University of Piraeus
School of Information and Communication Technologies
Department of Digital Systems

M.Sc. Digital Systems Security

Master Thesis

Threat Categorization on CVE Descriptions Using Text Classification

Giannakopoulos Thrasyvoulos

Piraeus, 2022

University of Piraeus

School of Information and Communication Technologies

Department of Digital Systems

Tripartite Examination Committee

Supervisor

Gritzalis Stefanos

Professor, Department of Digital Systems, University of Piraeus

Members

Lambrinoudakis Konstantinos

Professor, Department of Digital Systems, University of Piraeus

Xenakis Christos

Professor, Department of Digital Systems, University of Piraeus

Acknowledgements

I would like to thank my supervisor, Professor Stefanos Gritzalis, for his continuous support and guidance during the pursuit of this thesis. I would also like to thank Professor Konstantinos Lambrinoudakis for his constant feedback and support in this undertaking and Professor Christos Xenakis for the knowledge bestowed upon my fellow students and myself in the postgraduate program he directs. I would also like to thank Professor George Vouros for his invaluable input in the machine learning aspects of this thesis.

My family and friends provided me with their unlimited support and encouragement through all my years of study and continued to do so during my postgraduate studies as well and I thank them very much.

I would also want to thank Explosion, a software company specializing in the development of AI and NLP tools, with spaCy being the most popular of them, for providing their Prodigy software for experimentation in manual CVE annotation.

Last but not least, I would like to thank the Systems Security Laboratory and Telecommunication Systems Laboratory members of the University of Piraeus, namely, Professor Athanasios Kanatas, Assistant Professor Konstantinos Maliatsos, Dr. Christos Lyvas, Dr. Viktor Nikolaidis and Andreas Menegatos, as well as Associate Professor Christos Kalloniatis from the Privacy Engineering and Social Informatics laboratory of the University of the Aegean. Their friendship and support while collaborating in research projects provided me with the perfect environment to both come up with the subject of my thesis, as well as follow through with its implementation.

Contents

Abstract

The goal of this thesis is to classify CVEs into potential threats based on the descriptions provided using text classification, to then be used in a vulnerability-based risk analysis for Cyber-Physical Systems and more specifically Intelligent Transportation Systems. Because not all CVEs provide related CWEs, or their CWE entries are too generic, such as NVD-CWE-Other, a mapping of threats to CWEs and CAPECs, as used by the CitySCAPE Project's Risk Modelling Tool, is not always possible. This thesis proposes using text classification in order to identify those threats based on CVEs that can be mapped to CWEs and CAPECs. An accuracy of over 90% was achieved for each of the 16 threats across 111,384 CVE entries computed on a 10-fold cross validation.

Keywords: Threats, CVE, CWE, CAPEC, Natural Language Processing, Text Classification

Περίληψη

Ο σκοπός της παρούσας διπλωματικής είναι η ταξινόμηση εγγραφών CVE σε πιθανές απειλές βάσει των περιγραφών που παρέχονται με χρήση ταξινόμηση κειμένου, ώστε να μπορέσουν στην συνέχεια να χρησιμοποιηθούν σε μια μεθοδολογία ανάλυσης ρίσκου με βάση τις ευπάθειες για Κυβερνο-Φυσικά Συστήματα και πιο συγκεκριμένα σε Ευφυή Συστήματα Μεταφορών. Επειδή δεν έχουν όλες οι εγγραφές CVE σχετικά CWE, ή τα σχετικά CWE είναι πολύ γενικά, όπως NVD-CWE-Other, η αντιστοιχία απειλών σε CWE και CAPEC, όπως χρησιμοποιείται από το Risk Modelling Tool του έργου CitySCAPE, δεν είναι πάντοτε δυνατό να επιτευχθεί. Η παρούσα διπλωματική προτείνει την χρήση ταξινόμησης κειμένου για να αναγνωρίσει τις απειλές αυτές με βάση CVE που μπορούν να αντιστοιχιστούν με CWE και CAPEC. Επιτεύχθηκε μια ακρίβεια άνω του 90% για κάθε μια από τις 16 απειλές σε 111.384 εγγραφές CVE με χρήση επικύρωσης 10-fold cross validation.

Λέξεις κλειδιά: Απειλές, CVE, CWE, CAPEC, Επεξεργασία Φυσικής Γλώσσας, Ταξινόμηση Κειμένου

List of Figures

List of Tables

Acronyms

CPS	Cyber-Physical System
ITS	Intelligent Transportation System
CVE	Common Vulnerabilities and Exposures
CPE	Common Platform Enumeration
CWE	Common Weakness Enumeration
CAPEC	Common Attack Pattern Enumeration and Classification
CNA	CVE Numbering Authority
NVD	National Vulnerability Database
SCAP	Security Content Automation Protocol
CVSS	Common Vulnerability Scoring System
FFRDC	Federally Funded Research and Development Center
NCF	National Cybersecurity FFRDC
NIST	National Institute of Standards and Technology
CISA	Cybersecurity and Infrastructure Security Agency
DHS	Department of Homeland Security
NIAC	National Infrastructure Advisory Council
FIRST	Forum of Incident Response and Security Teams
NLP	Natural Language Processing
NER	Named Entity Recognition
ML	Machine Learning
AI	Artificial Intelligence
SGD	Stochastic Gradient Descent

1 Introduction

In today's modern interconnected era, many types of systems rely increasingly on computers. Such systems include Intelligent Transport Systems (ITS), Cyber-Physical Power System (CPPS), oil pipelines etc. They typically have a computerised aspect and a physical component that are linked to create what are called Cyber-Physical Systems (CPSs). These systems typically use Industrial Control Systems (ICS). Due to their nature and widespread adoption, CPSs can also affect critical infrastructures, so their protection is of great importance.

