machine learning tasks such as classification, regression, clustering, and dimensionality reduction.

3.5 Jacobian-Based Saliency Map Attack (JSMA):

There are many reasons to choose the Jacobian-based Saliency Map Approach (JSMA) attack we mention some of them:

- Although originally designed for DNNs, researchers have also applied JSMA to SVM and other types of models. it is applicable to multiple model types which make it wide spread and favorable.
- The attack algorithm is simple and easy to implement which makes it a good "baseline" to demonstrate the concept of adversarial examples and attacks.

To implement the JSMA attack, we use:

- **Adversarial Robustness Toolbox (ART):** Developed by IBM to help researchers and developers to evaluate the robustness of machine learning models against adversarial attacks.
  ART provides a comprehensive set of tools for generating adversarial examples, evaluating model robustness, and implementing defense mechanisms. It supports a wide range of machine learning frameworks, including TensorFlow, Keras, PyTorch, and scikit-learn, and can be used with both deep learning and traditional machine learning models.

3.6 The hierarchy of concepts in CTO model :

In ontology, concepts refer to the abstract ideas or categories that are used to describe the entities and objects within a particular domain. Concepts are used to define the terms and relationships that are used within an ontology.

- The class Thing is often defined as the universal set of all entities within a domain, and it is used to represent any entity that can be described or referenced within the ontology. It is considered as the root of the class hierarchy, and all other classes within an ontology are sub-classes of the Thing class.
- We used the **MITRE ATT&CK** framework to collect information and fill our ontology, it is a comprehensive knowledge base of cyber adversary tactics, techniques, and procedures (TTPs) that are commonly used in cyber attacks. This later, is a non-profit organization that manages research and development centers funded by the federal government.

Here is our cyber threat ontology model as shown in Figures 3.4, 3.5, 3.6, 3.7 and 3.8.

- **Tactic :** This concept represents the adversary's tactical goal which is the reason for performing an action or a cyber attack. For example reconnaissance tactic;
- **Technique :** Describes the way of an adversary to achieve a tactical goal which is the cyber attack;
- **AdversaryGroup :** is a group of attackers that have been observed using specific tactics and techniques. For example, a phishing for information;
- **Mitigation :** Represent the classes of technologies and security concepts that can be used for the prevention of the techniques and cyber attacks. For instance, using Antivirus or Encrypt Sensitive Information;
- **Software :** Is a program used to implement a technique, such as using The Metasploit Tool to exploit a vulnerability.
- **Vulnerability :** It is a weakness or flaw in a system which can be exploited by a technique. For example, execute malicious code throw Backdoor.

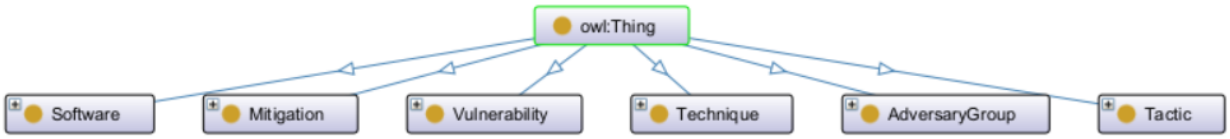

Figure 3.4: Cyber threat ontology model.

