Understanding and Using Impact so you know what Vulnerabilities to fix first

aka Using AI Language Models for CyberSecurity like a pro

18 May 2024

https://www.bsidesdub.ie/

About Me

Chris Madden

Yahoo Paranoids Product Security Engineer

Chris has worked as a software engineer and system architect building secure trustworthy software at scale for embedded and cloud for more than 20 years.

He's not big on titles, hierarchy, status quo, or hype.

He's big on analysis and validation and understanding things deeply - using data analysis and dumb questions to build that understanding.

https://www.linkedin.com/in/chrisamadden

yahoo!

Are you an Ernie or a Bert?

Ernie **Bert**

https://www.billboard.com/wp-content/uploads/2023/01/bert-and-ernie-song-association-elle-billboard-1548.jpg

Abstract

Understanding and Using Impact so you know what Vulnerabilities to fix first

Organizations struggle with an increasing number of vulnerabilities. Prioritizing what to fix first by Risk is the solution.

In this talk, we'll explore the Impact aspect of Risk by walking through:

1. where Impact fits in the overall Risk picture
2. how Impact is under-represented in the standards today
3. the taxonomy of Impacts to use and why

4. state-of-the-art Natural Language Processing tools and how to use them for CyberSecurity
5. how to build and validate a model to generate the Impact labels from a vulnerability description
6. how to use all this to effectively prioritize your vulnerabilities by Impact and Risk

In this talk, you'll learn how you can:

- **leverage Impact for effective Risk Based Prioritization.**
- **use state-of-the-art Natural Language Processing tools for CyberSecurity**

This is a follow on to:

- https://riskbasedprioritization.github.io/ Guide released March 2024
- the BSides 2023 "Understanding your vulnerability data to optimize your DevOps pipeline flow" talk, where the overall Risk picture was developed and a Risk Based Prioritization scheme was implemented (that prioritized by Likelihood of Exploitation)

Contributions to the Industry

1. **A Taxonomy for Impact, validated against reality (all published CVEs to date)**
   a. **https://github.com/theparanoids/PrioritizedRiskRemediation/**
2. **A dataset of all published CVEs to date labelled by Impact according to the taxonomy**
   a. **https://huggingface.co/yahoo-inc/cve-impacts with the dataset e.g.**

cveID	Description	Extracted Vulnerability/ Impact text	Extracted Vulnerability/Impact text, Start of text, End of text, CAPEC Technical Impact, CAPEC ID, CWE ID
CVE-2019 -0211	In Apache HTTP Server 2.4 releases 2.4.17 to 2.4.38, with MPM event, worker or prefork, code executing in less-privileged child processes or threads (including scripts executed by an in-process scripting interpreter) could **execute arbitrary code** with the **privileges of the parent process** (usually root) by manipulating the scoreboard. Non-Unix systems are not affected.	**execute arbitrary code, privileges of the parent process**	[{"value":"**execute arbitrary code**", "start": 100, "end":130, "Impact":" **execute unauthorized code or commands**", "CAPEC":123, "CWE":456}, {"value":"**privileges of the parent process**", "start": 150, "end":170, "Impact": " **gain privileges/assume identity**", "CAPEC":123, "CWE":456}]

3. **A model that labels a text description with one or more Impacts (as defined in the Impact Taxonomy).**
   a. **https://huggingface.co/yahoo-inc with a repo with the model binary and configs and Readme.md**
      - Values assigned are dummy values
      - "start" and "end" identify the location by characters of the text; this supports training NERs.
      - The CWE ID and CAPEC ID values are those associated with the Extracted Vulnerability, Impact (not those pre-assigned to the CVE)
      - label is per the defined Impact Taxonomy

Dataset

Dataset Viewer — Auto-converted to Parquet · API · View in Dataset Viewer

Split (1)
train · 230k rows

Search this dataset

CVE string · lengths 13 16	Description string · lengths 20 3.95k	KeyPhrases string · lengths 2 343
CVE-1999-0002	Buffer overflow in NFS mountd gives root access to remote attackers, mostly in Linux systems.	['buffer overflow', 'root access']
CVE-1999-0003	Execute commands as root via buffer overflow in Tooltalk database server (rpc.ttdbserverd).	['buffer overflow', 'execute commands']
CVE-1999-0004	MIME buffer overflow in email clients, e.g. Solaris mailtool and Outlook.	['mime buffer overflow']
CVE-1999-0005	Arbitrary command execution via IMAP buffer overflow in authenticate command.	['imap buffer overflow', 'arbitrary commands', 'arbitrary command execution']
CVE-1999-0006	Buffer overflow in POP servers based on BSD/Qualcomm's qpopper allows remote attackers to gain root access using a long PASS command.	['gain root access', 'gain root access using a long PASS command', 'buffer overflow']
CVE-1999-0007	Information from SSL-encrypted sessions via PKCS #1.	[]

‹ Previous 1 2 3 ... 2,296 Next ›

CVE_KeyPhrases

CVE_KeyPhrases is a dataset of published CVEs with the Key Risk Phrases (for Impact, Weakness, Attack) extracted.

- It is released under license": "cc-by-sa-4.0"
- Please see the BSides Dublin 2024 presentation video and deck.

The dataset includes:

- ~230K published CVEs (excluding those marked Rejected) i.e. all CVEs up to April 3 2024 NVD Published date.
- The CVE ID, Description text, and Key Risk Phrases

As of April 2024, CVE_KeyPhrases stands as the largest dataset of CVEs with extracted Key Risk Phrases with high Precision (how many retrieved items are relevant?) and Recall (how many relevant items are retrieved?).

- It's an initial release and will be improved and enriched over time.

https://huggingface.co/datasets/yahoo-inc/CVE_KeyPhrases/cve-impacts

Risk is per Asset and depends on the Impact of a Vulnerability being exploited by a Threat

Risk Definition from NIST Special Publication 800-30 r1 Guide for Conducting Risk Assessments

Risk Based Prioritization Context - Content

Risk per Vulnerability

Understanding and Using the building blocks.

Understanding Your Vulnerability Data To Optimize Your DevOps Pipeline Flow by Chris Madden, BSides Dublin 2023 with a Taxonomy

AKA Chris tries to understand Risk and the Vulnerability Management landscape to optimize flow of s/w, and risk.

RBP for the masses

https://riskbasedprioritization.github.io/ or riskbasedprioritization.com March, 2024

Vulnerability Prioritization Options

Now that we really understand Risk (Exploitation and Impact), let's understand what we do with this info.