Cyber-Physical Systems Security isn't new and there have been plenty of attacks targeting them, with Stuxnet being one of the most famous malware targeting Iran's Natanz nuclear enrichment facility and recently a compromise of the Colonial Pipeline that led to a disruption of the US's oil supply. As such, a holistic risk analysis is important as it can identify such threats.

The City-level Cyber-Secure Multimodal Transport Ecosystem (CitySCAPE)[1] is a research program sponsored by the European Union's Horizon 2020 research & innovation program. One of its aims is to develop risk analysis techniques using a variety of different tools for multimodal transport. One such tool is the University of Piraeus's (UPRC) Risk Modelling Tool (RMT). Its aim is to find vulnerabilities for various ecosystems that are comprised of a multitude of components with different interconnections. Once those vulnerabilities are discovered, it finds the possible threats that can explore them, posing risks to the various components of the ecosystem, based on a relation of CWEs, CAPECs and Threats, and is then able to produce a risk score for that ecosystem. Those threats are derived from various publications and can affect ITSs. While working on the RMT, both the amount of unlabeled weaknesses, such as NVD-CWE-noinfom, and the time it takes to for NIST to publish and analyze new CVEs, as [1] also states, were discovered.

This thesis aims to solve both of those problems by identifying the threats that both new and unlabeled CVEs pose, as well as identify the threats of CVEs that cannot be mapped using CWE - CAPEC - Threat relations. Section 2 provides technical background for cybersecurity and natural language processing; Section 3 presents related works; Section 4 presents the

[1] `https://www.cityscape-project.eu/`

methodology that was followed; finally, Section 5 concludes this thesis.

2 Background

2.1 Cybersecurity related background

2.1.1 Common Vulnerabilities and Exposures (CVE)

Common Vulnerabilities and Exposures (CVE) is a program used for the identification and disclosure of cybersecurity vulnerabilities. Its initial proposal was in the 1999 paper titled "Towards a Common Enumeration of Vulnerabilities" [2] that would then lead to the CVE List on the same year. Launched by The MITRE Corporation, an organization that manages federally-funded research and development centers (FFRDCs), including the National Cybersecurity FFRDC (NCF). The MITRE Corporation owns the rights to CVE, making it freely available to everyone. Its usage in security advisories have further cemented the CVE Program's adoption [3]. Several U.S. Government agencies also recommend and use CVE like the National Institute of Standards and Technology (NIST) in "NIST Special Publication (SP) 800-51, Use of the CVE Vulnerability Naming Scheme" [3] or the Cybersecurity and Infrastructure Security Agency (CISA) of the Department of Homeland Security (DHS) in its Log4j Vulnerability Guidance [4]. CISA also currently sponsors the program, alongside NIST's National Vulnerability Database (NVD). CVEs are assigned from CVE Numbering Authorities (CNAs). Such authorities are partners to the CVE Program and include software vendors, bug bounty programs and others. As of March 31st, 2022, a total of 172,934 vulnerabilities have been disclosed using the CVE Program.

Each CVE has its tag in the beginning of the Description field. Below is a table of CVE tags found in CVEs as of writing, with the number of corresponding entries:

Tag	Entries
** RESERVED **	N/A
** REJECT **	10,618
** DISPUTED **	919
DISPUTED	2
** DISPUTED *	1
** UNVERIFIABLE **	5
** UNVERIFIABLE, PRERELEASE **	2
** UNSUPPORTED WHEN ASSIGNED **	108
VERSION NOT SUPPORTED WHEN ASSIGNED	5
** PRODUCT NOT SUPPORTED WHEN ASSIGNED **	6
Resolved	2
** SPLIT **	1

Table 1: CVE Tags

Note: Reserved CVEs are not in the NVD data feeds that were used to generate the table, so no information is available. Also, there is a slight discrepancy between the cve.org website and NISTs data feed, because cve.org is updated in real time, whereas NVD data feeds are updated daily.

As we can see from the table above, the most common CVE tags are Reject, Disputed and Unsupported when assigned. Rejected CVEs are entries where a CVE ID should not have been assigned. Such is the case when, for example, upon further inspection, the reported issue is not a vulnerability, or the researcher wants to keep the vulnerability private. Disputed entries occur when an authoritative source such as a vendor, coordinator or researcher, disputes the vulnerability [5]. Unsupported when assigned are entries where the vendor no longer supports the vulnerable product or version. Typically, the general Unsupported When Assigned is used, however there are a handful of examples where the unsupported aspect affects the product itself or a dated version [6]. Typically, vendors do not issue patches on End-of-Life (EOL) products, however in extreme cases vendors have issued patches to critical vulnerabilities. An example of this is Microsoft with Windows XP, following the WannaCrypt outbreak [7]. CVE also lists in its guidance other tags including Merge and Partial Duplicate, however we were unable to locate any entries. The single occurrence of a Split tag appeared in CVE-2005-2759. The merge tag indicates that there were multiple CVE IDs issued to the same vulnerability, whereas the Split tag indicates that a single CVE ID was issued when multiple should have been issued [5].