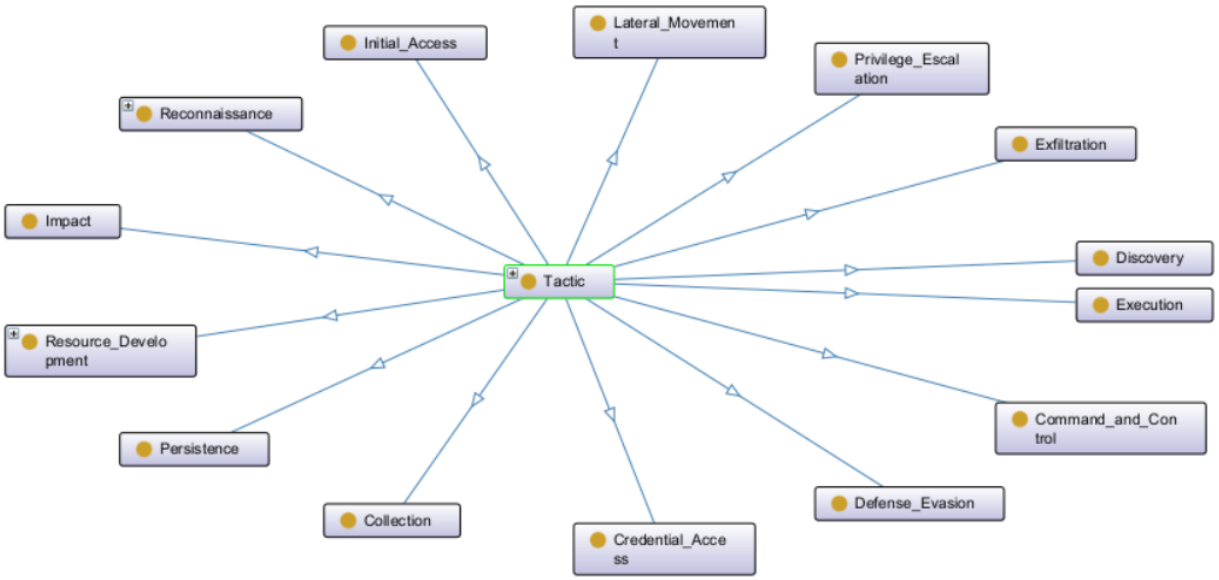

Figure 3.5: Tactics Ontology Model

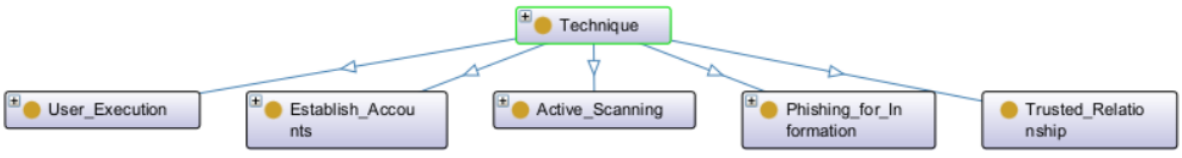

Figure 3.6: Techniques Ontology Model

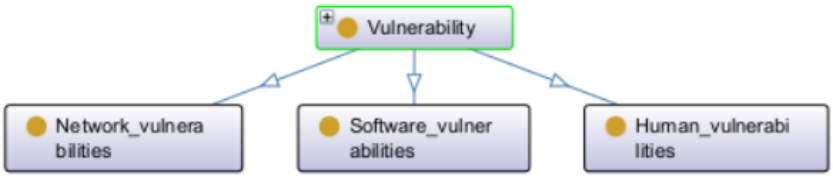

Figure 3.7: Vulnerability Ontology Model

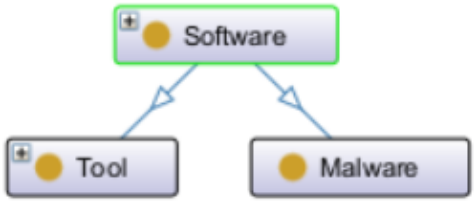

Figure 3.8: Software Ontology Model

3.6.1 Domain Ontology

After defining the hierarchy and creating the concepts, it is necessary to link these concepts by using the semantic relationships. As illustrated in the upcoming Figure 3.9, for example, we create the relationship “Uses Technique” between “Adversary Group” and “Technique” concepts.

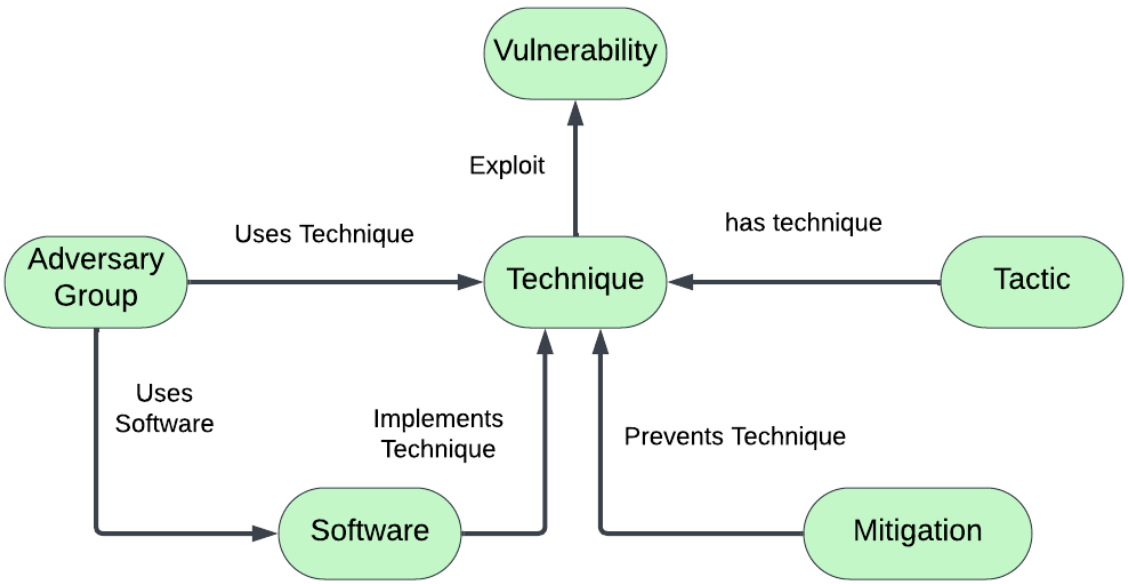

Figure 3.9: Domain Ontology

3.6.2 Semantic Web Rule Language (SWRL)

3.6.2.1 Definition

SWRL (Semantic Web Rule Language) is a language for writing rules that operate on RDF data. It is designed to be used in conjunction with OWL and can be used to express complex relationships and constraints between entities in an ontology.

3.6.2.2 SWRL Rule Structure

The structure of a SWRL rule consists of two parts:

- body (also known as the antecedent or precondition) : The body of the rule specifies the conditions that must be met in order for the rule to trigger.
- head (also known as the consequent or conclusion) : The head of the rule specifies the actions that should be taken if the conditions in the body are satisfied.

Example 1 (Adult person) *In rule 3.1, the body specifies that the entity ?p must be a Person and must have an age greater than 18. If these conditions are met, the head of the rule asserts that the entity ?p has the status "adult".*

```
Person(?p)^hasAge(?p, ?age)^ swrlb:greaterThan(?age, 18) => hasStatus(?p, "adult")
```

Code 3.1: Example of rule.

3.6.3 SPARQL Protocol and RDF Query Language (SPARQL)

SPARQL (SPARQL Protocol and RDF Query Language) is a query language designed for querying RDF data. It allows users to retrieve and manipulate data stored as RDF triples and provides a way to search for patterns in the data and to extract information based on these patterns. SPARQL is flexible and powerful, and it supports different types of queries, including SELECT, ASK, CONSTRUCT, and DESCRIBE.
SPARQL queries consist of a set of triple patterns, which are similar to RDF triples, but with variables in place of specific values. These triple patterns can be combined using logical operators such as "AND", "OR", and "NOT", and can also include filters, functions, and aggregations.

```
PREFIX ontology: <http://example.org/ontologies/cyberthreat#>
SELECT {?vulnerability ?cvssScore}
WHERE { ?vulnerability a ontology:Vulnerability; ontology:CVSS_Score ?cvssScore}
FILTER (?cvssScore>=7.0)
}
```

Code 3.2: Sparql query example.