Vulnerability Prioritization Options - what data sources to use, and how to prioritize with them, Chris Madden, CERT Vendor Meeting, May 6 2024

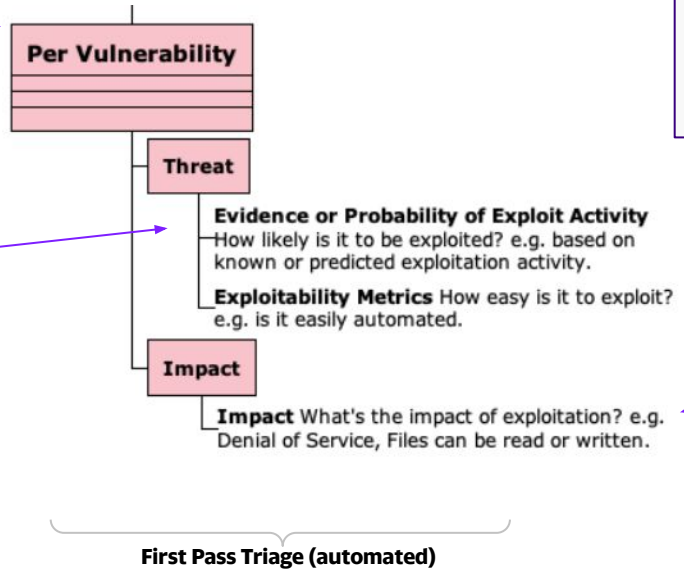
AKA Chris gives a user-centric view of the value of SSVC.

EPSS Likelihood of Exploitation

EPSS for the masses.

Exploit Prediction Scoring System (EPSS) - The User Guide by Chris Madden, BSides Dublin 2024, May 18

AKA Chris tries to help others understand EPSS and how to use it.



Impact

For the subset prioritized by Likelihood of Exploitation, focus on Technical Impacts that are most relevant to you.

Understanding and Using Impact so you know what Vulnerabilities to fix first by Chris Madden, BSides Dublin 2024, May 18

AKA Chris tries to understand the Technical Impact part of Risk, and learns NLP (Natural Language Processing) and LMs (Language Models) to extract the impact text from the 230K published CVEs Descriptions.

Slides here:
https://riskbasedprioritizati

Risk is per Asset and depends on the Impact of a Vulnerability being exploited by a Threat

https://github.com/theparanoids/PrioritizedRiskRemediation Risk Definition from NIST Special Publication 800-30 r1 Guide for Conducting Risk Assessments

User Need

User Need

Frank Siepmann · 1st
Cybersecurity Executive, CISO, Executive Advisor

A question for the vulnerability management wizards (**Wade Baker, Ph.D.** , **Chris H.** and **Patrick Garrity** 🐙🛹💙 come to mind) or anyone else that I forgot to include:

Is there any machine readable version of the "technical" impact of a vulnerability that is standardized and grouped?

For example:

- Allows for privilege escalation
- Allows for application data manipulation
- Allows to run shell code at the user level

Thank you!

User Need

Chris, you must be prescient. I came across this problem yesterday and thought, "Man, it would be nice to be able to extract only the impactful terms from CVE descriptions."
JosephM, Apr 5 2024, EPSS SIG

There's a user need, and there isn't a good solution...

linkedin thread Oct 2023 with some of the big guns replying

The Pitch

Problem

- Our industry relies heavily on accurate, timely CVE (vulnerability) data to inform risk.

Currently

- CVEs are manually enriched
- The enrichment data is not as useful as it could be - because it isn't there, or has typos, errors, inconsistencies, or does not allow meaningful prioritization.

Solution

- **Prioritize vulnerabilities**
  - by **Exploitation** (as recommended by CISA, Gartner) i.e. are being exploited in the wild, or are more likely to be exploited, significantly reduces the
    - cost of vulnerability management
    - risk by reducing the time adversaries have access to vulnerable systems they are trying to exploit
  - by **Impact** (as recommended by MITRE) allows for additional independent cost and risk reduction
- **Classify the vulnerability impact for a CVE** or other vulnerability description automatically at scale
  - so **you can prioritize by impact** as part of your Risk Based Prioritization resulting in a significant
    - reduction in your security effort
    - improvement in your security posture by remediating the higher risk vulnerabilities first
- **Provide a dataset** (the first in the industry of this scale) to allow related solutions (LM-based) to be built by anyone i.e. this addresses the hard problem of getting a labeled dataset; (training models is easy once you have a good dataset).