2.1.2 Common Platform Enumeration (CPE)

Common Platform Enumeration (CPE) is a standardization method for describing products, applications and hardware from vendors. Initially maintained by The MITRE Corporation, CPE has been transferred to NIST, who currently maintains it. Labeling it as part of their Security Content Automation Protocol (SCAP), NIST highlights its importance for usage within IT management tools, in order to actively monitor for new vulnerabilities for deployed products, applications and hardware [8]. Currently, multiple network and vulnerability scanning utilities report found products in CPE format. Such utilities include Nmap, Nessus and OpenVAS. In August of 2011, NIST released CPE Version 2.3, which is still in use today. There are three naming scemes for CPEs [9]:

1. Well-Formed CPE Name (WFN): It is defined as comma-separated attribute - value pairs as shown below:

   wfn:[attribute_1="value_1",attribute_2="value_2"]

   With the allowed attributes being:

   - part
   - vendor
   - product
   - version
   - update
   - edition

- language
- sw_edition
- target_sw
- target_hw
- other

An example representing the 64-bit version of Microsoft Windows 10 is shown below:

wfn:[part="o",vendor="microsoft",product="windows_10",target_hw="x64"]

2. Uniform Resource Identifier (URI) binding: Also associated with CPE 2.2, it is defined in a string as shown below:

   cpe:/part:vendor:product:version:update:edition:language

   Representing the same product as with WFN in URI binding format yields:

   cpe:/o:microsoft:windows_10:::~~~~x64~

   From the specification, and by looking at the above CPE, we can see that for future CPE versions the tilde character (~) is used to add in the missing fields that CPE 2.3 contains. Specifically it states that the tilde character is used to "pack" multiple attribute values into the edition component.

3. Formatted String Binding: It is most commonly used to describe CPE 2.3 entries. It is defined as shown below:

   cpe:<cpe_version>:<part>:<vendor>:<product>:<version>:<update>:<edition>:<language>:<sw_edition>:<target_sw>:<target_hw>:<other>

   The same product can be specified in Formatted String Binding as:

   cpe:2.3:o:microsoft:windows_10::*:*:*:*:*:x64:**

Asterisks denote wildcards and colons denote the field change. Colons can be delimited using a backslash (\). For example, in CVE-2016-0380 [2] the product sterling_connect:direct is used and is represented as:

cpe:2.3:a:ibm:sterling_connect\:direct:4.1.0.0:*:*:*:*:unix:*:*

2.1.3 Common Weakness Enumeration (CWE)

Common Weakness Enumeration (CWE) is a public list of weaknesses. It is managed by the Homeland Security Systems Engineering and Development Institute (HSSEDI) which is operated by The MITRE Corporation. Launched in 2006, it initially focused on software issues, however following the LoJax rootkit and the Meltdown/Spectre exploits, CWE included hardware weaknesses in 2020 [10]. CWEs are assigned to CVEs from NIST in their NVD database.

It has 6 main lists of weaknesses, namely:

- CWE-1000: Research Concepts (It contains all the weaknesses in the CWE list)
- CWE-1194: Hardware Design
- CWE-699: Software Development
- CWE-1003: Weaknesses for Simplified Mapping of Published Vulnerabilities
- CWE Top 25 Most Dangerous Software Weaknesses
- CWE Most Important Hardware Weaknesses

NVD analysts use CWE-1003 to categorize the weaknesses used in vulnerabilities as of 2016. It is a joint effort of NIST and the CWE Team, in order for analysts to be able to easily categorize the most common and impactful weaknesses into newly discovered vulnerabilities. As such, it is not a comprehensive list containing only 127 weaknesses, out of a total of 926,

[2] https://nvd.nist.gov/vuln/detail/CVE-2016-0380

as of writing [11]. NVD used CWE-635 from 2008 until 2016 when it was replaced by CWE-1003. It contained only 13 weaknesses [12]. Despite that, 316 weaknesses were found to have been in use in CVEs, more than double the amount that are supposed to be used during categorization. This increases the amount of manual mapping required and is another case where our methodology can assist in the threat categorization.

CWE and CWE Lists or Views are constantly updated, in order to be up-to-date with the current state of software and hardware weaknesses. For that reason, it uses a versioning system, with the latest version being version 4.7 as of writing.

The CWE Top 25 Most Dangerous Software Weaknesses is a yearly list that forms as a result of the most common and impactful weaknesses. Initially launched in 2009 and updated in 2010 and 2011, it remained the same until 2019 where the list was updated once more, and has been updated yearly ever since. It serves as a resource to inform users of the most important weaknesses. In order to generate the list, data from CVE and NVD were used. The two most important metrics for generating the CWE Top 25 were the number of CVE occurrences and the overall severity is used, with CVSS being the vulnerability severity metric [13].

CVE records can contain 0, 1 or more CWE entries, with the table below showing the distribution of the number of CWEs found in CVEs. Note that the total number of entries adds up to 183,492, which is the number of CVEs in the NVD JSON feeds as of March 31, 2022.

No. of CWE Entries	CVE Count	Percent of Entries
0	10,980	5.98%
1	169,368	92.30%
2	2,996	1.63%
3	128	0.07%
4	15	0.01%
5	5	0.00%

Table 2: Number of CWE entries in CVEs

The number of CVE entries that have zero CWEs assigned to them fluctuates, depending on the number of CVEs that have yet to be analyzed. There are however, a lot of old CVE entries that may never get assigned a CWE.

Below is a table containing the 15 most used CWE entries in CVEs.