The example above shows how to select vulnerabilities (?vulnerability) and their corresponding CVSS scores (?cvssScore) from an ontology or RDF graph.

- The **PREFIX** statement sets the namespace prefix ontology to the appropriate URI for your ontology.
- The **SELECT** statement specifies the variables to retrieve. In this case, ?vulnerability and ?cvssScore are the variables.
- The **WHERE** clause defines the pattern to match. It states that the ?vulnerability should have the type ontology:Vulnerability and a ontology:CVSS_Score property with value ?cvssScore.
- The **FILTER** clause applies a condition to filter vulnerabilities based on their CVSS score. In this example, it filters for vulnerabilities with a CVSS score greater than or equal to 7.0

3.6.4 Protege Software and Other Tools

- **Protege:** Protege is an open-source ontology editor and knowledge management system used to create, maintain, and manage ontologies. It provides a graphical user interface that allows users to create and edit ontologies in a user-friendly way, without requiring knowledge of ontology languages such as OWL or RDF. Protege is widely used in academic and industry settings for various purposes, including biomedical research, knowledge engineering, and semantic web applications.
  Here are the different plugins available in Protégé:

  1. **Ontology Web Browser:** An ontology web browser is a software tool that allows users to explore and navigate ontologies on the web.
  2. **XML Schema:** An XML schema is a description of the structure and content of an XML document. It defines the rules and constraints that an XML document must follow in order to be considered valid.
  3. **OntoGraf:** It is a graphical representation or visualization of an ontology, which displays the concepts and relationships defined in the ontology in a visual format.

- **Rdflib:** RDFlib is a Python library for working with RDF data. It provides a set of tools and APIs for parsing, manipulating, and serializing RDF data in various formats, such as RDF/XML, Turtle, and N-Triples.
- **Owlready2:** Owlready2 is a Python library for working with ontologies and Semantic Web data. It provides a set of tools and APIs for creating, manipulating, and querying ontologies expressed in the OWL format. Owlready2 is designed to be easy to use, and provides a high-level, object-oriented interface for working with ontologies.

3.7 Conclusion

In this chapter, we have described our conceptual model used to detect adversarial attacks. First, we have explained how to create the cybersecurity dataset and the tools used to collect and work with our data. Secondly, we have detailed our machine learning model and how it works. Next, we have specified the adversarial machine learning attack and how it changes the data and cause the ML model to predict false narratives. Finally, we have defined the hierarchy of our cyber threats ontology from the classes, properties into the rules, as well as the different tools that we used to create our ontology and detect the adversarial attacks.

Chapter 4

Implementation

4.1 Introduction

In this chapter we have detailed each phase of our conceptual model, we start with creating the cyber security dataset, train and test the data with the machine learning model. Then, we implement the adversarial attacks against the model and finaly using the cyber threat ontology to anaylse the data and conclude the true results.

4.2 Creating and analysing the dataset

In order to create our cybersecurity dataset, we have used the National Vulnerability Database (NVD) and the CVSS Calculator. Later, We create a csv file **cvss.csv** which collect 200 vulnerabilities and their scores. Therefore, we can display some information about our dataset, for example, the columns names, dataset lines and some other information, such as illustrated in Figures 4.1, 4.2 and 4.3.

```
[3]  dataset.columns

     Index(['Id', 'Attack_Vector', 'Attack_Complexity', 'Privileges_Required',
            'User_Interaction', 'Scope', 'Confidentiality_Impact',
            'Integrity_Impact', 'Availability_Impact', 'Score'],
           dtype='object')
```

Figure 4.1: Analyse the columns names.

	Id	Attack_Vector	Attack_Complexity	Privileges_Required	User_Interaction	Scope	Confidentiality_Impact	Integrity_Impact	Availability_Impact	Score
0	20130375	Network	Low	Low	None	Changed	Low	Low	None	6.4
1	20143566	Network	High	None	Required	Unchanged	Low	None	None	3.1
2	20121516	Network	Low	Low	None	Changed	High	High	High	9.9
3	20090783	Local	Low	High	None	Unchanged	Low	Low	Low	4.2
4	20120384	Network	Low	High	None	Unchanged	High	High	High	7.2

Figure 4.2: Display the lines.

```
#read the cvss file
dataset=pd.read_csv('cvss.csv')
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Id                     200 non-null    int64
 1   Attack_Vector          200 non-null    object
 2   Attack_Complexity      200 non-null    object
 3   Privileges_Required    200 non-null    object
 4   User_Interaction       200 non-null    object
 5   Scope                  200 non-null    object
 6   Confidentiality_Impact 200 non-null    object
 7   Integrity_Impact       200 non-null    object
 8   Availability_Impact    200 non-null    object
 9   Score                  200 non-null    float64
dtypes: float64(1), int64(1), object(8)
memory usage: 15.8+ KB
```

Figure 4.3: Load the Dataset.

4.3 Training the ML model on the dataset

After creating and analysing the dataset, we will implement the SVM model on the dataset we created to classify our data into two classes (High severity and Low severity). We consider 65% of samples for training (130) and the remaining 35% for testing (70). Figure 4.4 shows the performance of the SVM model.

```
dataset_train , dataset_test = split_data(dataset,0.65)
svm_model = train_model(dataset_train)
X_test , y_test , y_test_pred = test_model(dataset_test,svm_model)
show_res(y_test_pred, y_test)
```

```
[[38  0]
 [ 0 32]]
accuracy: 1.0000
precision= 1.0
recall= 1.0
```

Figure 4.4: Training the SVM model

Results of training, is detailed in the form of the following parameters :

1. **The Confusion Matrix:** such as represented in Figure 4.5, a confusion matrix is a table that summarizes the performance of a classification model by organizing the predicted and actual classifications into four categories:

a) **True Positive (TP):** the model correctly predicted a positive class. In our case, it is a 38 samples;

b) **False Positive (FP):** the model incorrectly predicted a positive class when the actual class was negative and in our case there is 0 samples;

c) **True Negative (TN):** the model correctly predicted a negative class. There is a 32 samples;

d) **False Negative (FN):** the model incorrectly predicted a negative class when the actual class was positive as 0 samples.