Why Now

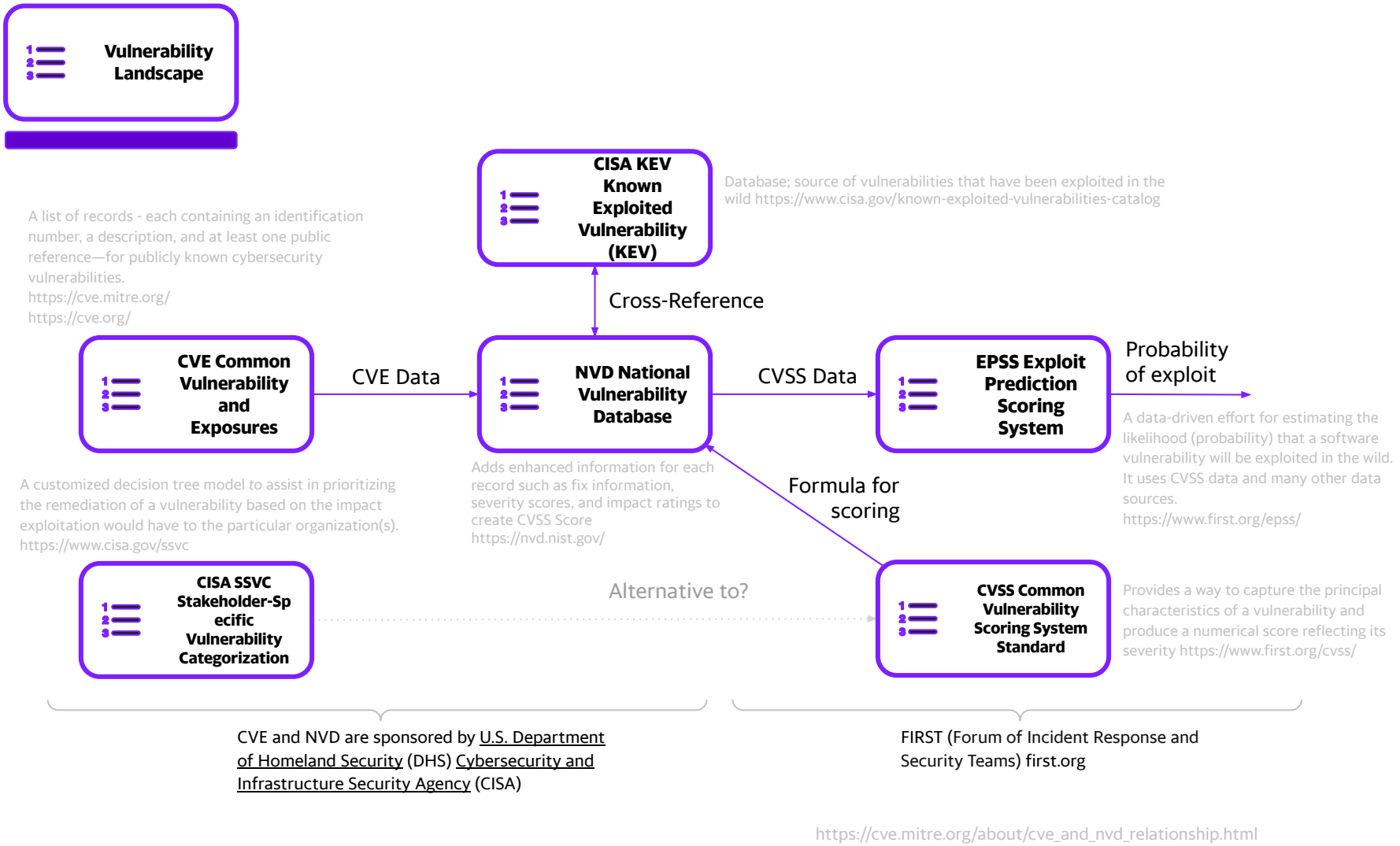
- There is an exponential increase in published CVEs
- There is a reduction in the human resources to process them; recent NVD enrichment disruption
- The tools to process text (Natural Language) have, in the last 3 years, entered a new generation: Language Models

Are you in?

NIST National Vulnerability Database Disruption Sees CVE Enrichment on Hold, Mar 15 2024

NVD slowdown leaves thousands of vulnerabilities without analysis data, Mar 22 2024

Vulnerability Landscape


Vulnerability Landscape
CISA KEV Known Exploited Vulnerability (KEV)
Database; source of vulnerabilities that have been exploited in the wild https://www.cisa.gov/known-exploited-vulnerabilities-catalog
A list of records - each containing an identification number, a description, and at least one public reference—for publicly known cybersecurity vulnerabilities.
https://cve.mitre.org/
https://cve.org/
Cross-Reference
CVE Common Vulnerability and Exposures
CVE Data
NVD National Vulnerability Database
CVSS Data
EPSS Exploit Prediction Scoring System
Probability of exploit
A data-driven effort for estimating the likelihood (probability) that a software vulnerability will be exploited in the wild. It uses CVSS data and many other data sources.
https://www.first.org/epss/
Adds enhanced information for each record such as fix information, severity scores, and impact ratings to create CVSS Score
https://nvd.nist.gov/
Formula for scoring
A customized decision tree model to assist in prioritizing the remediation of a vulnerability based on the impact exploitation would have to the particular organization(s).
https://www.cisa.gov/ssvc
CISA SSVC Stakeholder-Specific Vulnerability Categorization
Alternative to?
CVSS Common Vulnerability Scoring System Standard
Provides a way to capture the principal characteristics of a vulnerability and produce a numerical score reflecting its severity https://www.first.org/cvss/
CVE and NVD are sponsored by U.S. Department of Homeland Security (DHS) Cybersecurity and Infrastructure Security Agency (CISA)
FIRST (Forum of Incident Response and Security Teams) first.org
https://cve.mitre.org/about/cve_and_nvd_relationship.html

Vulnerability Landscape

"CWE is the root mistake, which can lead to a vulnerability (tracked by CVE in some cases when known), which can be exploited by an attacker (using techniques covered by CAPEC)", which can lead to **a Technical Impact (or consequence),** which can result in a Business Impact

- *"CWE focuses on a type of mistake that, in conditions where exploits will succeed, could contribute to the introduction of vulnerabilities within that product."*
- *"A vulnerability is an occurrence of one or more weaknesses within a product, in which **the weakness can be used by a party to cause the product to modify or access unintended data, interrupt proper execution, or perform actions that were not specifically granted** to the party who uses the weakness."*

Vulnerability

Log4 Shell

CWE-917 Improper Neutralization of Special Elements used in an Expression Language Statement ('Expression Language Injection')

CWE-20, CWE-502

CWE-400

CVE-2021-44228

Inject Unexpected Items

Remote Code Execution to read confidential data

Crypto miners, ransomware, backdoors,...

Standard

CWE Common Weakness Enumeration

A community-developed list of software and hardware weakness types. It serves as a common language, a measuring stick for security tools, and as a baseline for weakness identification, mitigation, and prevention efforts.
https://cwe.mitre.org/

CVE Common Vulnerability and Exposures

A list of records - each containing an identification number, a description, and at least one public reference—for publicly known cybersecurity vulnerabilities.
https://cve.mitre.org/
https://cve.org/

CAPEC Common Attack Pattern

Common Attack Pattern Enumeration and Classification Understanding how the adversary operates is essential to effective cybersecurity. CAPEC helps by providing a comprehensive dictionary of known patterns of attack employed by adversaries to exploit known weaknesses in cyber-enabled capabilities.
https://capec.mitre.org/

???