CWE ID	CWE Name	No. of occurrences
NVD-CWE-Other		26,821
NVD-CWE-noinfo		19,335
CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	18,203
CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer	11,507
CWE-20	Improper Input Validation	8,746
CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	7,263
CWE-200	Exposure of Sensitive Information to an Unauthorized Actor	6,845
CWE-787	Out-of-bounds Write	5,802
CWE-264	Permissions, Privileges, and Access Controls	5,304
CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	4,274
CWE-125	Out-of-bounds Read	4,081
CWE-352	Cross-Site Request Forgery (CSRF)	3,472
CWE-94	Improper Control of Generation of Code ('Code Injection')	2,774
CWE-416	Use After Free	2,770
CWE-287	Improper Authentication	2,640

Table 3: 15 most used CWE entries

Note that there are two CWE entries that are not included in CWE Lists. They are NVD-CWE-Other and NVD-CWE-noinfo that are used by NVD analysts. These two special CWEs make up 25.15% of all CVE entries. Combined with the CVE entries that don't have a CWE assigned to them, they make up nearly a third of all CVEs that are impossible to analyze using CWE data alone. This was a key rationale for this thesis.

2.1.4 Common Attack Pattern Enumeration and Classification (CAPEC)

Common Attack Pattern Enumeration and Classification (CAPEC) is a list of common attack patterns that help users understand how the most common weaknesses are exploited, that in turn create vulnerabilities. Initially released in 2007, it is currently managed by The MITRE Corporation [14]. While there is no direct link between CVEs and CAPECs, each attack pattern contains related weaknesses in its information page. There are two CAPEC Views:

- CAPEC-1000: Mechanisms of Attack
- CAPEC-3000: Domains of Attack

Both CAPEC views contain all the current attack patterns, 546 as of writing, with the differences being in how they are categorized. The current version of CAPEC is 3.7.

2.1.5 National Vulnerability Database (NVD)

NIST's National Vulnerability Database (NVD) is an extention to the CVE Program. Introduced in 1999 as the Internet Category of Attack Toolkit (ICAT), it initially hosted attack scripts, before shifting its focus onto vulnerabilities with the first analysts being students of the SANS Institute (SysAdmin, Audit, Network, and Security). In 2004, ICAT received funding from DHS and in 2005 it was rebranded as NVD [15].

Both CVE and NVD are vulnerability databases that contain the same vulnerabilities. While CVE simply lists the vulnerability, a description, references and the assigning CNA, NVD performs an analysis of each vulnerability to extract relevant CPE names, CVSS scoring information as well as CWE mappings. Because all the extra information needs to be extracted from NVD analysts, there is usually a delay between the initial CVE publication and the NVD enrty to include other information such as CVSS scoring.

Due to the fact that NVD must analyze each CVE separately in order to be able to extract the relevant information, it can take some time between the initial CVE publication and the analysis to be available in the NVD database. Because of this, it is critical that vendors and users are aware of new vulnerabilities that have not yet been analyzed by NVD and understand their impact. It is for that reason that companies like Tenable have created solutions in order to get that relevant information from CVE descriptions using natural language processing and machine learning [1].

NVD provides both an API for making relatively fast queries for vulnerabilities related to a specific CPE, as well as JSON feeds that are updated daily. The JSON feeds from NVD were used in order to get all the information necessary to map and consequently train our model with the specific threats.

2.1.6 Common Vulnerability Scoring System (CVSS)

CVSS

Common Vulnerability Scoring System (CVSS) is a standardized system for assessing the severity of computer vulnerabilities. Initially created by the National Infrastructure Advisory Council (NIAC), CVSS v1 was released in February 2005, with the Forum of Incident Response and Security Teams (FIRST) chosen to be the custodian of CVSS in April 2005 [16]. In June of 2007, CVSS v2 was launched after feedback received from vendors. It reduced inconsistencies, increased granularity and reflected a wider variety of vulnerabilities [17]. CVSS v3.0 was released in June 2015 and brought forth several changes including:

- In the Base Metric Group Confidentiality, Integrity, and Availability metrics were changed from Low (L), Medium (M), High (H) to None (N), Low (L) or High (H).
- The Attack Vector (AV) metric added the Physical (P) metric value, to include vulnerabilities that require physical access, like cold boot attacks.

- The metric User Interaction (UI) was added, with 2 options, None (N) or Required (R).
- The Privileges Required (PR) metric was also added with the None (N), Low (L) or High (H) options, to reflect the privilege level the attacker must obtain in order to successfully exploit the vulnerability.

Finally Metric Severity Ratings depending on the Metric Score were changed:

CVSS v2 Ratings

Low	0.0 - 3.9
Medium	4.0 - 6.9
High	7.0 - 10.0

CVSS v3.0 Ratings

None	0.0
Low	0.1 - 3.9
Medium	4.0 - 6.9
High	7.0 - 8.9
Critical	9.0 - 10.0

Figure 1: CVSS v2 vs CVSS v3.0 Metric Severity Ratings

The current version, CVSS v3.1, was launched in June 17th, 2019 and adopted by NVD on September 10th, 2019. It focused on making clarifications and improvements to the 3.0 standard without making significant changes to the formulas [18]. The biggest change was a clarification in the CVSS specification that CVSS measures severity and not risk. That is because CVSS Base Score was solely used as a method to measure risk, while Base Score only takes into account the constant characteristics of vulnerabilities. Temporal and Environmental Metrics are recommended, that involve, among others, the exploit code maturity and the remediation level that can change across the timeline of a vulnerabilities discovery, and the specific required CIA requirements in the case of Environmental Metrics. However, NVD does not provide Temporal Metrics which makes using the metrics that the Specification suggests cumbersome, as they require manual intervention or an exhaustive search across CNAs. For example in CVE-2019-9516 NIST[3] provides its CVSS score based on its analysis, as well as the Computer Emergency Response Team (CERT) Coordination Center's (CERT/CC) CVSS score, with neither displaying any Temporal Metrics. However, in CERT/CCs report[4], Temporal Metrics are provided. NVD also does not provide the CVSS rating of the assigning CNA in its API[5].