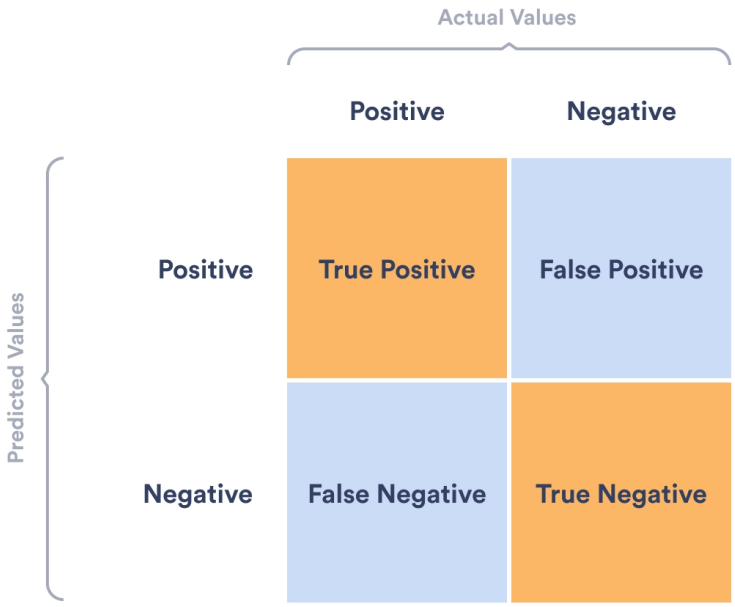

Figure 4.5: Confusion matrix.

2. **Accuracy:** measures the overall correctness of the model's predictions and it is defined as the ratio of the number of correct predictions to the total number of predictions.

3. **Precision:** measures the proportion of correctly predicted positive instances out of all instances that the model predicted as positive. It is defined as the ratio of true positives to the sum of true positives and false positives.

4. **Recall:** measures the proportion of correctly predicted positive instances out of all actual positive instances. It is defined as the ratio of true positives to the sum of true positives and false negatives.

With an accuracy , precision and recall of 100% we can confidently say that the model did very well at classifying the data correctly.

4.4 Realizing the Adversarial Attack

In order to get a simulate attack, we need to generate adversarial samples that we have used of the Adversarial robustness toolbox (ART) library to perform a JSMA attack on the testing samples.
After that, we inject the new created samples into the SVM model to compare it with the original results to in order to know how much the model is effected.
Figure 4.6 illustrates the performance of the SVM model on the adversarial samples. The accuracy dropped from 100% to 0% which is a noticeably big impact.

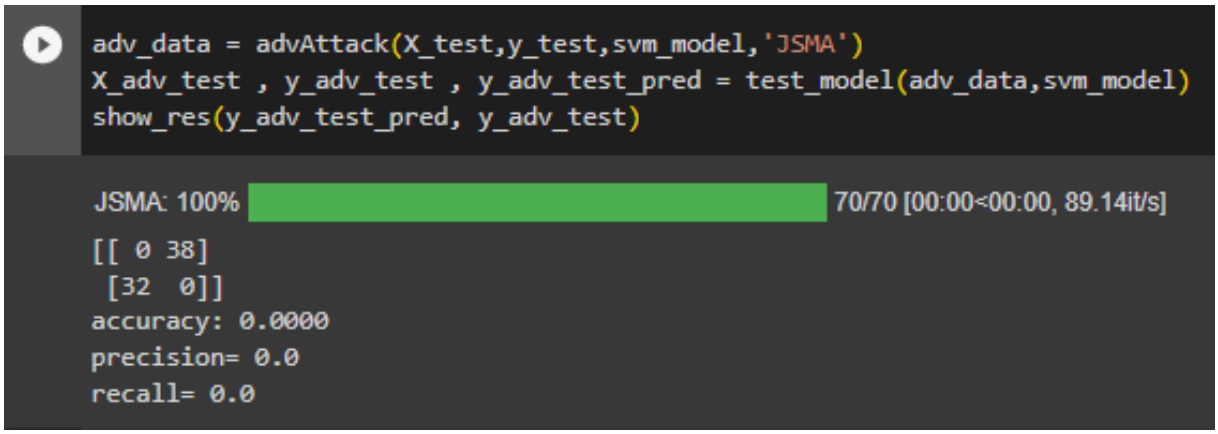

Figure 4.6: Showcasing the effects of the attack.

4.5 Create the Cyber Threat Ontology

As we mentioned earlier in the conception part of Chapter 3, we created the classes of our use case study such as, Tactic, Technique, AdversaryGroup, Mitigation, Software and Vulnerability.