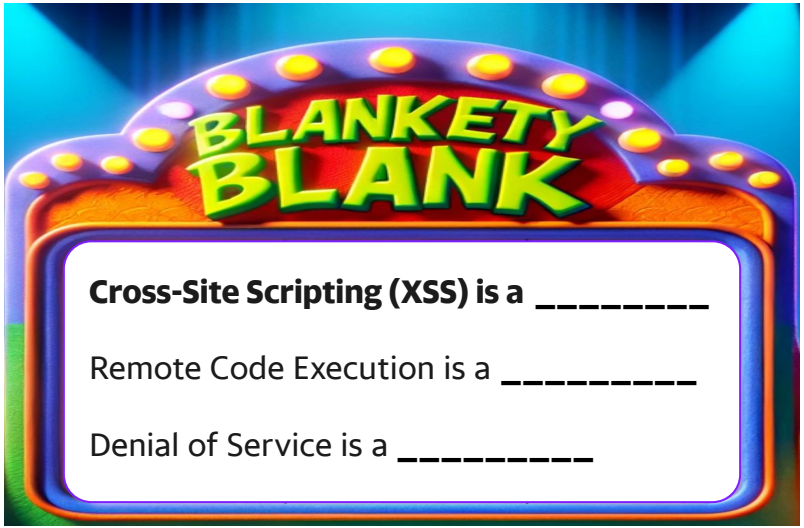
???

?
Common Weakness Scoring System (CWSS)
https://cwe.mitre.org/cwss/cwss_v1.0.1.html
2014

from MITRE.org

As a user/defender, I care most about these

Quick Quiz

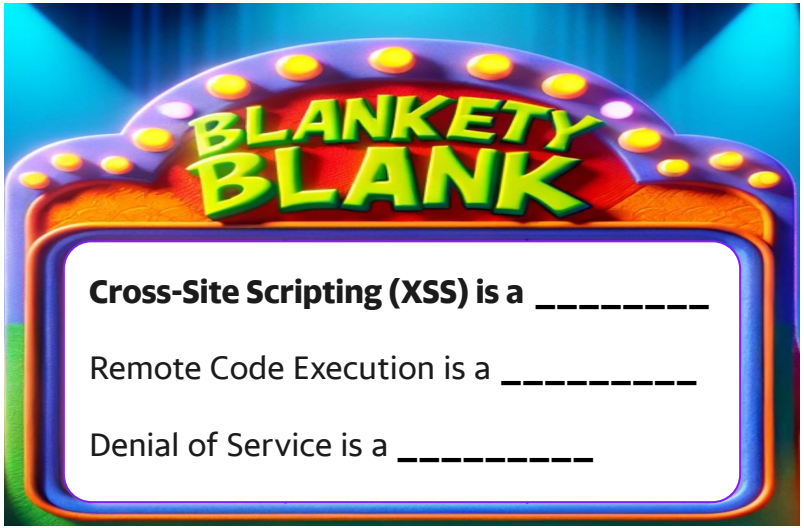


1. Weakness
2. Attack Pattern
3. Impact

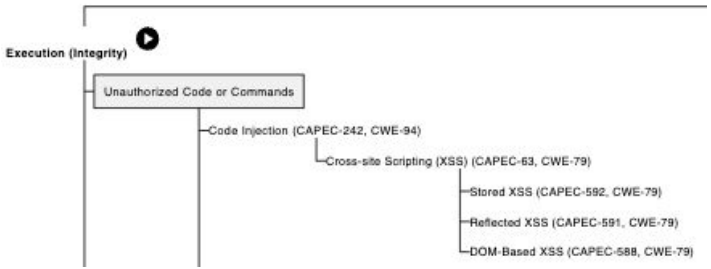
Answer now...

https://capec.mitre.org/about/new_to_capec.html

Quick Quiz



1. Weakness
2. Attack Pattern
3. Impact

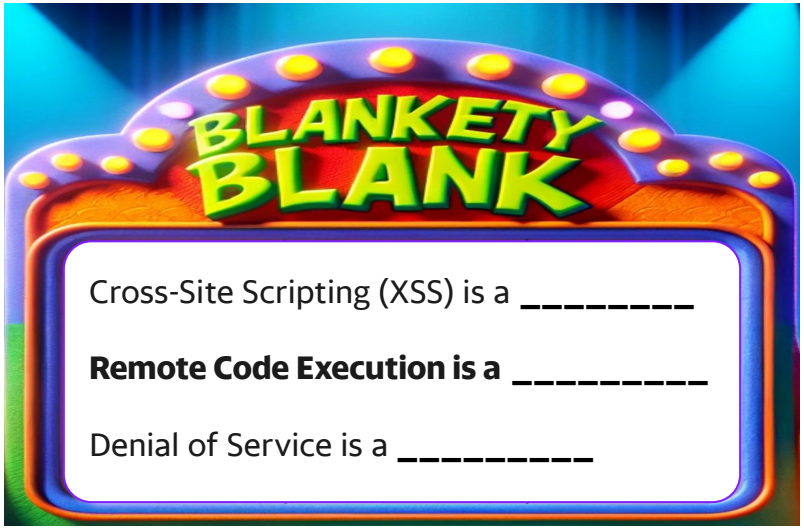


1. Weakness CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
2. Impact Execution Integrity - Unauthorized Code or Commands
3. **Attack Pattern CAPEC-63: Cross-Site Scripting (XSS)**

Cross-Site Scripting (XSS) is an Attack Pattern (but has a CWE/weakness associated with it)

https://capec.mitre.org/about/new_to_capec.html

Quick Quiz



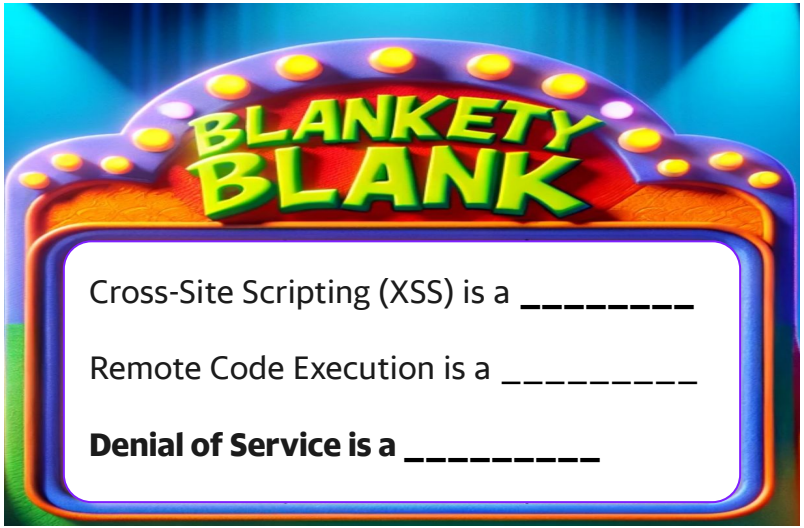
1. Weakness
2. Attack Pattern
3. Impact

1. Weakness: CWE-94: Improper Control of Generation of Code ('Code Injection')
2. **Impact Execution Integrity - Unauthorized Code or Commands**
3. Attack Pattern: CAPEC-242 Code Injection