[3] `https://nvd.nist.gov/vuln/detail/CVE-2019-9516`

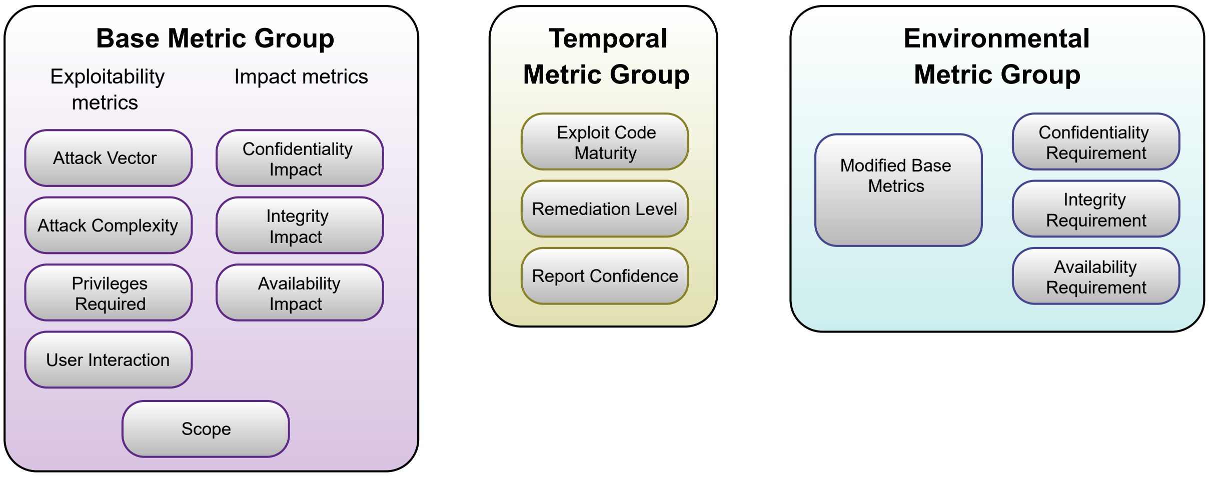
[4] `https://kb.cert.org/vuls/id/605641/`

[5] `https://nvd.nist.gov/developers/vulnerabilities`

2.1.6.1 CVSS v3.1 Specification

CVSS v3.1 is comprised of three Metric Groups:

- Base Metric Group
- Temporal Metric Group
- Environmental Metric Group



Source: `https://www.first.org/cvss/v3.1/specification-document`

Figure 2: CVSS v3.1 Metric Groups

The Base Metric Group represents characteristics of a vulnerability that remain unaltered through its lifetime, such as how it is exploited (Exploitability Metrics), the impact it has (Impact Metrics), and whether it impacts components outside of its security scope (Scope). The following table displays the metrics that make up the Base Metric Group, as well as possible values:

Base Metric Group	Exploitability Metrics	Attack Vector (AV)	Network (N)
			Adjacent (A)
			Local (L)
			Physical (P)
		Attack Complexity (AC)	Low (L)
			High (H)
		Privileges Required (PR)	None (N)
			Low (L)
			High (H)
		User Interaction (UI)	None (N)
			Required (R)
	Impact Metrics	Confidentiality (C)	None (N)
			Low (L)
			High (H)
		Integrity (I)	None (N)
			Low (L)
			High (H)
		Availability (A)	None (N)
			Low (L)
			High (H)
	Scope		Unchanged (U)
			Changed (C)

Table 4: Base Metric Group metrics and possible values

CVSS Base Score is calculated using these metrics. Specifically, Scope impacts both how the Impact sub score is calculated, as well as how the the Base Score is calculated. The Exploitability subscore is not affected by the Scope metric for the Base Score calculations.

The Temporal Metric Group contains metrics related to the current vulnerability status, including exploit availability status, available patches and confidence in the vulnerability existence. The following table displays the metrics that make up the Temporal Metric Group, as well as possible values:

Temporal Metric Group	Exploit Code Maturity (E)	Not Defined (X)
		High (H)
		Functional (F)
		Proof-of-Concept (P)
		Unproven (U)
	Remediation Level (RL)	Not Defined (X)
		Unavailable (U)
		Workaround (W)
		Temporary Fix (T)
		Official Fix (O)
	Report Confidence (RC)	Not Defined (X)
		Confirmed (C)
		Reasonable (R)
		Unknown (U)

Table 5: Temporal Metric Group metrics and possible values

Finally, there is the Environmental Metric Group that enables analysts to modify the CVSS score, by taking into account the requirements in terms of Confidentiality, Integrity and Availability, as well as the modifications existing security controls may have on the vulnerabilities. The table below presents the metrics in the Environmental Metric Group, along with possible values:

Environmental Metric Group	Security Requirements	Confidentiality Requirement (CR)	Not Defined (X)
			Low (L)
			Medium (M)
			High (H)
		Integrity Requirement (IR)	Not Defined (X)
			Low (L)
			Medium (M)
			High (H)
		Availability Requirement (AR)	Not Defined (X)
			Low (L)
			Medium (M)
			High (H)
	Modified Base Metrics	Modified Attack Vector (MAV)	Same as in Base Metric Group with the addition of the Not Defined (X) option.
		Modified Attack Complexity (MAC)	
		Modified Privileges Required (MPR)	
		Modified User Interaction (MUI)	
		Modified Scope (MS)	
		Modified Confidentiality (MC)	
		Modified Integrity (MI)	
		Modified Availability (MA)	

Table 6: Environmental Metric Group metrics and possible values

More information is available in the CVSS v3.1 Specification Document. [6]

[6] `https://www.first.org/cvss/v3.1/specification-document`

2.2 Natural Language Processing related background

Natural Language Processing or NLP refers to a branch of computer science that aims to make computers understand written and spoken words like a human would. It combines multiple fields of computer science including computational linguistics, machine learning, and deep learning models. There is a plethora of tasks that can be accomplished using NLP, such as named-entity recognition (NER), that identifies entities in sentences; sentiment analysis, that can extract subjective information; speech-to-text; text-to-speech; AI-powered translations; and the one that is of interest in this thesis: text classification.