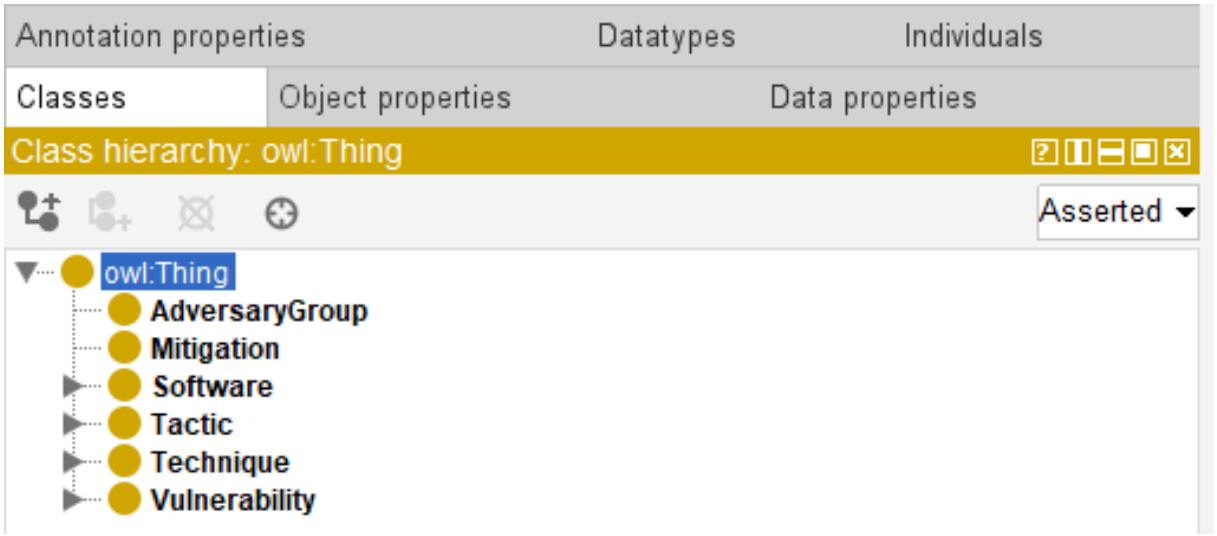

Figure 4.7: Ontology classes.

Moreover, we created the object properties and data properties for our ontology as demonstrated in Figure 4.8. For example, the class Technique “Exploit” a Vulnerability and the inverse of Exploit is “Exploited_by”.

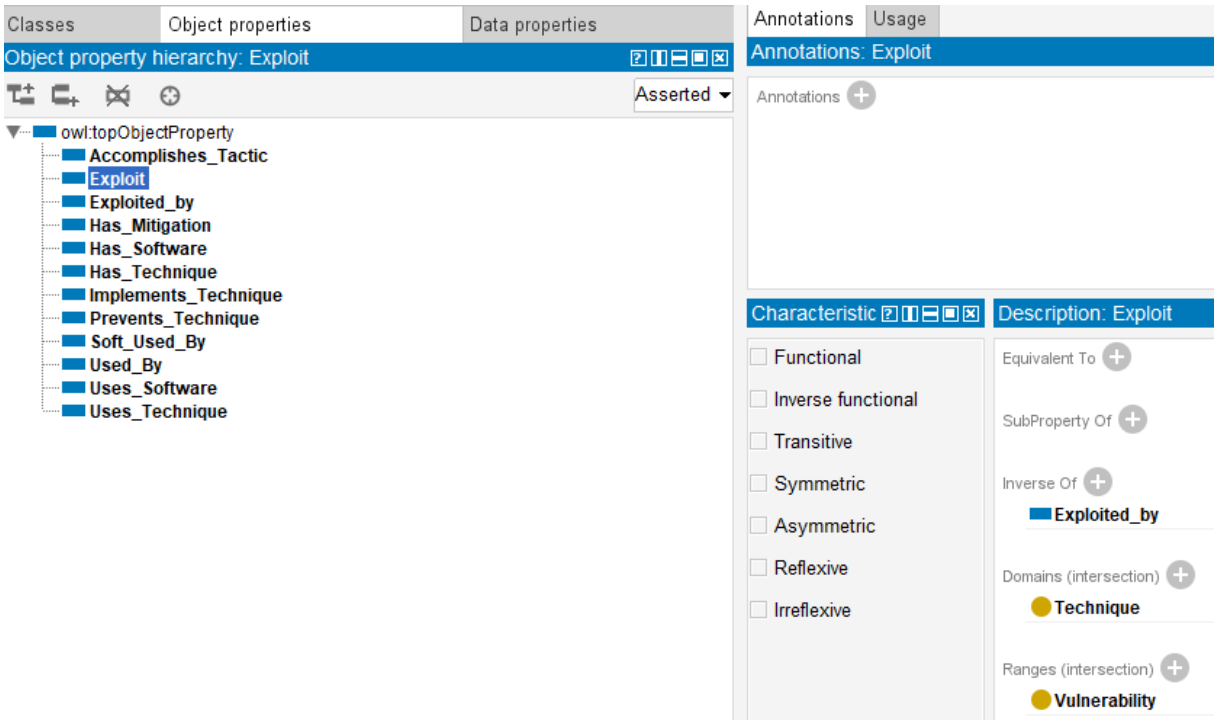

Figure 4.8: Object Properties.

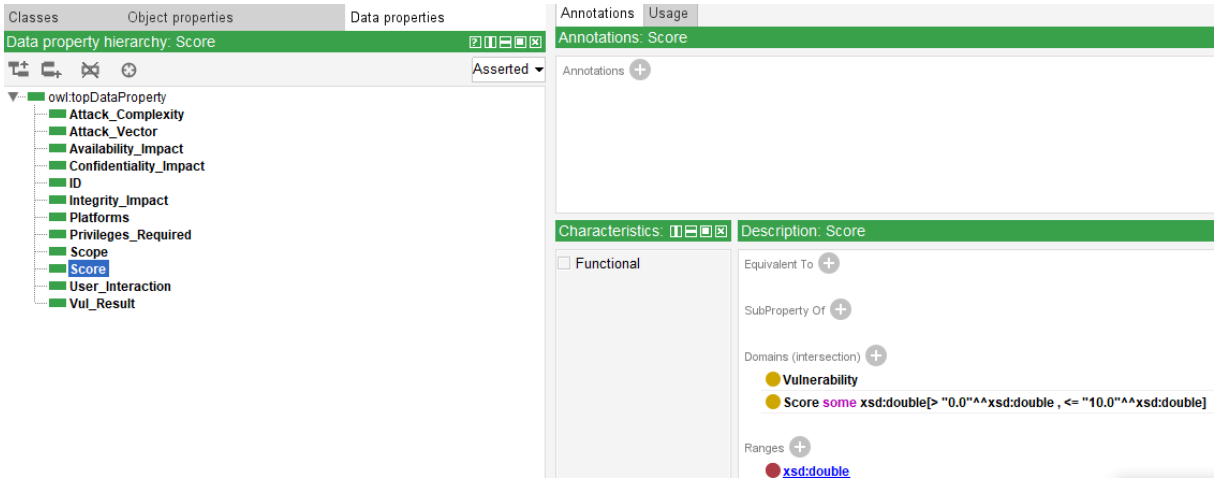

Figure 4.9: Data Properties.