Remote Code Execution is an Impact

https://capec.mitre.org/about/new_to_capec.html

Quick Quiz



1. Weakness
2. Attack Pattern
3. Impact

1. Weakness: CWE-400: Uncontrolled Resource Consumption
2. **Impact: Denial Of Service (Availability)**
3. Attack Pattern: CAPEC-469: HTTP DoS

Denial of Service is an Impact (but has a CAPEC/Attack Pattern, and CWE/Weakness associated with it)

CWE-400: Uncontrolled Resource Consumption

https://capec.mitre.org/about/new_to_capec.html

Example Log4Shell

Description

Apache Log4j2 2.0-beta9 through 2.15.0 (excluding security releases 2.12.2, 2.12.3, and 2.3.1) JNDI features used in configuration, log messages, and parameters do not protect against attacker controlled LDAP and other JNDI related endpoints. An attacker who can control log messages or log message parameters can execute arbitrary code loaded from LDAP servers when message lookup substitution is enabled. From log4j 2.15.0, this behavior has been disabled by default. From version 2.16.0 (along with 2.12.2, 2.12.3, and 2.3.1), this functionality has been completely removed. Note that this vulnerability is specific to log4j-core and does not affect log4net, log4cxx, or other Apache Logging Services projects.

Weakness

Technical Impact

Weakness Enumeration

CWE-ID	CWE Name	Source
CWE-917	Improper Neutralization of Special Elements used in an Expression Language Statement ('Expression La	NIST
CWE-502	Deserialization of Untrusted Data	Apache Software Foundation
CWE-400	Uncontrolled Resource Consumption	Apache Software Foundation
CWE-20	Improper Input Validation	Apache Software Foundation

Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/**C:H/I:H/A:H**
Confidentiality, Integrity, Availability all High

This CVE is in CISA's Known Exploited Vulnerabilities Catalog

Reference CISA's BOD 22-01 and Known Exploited Vulnerabilities Catalog for further guidance and requirements.

Vulnerability Name	Date Added	Due Date	Required Action
Apache Log4j2 Remote Code Execution Vulnerability	12/10/2021	12/24/2021	For all affected software assets for which updates exist, the only acceptable remediation actions are: 1) Apply updates; OR 2) remove affected assets from agency networks. Temporary mitigations using one of the measures provided at https://www.cisa.gov/uscert/ed-22-02-apache-log4j-recommended-mitigation-measures are only acceptable until updates are available.

"Execute Arbitrary Code" is the most important part for many users

https://nvd.nist.gov/vuln/detail/CVE-2021-44228

Example SpringShell

Description

A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. The specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it.

Weakness

Technical Impact

Weakness Enumeration

CWE-ID	CWE Name	Source
CWE-94	Improper Control of Generation of Code ('Code Injection')	NIST VMware

Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/**C:H/I:H/A:H**
Confidentiality, Integrity, Availability all High

This CVE is in CISA's Known Exploited Vulnerabilities Catalog

Reference CISA's BOD 22-01 and Known Exploited Vulnerabilities Catalog for further guidance and requirements.

Vulnerability Name	Date Added	Due Date	Required Action
Spring Framework JDK 9+ Remote Code Execution Vulnerability	04/04/2022	04/25/2022	Apply updates per vendor instructions.

"Remote Code Execution" is the most important part for many users

https://nvd.nist.gov/vuln/detail/cve-2022-22965

Example CVE-2022-42475

Current Description

A heap-based buffer overflow vulnerability [CWE-122] in FortiOS SSL-VPN 7.2.0 through 7.2.2, 7.0.0 through 7.0.8, 6.4.0 through 6.4.10, 6.2.0 through 6.2.11, 6.0.15 and earlier and FortiProxy SSL-VPN 7.2.0 through 7.2.1, 7.0.7 and earlier may allow a remote unauthenticated attacker to execute arbitrary code or commands via specifically crafted requests.

Weakness

Technical Impact

Weakness Enumeration

CWE-ID	CWE Name	Source
CWE-787	Out-of-bounds Write	NIST
CWE-197	Numeric Truncation Error	Fortinet, Inc.

Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/**C:H/I:H/A:H**
Confidentiality, Integrity, Availability all High

This CVE is in CISA's Known Exploited Vulnerabilities Catalog

Reference CISA's BOD 22-01 and Known Exploited Vulnerabilities Catalog for further guidance and requirements.

Vulnerability Name	Date Added	Due Date	Required Action
Fortinet FortiOS Heap-Based Buffer Overflow Vulnerability	12/13/2022	01/03/2023	Apply updates per vendor instructions.

"Execute arbitrary code or commands" is the most important part for many users

https://nvd.nist.gov/vuln/detail/CVE-2022-42475

Example CVE-2009-1936

Weakness

Technical Impact

Description

_functions.php in cpCommerce 1.2.x, possibly including 1.2.9, sends a redirect but does not exit when it is called directly, which allows remote attackers to bypass a protection mechanism to conduct remote file inclusion and directory traversal attacks, execute arbitrary PHP code, or read arbitrary files via the GLOBALS[prefix] parameter, a different vector than CVE-2003-1500.

Confidentiality, Integrity, Availability Impact not defined.

Weakness Enumeration

CWE-ID	CWE Name	Source
CWE-20	Improper Input Validation	NIST

1 Weakness, 5 Technical Impacts. Other CWEs could / should be added for these Impacts?

https://nvd.nist.gov/vuln/detail/CVE-2009-1936

What's the most important Vulnerability info?

CISA Known Exploited Vulnerabilities (KEV)

CISA CyberSecurity Advisories (CSA)

CISA Known Exploited Vulnerabilities (KEV)

CISA (Cybersecurity and Infrastructure Security Agency) maintains a source of vulnerabilities that have been **exploited in the wild** called the Known Exploited Vulnerabilities Catalog.

- This catalog contains ~1K CVEs.

CISA KEV provides a more succinct and consistent (Structured) description of these ~1K CVEs (relative to the ~220K CVEs in the NVD) e.g. CVE-2021-44228

CISA KEV

APACHE | LOG4J2

CVE-2021-44228

Apache Log4j2 Remote Code Execution Vulnerability

Apache Log4j2 contains a vulnerability where JNDI features do not protect against attacker-controlled JNDI-related endpoints, allowing for remote code execution.