2.2.1 Text Classification

Text classification is the process of categorizing text into certain categories. When a computer model classifies information, the result will fall into 4 categories of prediction correctness, depending on the predicted label and the actual label. Below is a confusion matrix that shows the different categories.

		True Label	
		Positive	Negative
Predicted Label	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Figure 3: Confusion Matrix of Classification Results

As we can see, when the predicted label matches the actual label, the result is classified as a true positive or a true negative, when the true label is positive or negative, respectively. Otherwise, it is classified as a false negative when the true label is positive and the predicted is negative, and as a false positive when the true label is negative and the predicted is positive. Using these categories we can extract several information and statistics about our model. The most common metric for classification models is accuracy, which is calculated using the formula below.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

Using the categories, that can be generated by the confusion matrix above this can also be represented as:

$$Accuracy = \frac{True\ Positives + True\ Negatives}{True\ Positives + True\ Negatives + False\ Positives + False\ Negatives}$$

Another common metric is precision, which is calculated using the formula below:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Along with precision, another common metric is recall. It is calculated using the formula below:

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Finally, another common metric is F1-score, the harmonic mean of precision and recall, which is defined as:

$$F1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

There are other metrics to measure the effectiveness of classification models, however, for the purposes of this thesis, only the ones mentioned above will be used. Because the output of the model outputs a probabilistic value between 0 and 1, a classification threshold is set to be able to distinguish between the two classifications, for the existence or not of a certain threat.

2.2.2 spaCy

spaCy

spaCy [7] is an open-source Python library focused on natural language processing (NLP). It is built and maintained by the Explosion company, and uses transformers for its deep learning models. It features, among other things:

- Named-entity recognition (NER)

[7]`https://spacy.io/`

- Text Classification
- Lemmatization

It was selected for its ease of use, without needing too much technical knowledge of NLP. During the initial research phases of this thesis, NER was considered in order to identify the threats, which required a manual annotation in order to train the ML model to recognize threats. Explosion also makes an annotation program called Prodigy which, alongside NLP, features computer vision annotation tools. Prodigy [8] was kindly provided for our research by Explosion, however the effort of finding examples of all the threats in CVEs and annotating a large enough set to use for training and validation, was deemed substantial. Through the CitySCAPE project, a mapping of threats with CWEs and CAPECs was generated for the RMT subcomponent. The existence of such a mapping shifted our focus to text classification, which would use the mapping that had already been made manually.

[8] `https://prodi.gy/`

3 Related Work

There have been several cases where NLP and ML techniques have been applied on CVE descriptions to extract information [1, 19, 20]. However, to our knowledge threat extraction as presented in this thesis has not been proposed. In this section we will briefly analyze these related works.

Tenable, the company that created Nessus, a popular vulnerability assessment tool, uses NLP on CVE descriptions as part of their vulnerability priority rating (VPR) [1]. VPR is a vulnerability impact tool that aims to more accurately analyze the impact vulnerabilities pose, as, according to Tenable, CVSS has a high proportion of High and Critical vulnerabilities, thus making prioritization difficult [21]. In their VPR tool, they use NLP in order to calculate CVSS impact metrics from CVE descriptions. Their reasoning for this approach is that it takes some time between a CVE disclosure and NVD publication, which can take more than 30 days in some cases.

The work of Sun et al. [20] is impressive, as it is capable of extracting information from sometimes lengthy ExploitDB posts. This includes the title, the description, metadata, etc. Using this information, they are able to generate CVE descriptions that include the vulnerable product, the vendor, the vulnerability type, the attack vector, the impact, etc. As is often the case with machine learning, a lot of annotation is required and in their case they annotated 765 posts. Given the number of annotations, as well as all the parameters that were annotated, we can assume that it must have been a very lengthy process. The final CVE descriptions closely resemble the ones that were published and contain very similar information.

Perhaps the most relevant compared to this thesis is the work of Kanakogi et al. [19]. Their proposed method uses NLP to identify relevant CAPEC Attack Patterns, based on CVE descriptions. Their rationale is that CVEs cannot always be mapped to the correct CAPEC, because they can't be traced through CWE, as there is no CWE - CAPEC relation. Their methodology compares the similarity of CAPEC descriptions with CVE descriptions in order to find their similarity, rank them based on it, and find the most plausible CAPEC.

4 Contribution

Our research for identifying threats in CVE descriptions began using NER, with an initial proof of concept being created. However, a combination of the substantial annotation effort needed, and the fact that a mapping for threats to CVEs was already under development for the CitySCAPE project, we opted to shift our focus to text classification, in order to identify threats in CVE descriptions. The mapping used for this thesis was a snapshot of the mapping used for the RMT subcomponent of the RITA engine of the CitySCAPE project. The mapping uses a combination of CVE - CWE and CWE - CAPEC relations to map CVEs to one or more threats. The reason that CAPECs aren't directly mapped to CVEs is due to the fact that CAPECs aren't mapped to CVEs, but rather CWEs. In the end, the mapped dataset contained 111,384 CVEs that had one or more threats assigned to them. In order not to confuse the model, only CVEs that had one or more threats were used, because unannotated CVEs would be automatically assigned as not containing any threats that would reduce our model's accuracy.