Furthermore, we populate the ontology concepts by their instances and data properties, for instance in Figure 4.9, vulnerability instance has a score property and his type is double. The range of the score is between 0 to 10.

Another example of class individual part of our ontology, the instance of a Mitigation class is an **Antivirus**. Figure 4.10 shows an "Antivirus prevents the spearphishing attachment technique" and it's ID is **M1049**.

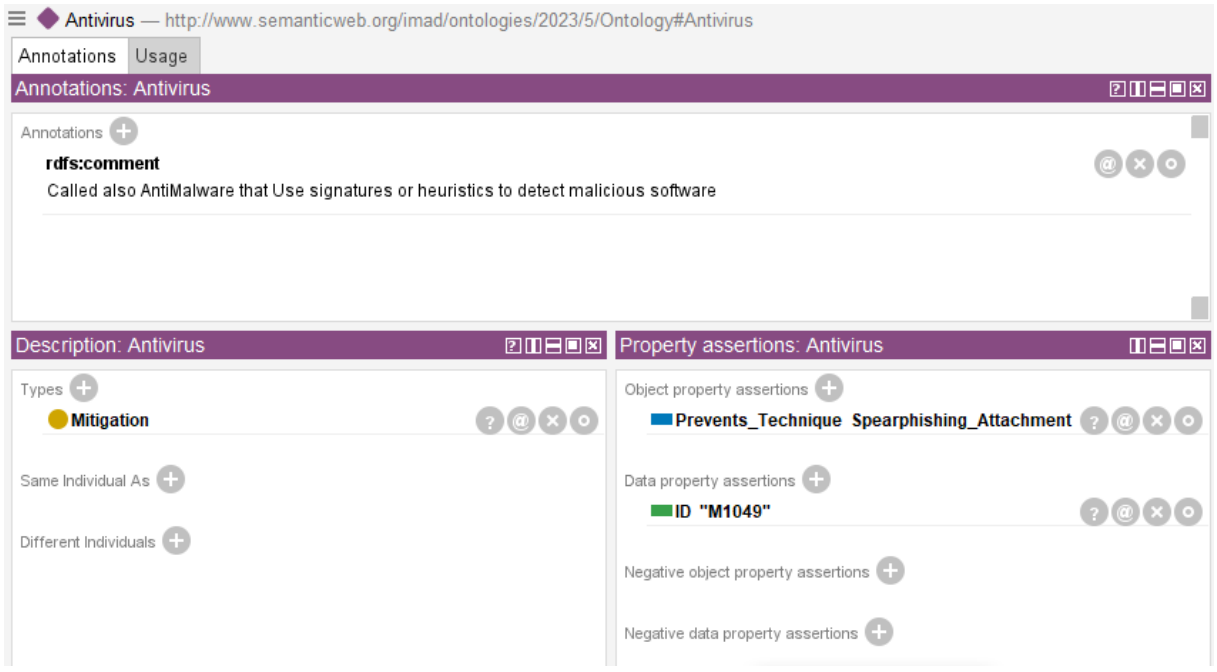

Figure 4.10: Ontology individuals.

4.6 Using CTO to detect Adversarial Attack

To use the CTO rules for the conclusion of the results, we follow these steps :

1. Define the SWRL rules in the cyber threat ontology.
2. Save our testing data in the ontology as individuals.
3. Apply the rules on individuals.
4. Return the result and compare it to the Adversarial attack results

First, we define the SWRL rules as detailed in Figure 4.11. shows :

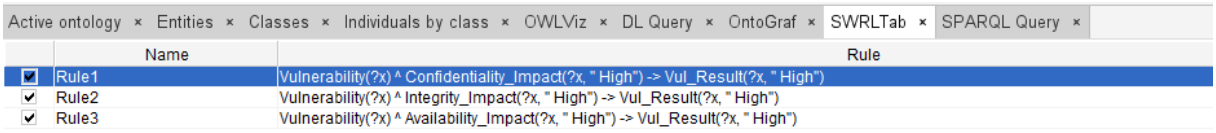

Figure 4.11: Save the dataset Lines into the ontology.

These rules indicates that if the vulnerability instance has one of the impact metrics (Confidentiality, integrity and Availability) as high value, then the result must be high. After that, by saving the testing data in the ontology and apply the rules on it. Finally, we obtain the final result of our vulnerability which can be **High** or **Low** such as defined in Figure 4.12.