NATIONAL VULNERABILITY DATABASE

Description

Apache Log4j2 2.0-beta9 through 2.15.0 (excluding security releases 2.12.2, 2.12.3, and 2.3.1) JNDI features used in configuration, log messages, and parameters do not protect against attacker controlled LDAP and other JNDI related endpoints. An attacker who can control log messages or log message parameters can execute arbitrary code loaded from LDAP servers when message lookup substitution is enabled. From log4j 2.15.0, this behavior has been disabled by default. From version 2.16.0 (along with 2.12.2, 2.12.3, and 2.3.1), this functionality has been completely removed. Note that this vulnerability is specific to log4j-core and does not affect log4net, log4cxx, or other Apache Logging Services projects.

CISA KEV provides a succinct and consistent summary of CVEs that includes Impact.
It can be automatically extracted for all CISA KEV CVEs (with minor manual post processing for some)

https://www.cisa.gov/known-exploited-vulnerabilities-catalog

https://nvd.nist.gov/vuln/detail/CVE-2021-44228

CISA KEV and NVD Descriptions

Description	CVE-2018-13379	CVE-2021-44228
CISA KEV vulnerabilityName	Fortinet FortiOS SSL VPN **Path Traversal** Vulnerability	Apache Log4j2 **Remote Code Execution** Vulnerability
CISA KEV shortDescription	Fortinet FortiOS SSL VPN web portal contains a **path traversal** vulnerability that may allow an unauthenticated attacker to **download** FortiOS **system files** through specially crafted HTTP resource requests.	Apache Log4j2 contains a vulnerability where JNDI features do not protect against attacker-controlled JNDI-related endpoints, allowing for **remote code execution**.
NVD Description Weakness Technical Impact	An Improper Limitation of a Pathname to a Restricted Directory (**"Path Traversal"**) in Fortinet FortiOS 6.0.0 to 6.0.4, 5.6.3 to 5.6.7 and 5.4.6 to 5.4.12 and FortiProxy 2.0.0, 1.2.0 to 1.2.8, 1.1.0 to 1.1.6, 1.0.0 to 1.0.7 under SSL VPN web portal allows an unauthenticated attacker to **download system files** via special crafted HTTP resource requests.	Apache Log4j2 2.0-beta9 through 2.15.0 (excluding security releases 2.12.2, 2.12.3, and 2.3.1) JNDI features used in configuration, log messages, and parameters do not protect against attacker controlled LDAP and other JNDI related endpoints. An attacker who can control log messages or log message parameters can **execute arbitrary code** loaded from LDAP servers when message lookup substitution is enabled. From log4j 2.15.0, this behavior has been disabled by default. From version 2.16.0 (along with 2.12.2, 2.12.3, and 2.3.1), this functionality has been completely removed. Note that this vulnerability is specific to log4j-core and does not affect log4net, log4cxx, or other Apache Logging Services projects.

CISA KEV vulnerabilityName summarizes a CVE description with the Impacts or Weaknesses

CISA Top Routinely Exploited Vulnerabilities (CSA)

CISA (Cybersecurity and Infrastructure Security Agency) co authors (with several international cybersecurity agencies) separate Cybersecurity Advisories (CSA) on the Top Routinely Exploited Vulnerabilities from the CISA KEV Catalog e.g.

1. AA23-215A Joint CSA 2022 Top Routinely Exploited Vulnerabilities August 2023
2. AA21-209A Joint CSA Top Routinely Exploited Vulnerabilities July 2021
3. AA22-279A 2022 covering CVEs from 2022, 2021
4. AA22-117A 2022 covering CVEs from 2021
5. AA20-133A 2020 covering CVEs from 2016 to 2019

These include a "Type" description that is the Impact
(except the first report AA20-133A (2016 to 2019).

.

Impact

Weakness

CVE	Vendor	Product	Type	CWE
CVE-2018-13379	Fortinet	FortiOS and FortiProxy	SSL VPN credential exposure	CWE-22 Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
CVE-2021-34473 (Proxy Shell)	Microsoft	Exchange Server	RCE	CWE-918 Server-Side Request Forgery (SSRF)
CVE-2021-31207 (Proxy Shell)	Microsoft	Exchange Server	Security Feature Bypass	CWE-22 Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
CVE-2021-34523 (Proxy Shell)	Microsoft	Exchange Server	Elevation of Privilege	CWE-287 Improper Authentication
CVE-2021-40539	Zoho ManageEngine	ADSelfService Plus	RCE/ Authentication Bypass	CWE-287 Improper Authentication
CVE-2021-26084	Atlassian	Confluence Server and Data Center	Arbitrary code execution	CWE-74 Improper Neutralization of Special Elements in Output Used by a Downstream Component ('Injection')
CVE-2021-44228 (Log4Shell)	Apache	Log4j2	RCE	CWE-917 Improper Neutralization of Special Elements used in an Expression Language Statement ('Expression Language Injection') CWE-20 Improper Input Validation CWE-400 Uncontrolled Resource Consumption CWE-502 Deserialization of Untrusted Data

CISA CyberSecurity Advisories list CWEs and Type (Impact) - and they're loosely related.

https://www.cisa.gov/news-events/cybersecurity-advisories/aa23-215a

Existing Definition and Usage of Impacts

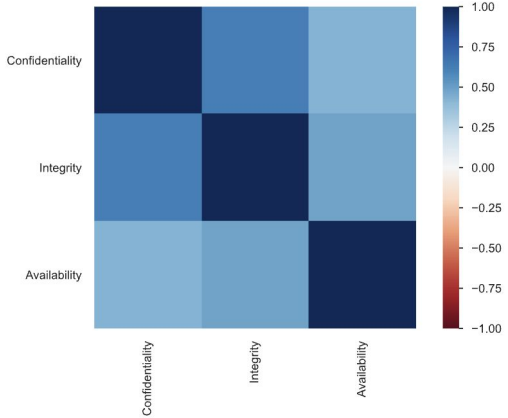
CVSS Impact

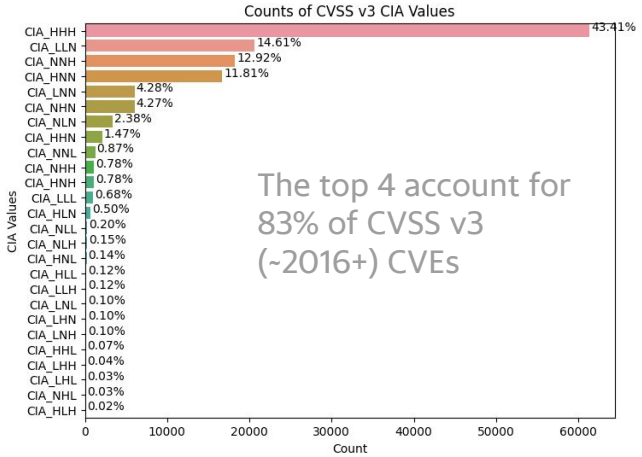
"3.4 Confidentiality and Integrity, Versus Availability Impacts