With that initial mapping in place, a program was written in Python 3 using the spaCy library to train a ML model to extract threats from CVE descriptions, using the mapping stated above. We conducted three experiments:

- Using CVE descriptions, without any modifications.
- Using CVE descriptions that had been stemmed using NLTK.
- Using CVE descriptions that had been lemmatized using spaCy.

Stemming and Lemmatization are both word normalization techniques [22] that are capable of reducing the inflectional forms of words. The difference in that stemming is a more unrefined process as it trims words in order to find the root word. Lemmatization, however, takes into account the context the word is used in. For example, as Manning et al. [22] state, "the word *saw* using stemming would become *s*, while lemmatization would return *see* or *saw*, depending on whether the original token was a verb or a noun". Because spaCy does not support stemming, the Natural Language Toolkit (NLTK) [9] was used to create file containing the

[9]https://www.nltk.org/

stemmed descriptions and the CVE ID. The hope was that using either stemmed or lemmatized descriptions would yield a better accuracy. All three experiments were run using k-fold cross-validation. Using 10 folds, the mapped dataset was split into 10 subsets and for each fold 1 subset was used for validation, while all the other subsets were used for training, with special care to ensure that all the data in the initial dataset were used for both training and validation. Training used compounded mini-batches found in spaCy, with an initial size of 4, a max size of 32 and a compounding rate of 1.001. Finally, mini-batch gradient descent was used as the optimising function with a 0.2 dropout rate. Optimization algorithms are ways to find the minimum or maximum of a function. In this case, the goal is to minimize loss or the error in the model. There are plenty of optimization algorithms, but the main are batch gradient descent, stochastic gradient descent (SGD) and mini-batch gradient descent. Batch gradient descent optimizes on the entire dataset at once requiring a lot of memory to run on large datasets, SGD optimizes for each item, while mini-batch gradient descent uses a batch of data to perform an update on, with the main advantage being a reduction in variance on the updates and faster runtime using matrix optimizations [23].

We noticed that stemming and lemmatization had a minimal effect on our model, with the differences being within margin of error for the different models. As such, only the results of the unaltered CVE descriptions will be presented below. Note that multiple threats can be identified per CVE.

Table 7, presented below, shows the accuracy of the predictions after a 10-fold cross-validation. Count is the number of CVEs containing a threat in the dataset, accuracy the average accuracy across the ten folds with its variance and standard deviation. Precision, recall and F1-score are the average values across the ten folds, while TP (True Positives), TN (True Negatives), FP (False Positives) and FN (False Negatives) are the results of the addition of the respectable values for the ten folds. The total number of CVEs used is 111,384.

From Table 7 and the tables with the statistics for each fold that are omitted in this thesis, we can extract some information about the accuracy of the predictions and information about our model. For example, threats TH-05, TH-06 and TH-08 show major overfitting, with TH-14, TH-19, TH-23 and TH-24 overfitting to a lesser extent. This is because there was a relatively small amount of samples for those threats for the model to train from. Overall accuracy was above 90% and presented minimal variance and standard deviation. A slight reduction in accuracy can be attributed to the fact that some CVE descriptions do not provide the same level of information as others.

Threat ID	Threat Name	Count	Accuracy	Variance	Standard Deviation	Precision	Recall	F1	TP	TN	FP	FN
TH-02	Denial of Service	41,116	0.9295	0.00000696	0.00263888	0.9212	0.8799	0.8999	35,292	68,242	3,034	4,816
TH-03	Modification of Information / Data Manipulation	6,134	0.9778	0.00000227	0.00150708	0.9583	0.6233	0.7550	3,816	105,094	167	2,307
TH-05	Interception of Information	362	0.9967	0.00000028	0.00053083	0	0	-	0	111,022	0	362
TH-06	Replay of Messages	64	0.9994	0.00000002	0.00014024	0	0	-	0	111,320	0	64
TH-08	Failures of Devices	3	1	0.00000000	0.00005748	0	0	-	0	111,381	0	3
TH-09	Failure of System	10,471	0.9676	0.00000595	0.00243832	0.8411	0.8014	0.8203	8,229	99,550	1,563	2,042
TH-11	Software Exploitation / Malicious Code Injection	76,222	0.9081	0.00001650	0.00406207	0.9554	0.9066	0.9303	68,344	32,799	3,198	7,043
TH-14	Device Modification	866	0.9960	0.00000023	0.00048226	0.9041	0.5437	0.6743	467	110,471	54	392
TH-19	Phishing Attacks	582	0.9973	0.00000019	0.00043329	0.8992	0.5638	0.6762	329	110,756	48	251
TH-21	Resource Exhaustion/Lack of resources	34,390	0.9230	0.00001194	0.00345562	0.9141	0.8247	0.8667	27,900	74,905	2,648	5,931
TH-23	Management Interface Compromise	517	0.9974	0.00000028	0.00052843	0.7469	0.6674	0.6917	339	110,759	116	170
TH-24	Unauthorized Access To Premises	862	0.9924	0.00000105	0.00102274	0.0800	0.0079	0.0144	15	110,517	6	846
TH-25	Abuse of Authorisation / Privilege Escalation	17,084	0.9271	0.00000443	0.00210532	0.7812	0.7291	0.7529	12,400	90,862	3,518	4,604
TH-26	Loss/Leakage of Information	15,908	0.9448	0.00000256	0.00159967	0.8560	0.7338	0.7897	11,558	93,676	1,954	4,196
TH-27	Abuse of Authentication	7,931	0.9716	0.00000285	0.00168843	0.8571	0.7251	0.7840	5,738	102,487	984	2,175
TH-28	Identity Theft	8,077	0.9716	0.00000266	0.00163010	0.8429	0.7508	0.7927	6,046	102,177	1,154	2,007