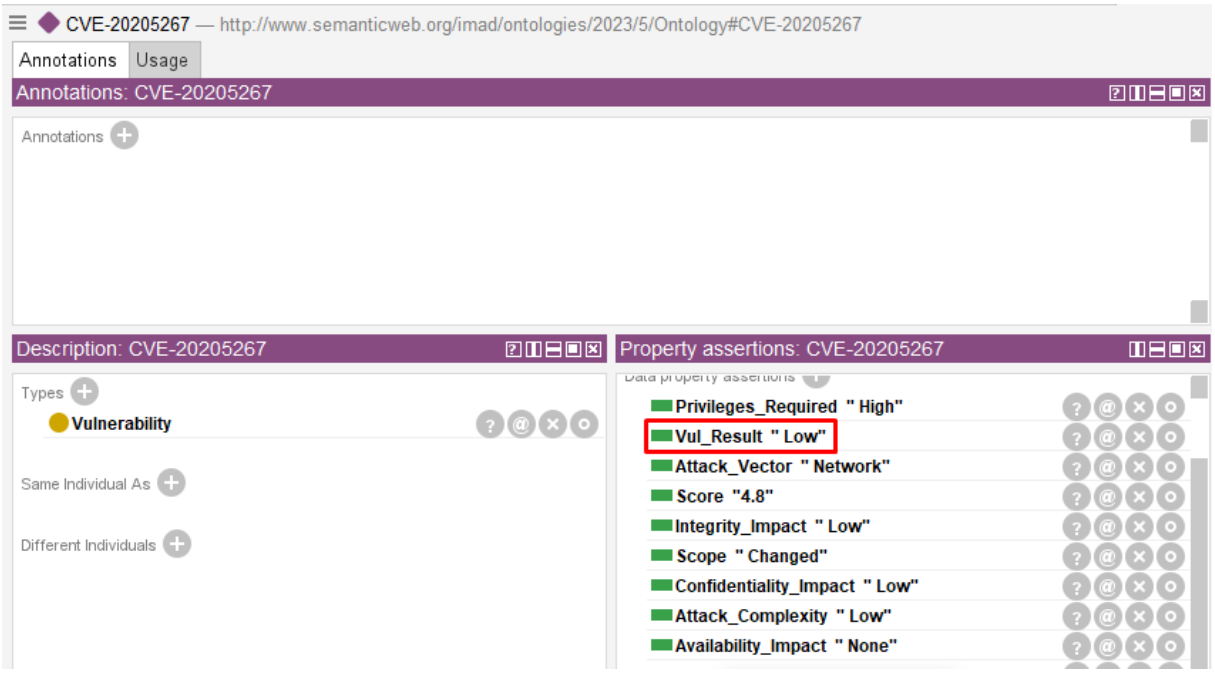

Figure 4.12: Ontology Result.

As illustrated in Figure 4.13, vulnerability result setted into **Low** by applying the previous rules.

Then, we return these results and compare it with the predicted results by the machine learning model (SVM) after doing the JSMA adversarial attack.

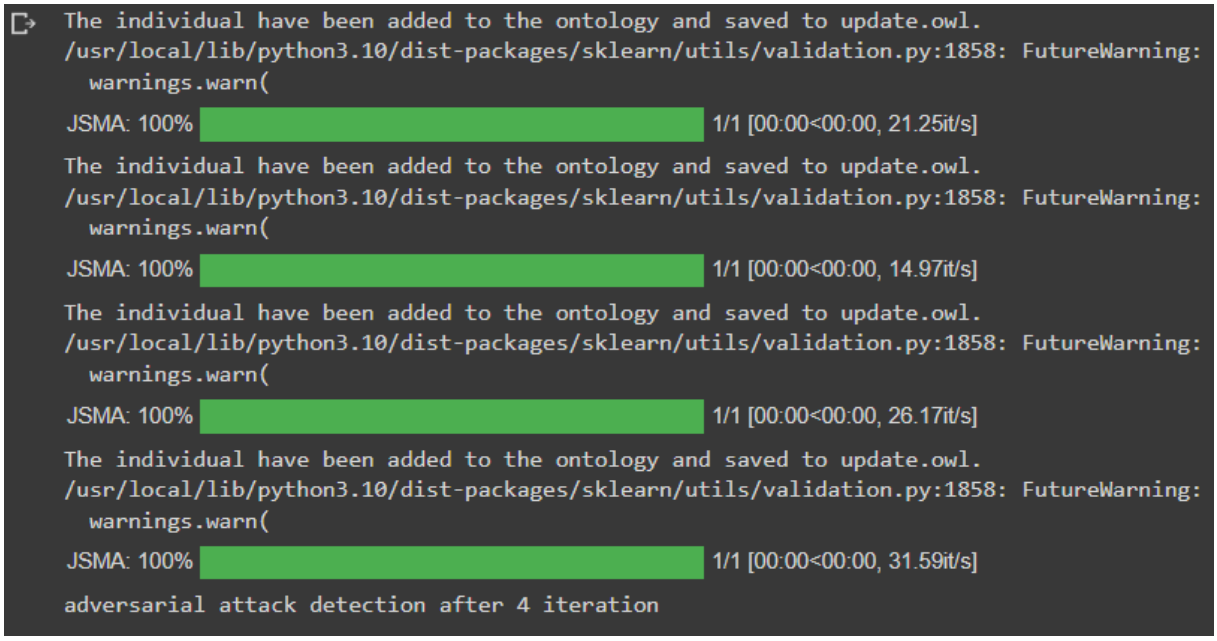

Figure 4.13: Final Result.

The message **"adversarial attack detection after 4 iteration"** displayed when the number of the true samples predicted is lower than (70%) as defined in the threshold function of figure 4.14.

```
threshold = true / (false + true )

if threshold < 0.7:
  print('adversarial attack detection after', i+1, 'iteration')
  break
```

Figure 4.14: Threshold function.

4.7 Conclusion

As a conclusion, we have implemented an ontology-based solution for detecting adversarial attacks which requires a careful consideration of the specific use case and the types of attacks that are likely to be encountered. It may also requires significant expertise in both machine learning and ontology design.

We have implemented our conceptual model by creating and analysing our cybersecurity dataset and training the machine model on it. Next, we create the concepts and define the rules of the cyber threat ontology which we use to conclude the result of dataset samples and compare it with

the prediction of the machine learning model to check if there is an adversarial attack or not. Additionally, ontologies alone may not be sufficient for detecting all types of adversarial attacks, as attackers may be able to craft attacks that are specifically designed to evade detection by the ontology-based system. Therefore, it is important to consider ontologies as one component of a larger defense strategy that includes other techniques such as anomaly detection and robust machine learning algorithms.