- ***The Confidentiality and Integrity metrics refer to impacts that affect the data used by the service.*** *For example, web content that has been maliciously altered, or system files that have been stolen.*
- ***The Availability impact metric refers to the operation of the service****. That is, the Availability metric speaks to the performance and operation of the service itself – not the availability of the data."*

Impact Values: High, Low, None

CIA (CVSS v3) Impact: High Correlation, Low Information



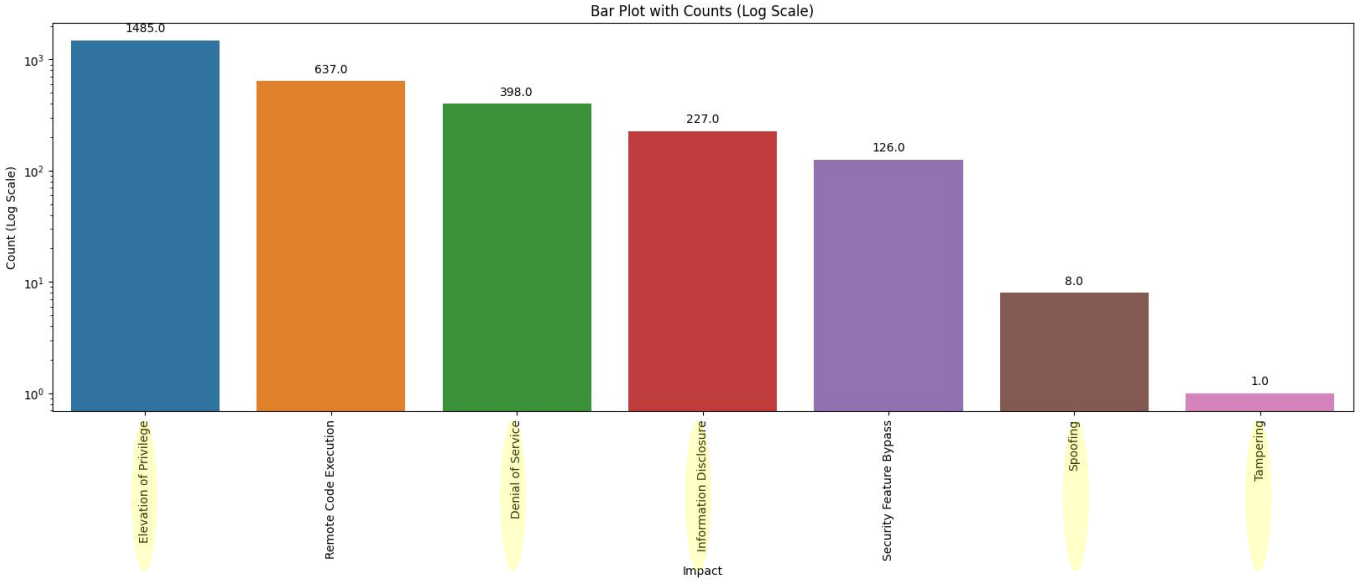


CVSS Impact lacks granularity to know what Impact to focus on e.g. C High ~60% of CVEs

https://www.first.org/cvss/v3.1/user-guide#3-2-Confidentiality-and-Integrity-Versus-Availability-Impacts https://www.first.org/cvss/calculator/3.1

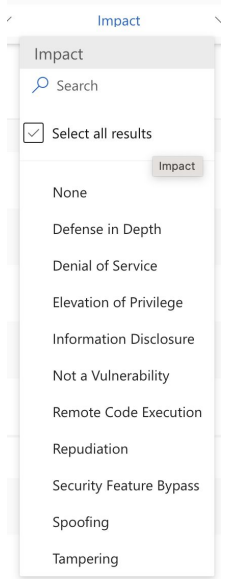
Microsoft Exploitability Index

"The Exploitability Index may help customers evaluate risk for a vulnerability. Microsoft evaluates the potential exploitability of each vulnerability associated with a Microsoft security update and then publishes the exploitability information as part of the monthly Microsoft security update details"



STRIDE Threat
Spoofing
Tampering
Repudiation
Information disclosure
Denial of service
Elevation of privilege

STRIDE



https://en.wikipedia.org/wiki/STRIDE_(security)

MSEI assigns Impacts to vulnerabilities (and some of these map to STRIDE)

https://ostering.com/blog/2022/03/07/capec-stride-mapping/ https://riskbasedprioritization.github.io/vendors/Microsoft_Exploitability_Index/ https://msrc.microsoft.com/update-guide

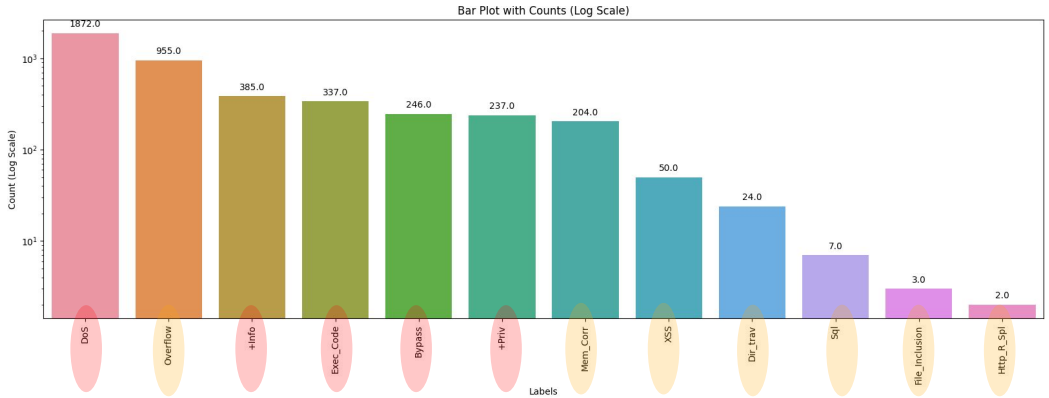
Cvedetails.com

CVEdetails.com assigns types/categories to vulnerabilities using CWE IDs and keywords.