Table 7: Results after 10-fold cross validation

5 Conclusion

As we have discovered, threat identification based on vulnerabilities isn't always possible, since there is not always a link between CVEs to CWEs in order to be able to take advantage of CWE - CAPEC - Threat relations. Using text classification we were able to identify threats with an accuracy of over 90%. As new vulnerabilities are discovered, and the threat landscape evolves, more effort will be required in order to accurately perform a risk analysis with the proposed methodology in this thesis being a possible option, and in fact will be implemented into the CitySCAPE project's Risk Modelling Tool. Future work for this thesis may include manual annotation of threats in vulnerabilities in order to rely fully on text classification and reduce reliance on the predetermined list of weaknesses present in CWE.

References

[1] W. Tai, “How to Use VPR to Manage Threats Prior to NVD Publication,” Available at: https://www.tenable.com/blog/how-to-use-vpr-to-manage-threats-prior-to-nvd-publication (22/05/2020).

[2] D. Mann and S. Christey, “Towards a Common Enumeration of Vulnerabilities,” 1999. [Online]. Available: https://cve.mitre.org/docs/docs-2000/cerias.html

[3] CVE, “History,” Available at: https://www.cve.org/About/History [Accessed: 27-Apr-2022].

[4] CISA, “Apache Log4j Vulnerability Guidance,” Available at: https://www.cisa.gov/uscert/apache-log4j-vulnerability-guidance [Accessed: 27-Apr-2022].

[5] CVE, “CVE Numbering Authority (CNA) Rules,” Available at: https://www.cve.org/ResourcesSupport/AllResources/CNARules (05/03/2020).

[6] CVE, “Process for Assigning CVE IDs to End-of-Life (EOL) Products,” Available at: https://cve.mitre.org/cve/cna/CVE_Program_End_of_Life_EOL_Assignment_Process.html (11/12/2020).

[7] Microsoft Security Response Center, “Customer Guidance for WannaCrypt attacks,” Available at: https://msrc-blog.microsoft.com/2017/05/12/customer-guidance-for-wannacrypt-attacks/ (12/05/2017).

[8] NIST CSRC, “Common Platform Enumeration (CPE),” Available at: https://csrc.nist.gov/Projects/Security-Content-Automation-Protocol/Specifications/cpe (10/01/2022).

[9] B. Cheikes, D. Waltermire, and K. Scarfone, “Common Platform Enumeration: Naming Specification Version 2.3,” 2011. [Online]. Available: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=909010

[10] CWE, “About CWE,” Available at: https://cwe.mitre.org/about/index.html (13/03/2021).

[11] CWE, “CWE VIEW: Weaknesses for Simplified Mapping of Published Vulnerabilities,” Available at: https://cwe.mitre.org/data/definitions/1003.html (28/04/2022).

[12] CWE, "Weaknesses Originally Used by NVD from 2008 to 2016," Available at: https://cwe.mitre.org/data/definitions/635.html (28/04/2022).

[13] CWE, "2021 CWE Top 25 Most Dangerous Software Weaknesses," Available at: https://cwe.mitre.org/top25/archive/2021/2021_cwe_top25.html (26/07/2021).

[14] CAPEC, "About CAPEC," Available at: https://capec.mitre.org/about/index.html (04/04/2019).

[15] NVD, "A Brief History of the NVD," Available at: https://nvd.nist.gov/general/brief-history [Accessed: 27-Apr-2022].

[16] FIRST, "Common Vulnerability Scoring System v1 Archive," Available at: https://www.first.org/cvss/v1/ (14/04/2005).

[17] FIRST, "New version of Common Vulnerability Scoring System released," Available at: https://www.first.org/cvss/v2/ (20/06/2007).

[18] FIRST, "Common Vulnerability Scoring System version 3.1: User Guide," Available at: https://www.first.org/cvss/user-guide [Accessed: 5-May-2022].

[19] K. Kanakogi, H. Washizaki, Y. Fukazawa, S. Ogata, T. Okubo, T. Kato, H. Kanuka, A. Hazeyama, and N. Yoshioka, "Tracing CVE Vulnerability Information to CAPEC Attack Patterns Using Natural Language Processing Techniques," *Information*, vol. 12, no. 8, 2021. [Online]. Available: https://www.mdpi.com/2078-2489/12/8/298

[20] J. Sun, Z. Xing, H. Guo, D. Ye, X. Li, X. Xu, and L. Zhu, "Generating Informative CVE Description From ExploitDB Posts by Extractive Summarization," 2021. [Online]. Available: https://arxiv.org/abs/2101.01431

[21] W. Tai, "What Is VPR and How Is It Different from CVSS?" Available at: https://www.tenable.com/blog/what-is-vpr-and-how-is-it-different-from-cvss (16/04/2020).

[22] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[23] S. Ruder, "An overview of gradient descent optimization algorithms," 2016. [Online]. Available: https://arxiv.org/abs/1609.04747