General Conclusion

Our work falls within the field of cybersecurity. The goal of our work is to detect adversarial machine learning attacks by proposing a cyber threat ontology.

To do this, we started by introducing Adversarial Machine Learning Attacks. Then, we presented the concept of ontology. Finally, we showed the perturbance prediction of adversarial attacks using a cyber threat ontology.

The main contribution of our work is to design, use and exploit an ontological model containing base classes, subclasses and rules for the concepts of our work context. This ontology conclude the results based on ontological concepts and rules which can help the detection of adversarial attacks.

For this purpose, we have created a cybersecurity dataset and insert the dataset lines into a Cyber Threat Ontology (CTO). Then, we conclude the result of each line by applying rules and check the conclusion with the ML model prediction results after performing an adversarial machine learning attack.

Finally, we can conclude that adversarial attacks pose a significant threat to the accuracy and reliability of machine learning systems. Ontology-based solutions can be a valuable tool for detecting these attacks by providing a way to represent the relationships and apply rules between the various components of the system.

Bibliography

[1] Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. (2016). Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM.

[2] Allen, R. B. (2020). Semantic modeling with sumo. *arXiv preprint arXiv:2012.15835*.

[3] Alotaibi, A. and Rassam, M. A. (2023a). Adversarial machine learning attacks against intrusion detection systems: A survey on strategies and defense. *Future Internet*, 15(2):62.

[4] Alotaibi, A. and Rassam, M. A. (2023b). Adversarial machine learning attacks against intrusion detection systems: A survey on strategies and defense. *Future Internet*, 15(2).

[5] Baneyx, A. (2007). *Construire une ontologie de la Pneumologie Aspects théoriques, modèles et expérimentations*. PhD thesis, Université Pierre et Marie Curie-Paris VI.

[6] Biggio, B. and Roli, F. (2018). Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331.

[7] Dhanabal, L. and Shantharajah, S. (2015). A study on nsl-kdd dataset for intrusion detection system based on classification algorithms. *International journal of advanced research in computer and communication engineering*, 4(6):446–452.

[8] Doynikova, E., Fedorchenko, A., and Kotenko, I. (2019). Ontology of metrics for cyber security assessment. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, pages 1–8.

[9] Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., and Song, D. (2018). Robust physical-world attacks on deep learning models.

[10] Fadel, F. G., Fox, M. S., and Gruninger, M. (1994). A generic enterprise resource ontology. In *Proceedings of 3rd IEEE workshop on enabling technologies: infrastructure for collaborative enterprises*, pages 117–128. IEEE.

[11] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks.

[12] Grigoriadis, C., Berzovitis, A. M., Stellios, I., and Kotzanikolaou, P. (2022). A cybersecurity ontology to support risk information gathering in cyber-physical systems. In *Computer Security. ESORICS 2021 International Workshops: CyberICPS, SECPRE, ADIoT, SPOSE, CPS4CIP, and CDT&SECOMANE, Darmstadt, Germany, October 4–8, 2021, Revised Selected Papers*, pages 23–39. Springer.

[13] Huang, Q., Katsman, I., Gu, Z., He, H., Belongie, S., and Lim, S.-N. (2019). Enhancing adversarial example transferability with an intermediate level attack. pages 4732–4741.

[14] Ibitoye, O., Abou-Khamis, R., el Shehaby, M., Matrawy, A., and Shafiq, M. O. (2023). The threat of adversarial attacks on machine learning in network security – a survey.

[15] Ibitoye, O., Abou-Khamis, R., Matrawy, A., and Shafiq, M. O. (2019). The threat of adversarial attacks on machine learning in network security–a survey. *arXiv preprint arXiv:1911.02621*.

[16] Jia, Y., Qi, Y., Shang, H., Jiang, R., and Li, A. (2018). A practical approach to constructing a knowledge graph for cybersecurity. *Engineering*, 4(1):53–60.

[17] Liu, H. and Lang, B. (2019). Machine learning and deep learning methods for intrusion detection systems: A survey. *Applied Sciences*, 9(20).

[18] Merah, Y. and Kenaza, T. (2021). Ontology-based cyber risk monitoring using cyber threat intelligence. In *Proceedings of the 16th International Conference on Availability, Reliability and Security*, pages 1–8.

[19] Mozzaquatro, B. A., Agostinho, C., Goncalves, D., Martins, J., and Jardim-Goncalves, R. (2018). An ontology-based cybersecurity framework for the internet of things. *Sensors*, 18(9):3053.

[20] Panigrahi, R. and Borah, S. (2018). A detailed analysis of cicids2017 dataset for designing intrusion detection systems. *International Journal of Engineering & Technology*, 7(3.24):479–482.

[21] Sánchez-Zas, C., Villagrá, V. A., Vega-Barbas, M., Larriva-Novo, X., Moreno, J. I., and Berrocal, J. (2023). Ontology-based approach to real-time risk management and cyber-situational awareness. *Future Generation Computer Systems*, 141:462–472.

[22] Vickery, B. C. (1997). Ontologies. *Journal of information science*, 23(4):277–286.

[23] Wong, E. and Kolter, J. Z. (2018). Provable defenses against adversarial examples via the convex outer adversarial polytope.

[24] Xiao, H., Biggio, B., Nelson, B., Xiao, H., Eckert, C., and Roli, F. (2015). Support vector machines under adversarial label contamination. *Neurocomputing*, 160:53–62.

[25] Yeboah-Ofori, A., Ismail, U. M., Swidurski, T., and Opoku-Boateng, F. (2021). Cyber threat ontology and adversarial machine learning attacks: Analysis and prediction perturbance. In *2021 International Conference on Computing, Computational Modelling and Applications (ICCMA)*, pages 71–77.