Year	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	File Inclusion	CSRF	XXE	SSRF	Open Redirect	Input Validation
2013	844	577	153	645	111		126	38	3	31	499
2014	832	627	304	1099	207	3	266	67	10	48	536
2020	1222	1904	466	2199	441	110	416	119	132	101	831
2021	1675	2564	744	2726	557	93	520	126	197	133	702
2022	1881	3420	1790	3407	731	102	769	127	234	149	826
2023	1588	2502	1868	4450	713	117	1216	128	204	143	703
Total	16136	19230	7206	21731	4358	837	5023	1133	1082	988	8202

Vulnerabilities by impact types

Year	Code Execution	Bypass	Privilege Escalation	Denial of Service	Information Leak
2013	879	111	114	1454	251
2014	1041	165	186	1597	356
2020	1661	816	1384	1675	1095
2021	2084	805	1120	2298	926
2022	2065	950	1536	2438	1142
2023	2393	919	1374	2344	1376
Total	17924	6606	9371	22927	10591



Type/Category

Technical Impact

Sample is https://github.com/ZeoVan/MSR_20_Code_vulnerability_CSV_Dataset of ~4K CVEs with Vulnerability Type assigned (scraped from CVEDetails.com)

CVEs labeled with 16 labels: 11 types/categories + 5 Impact Types

https://www.cvedetails.com/vulnerabilities-by-types.php

NIST Vulntology (Vulnerability Data Ontology)

Impact Method Type

A description of the method used to exploit a vulnerability providing some additional information on the impact of exploitation. These are intended to be high level concepts that lead to more granular impacts as referenced in the logical and physical impact entities.

Values

Items that are indented represent more specific values that can be used to describe the parent value. For instance, choosing `Failure to Verify Content` as a value would imply `Trust Failure` as well.

- **Authentication Bypass**: Exploitation of the Vulnerability takes advantage of a failure to identify the adversary properly, directly leading to additional access or permissions.
- **Code Execution**: Exploitation of the Vulnerability allows an adversary to execute unauthorized code, causing an impact to a Context.
- **Context Escape**: The Vulnerability allows an adversary to exploit a trust mechanism by breaking out of a sandbox and into another workspace. This SHOULD be associated with the Context that has been escaped.
- **Trust Failure**: Exploitation of the Vulnerability takes advantage of an assumed trust relationship leading to unexpected impacts. Examples include failures of inherent trust, failure to verify a communicator, or the content being transmitted.
  - **Failure to Establish Trust**: The Context failed to verify the input originated from a trusted source, in other words a check is missing or non-existent.
  - **Failure to Verify Content**: The Context failed to ensure the content supplied is the expected content, is properly formatted and/or sanitized.
  - **Failure to Verify Receiver**: The Context failed to ensure the entity on the receiving end of the communication is the intended entity.
  - **Failure to Verify Transmitter**: The Context failed to ensure the entity on the transmitting end of the communication is the intended entity.
- **Privilege Escalation**: The Vulnerability allows an adversary to gain a level of privilege that was not intended.

Logical Impact

A description of the possible impacts to a Context that a successful exploitation of the Vulnerability will have. The same Vulnerability can have multiple and different Logical Impact values across different Context or Scenario instances.

Values

Items that are indented represent more specific values that can be used to describe the parent value. For instance, choosing `Hang` as a value would imply `Service Interrupt` as well.

- **Indirect Disclosure**: The Vulnerability allows an adversary to learn information about the Context, but the knowledge gained is not from a direct read operation. Examples include but are not limited to discovering memory locations protected by address space layout randomization (ASLR), information from side-channel attacks, or information gained from traffic analysis.
- **Read Direct**: The Vulnerability allows an adversary to cause a breach of confidentiality by gaining unauthorized access to data in the Context.
- **Resource Removal (Data)**: The Vulnerability allows an adversary to perform an unauthorized removal (deletion) of data from a resource in the Context.
- **Service Interrupt**: The Vulnerability allows an adversary to cause an unauthorized loss of availability by temporarily or permanently disabling all or a portion of the Context.
  - **Hang**: The service interruption results in the Context being stuck at a certain point and unable to continue function.
  - **Panic**: The service interruption results in the Context crashing.
  - **Reboot**: The service interruption results in the Context being disabled, but starting back up immediately.
  - **Shutdown**: The service interruption results in the Context being disabled, without starting back up immediately.
  - **Unrecoverable**: The service interruption results in a complete and unrecoverable loss of the Context but is non-physical in nature. For example, the corruption of the firmware on a hardware device with no possibility of reload.
- **Write Direct**: The Vulnerability allows an adversary to cause a breach in the integrity of the Context through unauthorized modification or addition of data.

My Feedback to this: https://github.com/usnistgov/vulntology/issues/159

Impact (end result) and Impact Method (means to the end) are split out

https://pages.nist.gov/vulntology/specification/values/impact-method-type/ https://pages.nist.gov/vulntology/specification/values/logical-impact/

Summarizing Vulnerabilities Paper

"Summarizing vulnerabilities' descriptions to support experts during vulnerability assessment activities" defined 10 types of vulnerability:

1. Authentication bypass or Improper Authorization
2. Cross-site scripting or HTML injection
3. Denial of service
4. Directory Traversal
5. Local/Remote file include and Arbitrary file upload
6. Information disclosure and/or Arbitrary file read
7. Buffer/stack/heap/integer overflow
8. Remote code execution
9. SQL injection
10. Unspecified vulnerability

```
AuthenticationBypass,
CrossSiteScripting,
DenialOfService,
DirectoryTraversal,
FileInclude,
InformationDisclosure,
Overflow,
RemoteCodeExecution,
SQLinjection,
UnspecifiedVulnerability}
```

https://github.com/jssrp2018/CVErizer/blob/master/CVErizer_replication_package/RQ1/train.arff

These 10 Types are a mix of Weakness and Technical Impact

Summarizing vulnerabilities' descriptions to support experts during vulnerability assessment activities.
Ernesto R Russo, Andrea D Sorbo, Corrado A Visaggio, and Gerardo Canfora. 2019. Journal of Systems and Software 156 (2019), 84–99.

Existing Definition and Usage of Impacts:

MITRE

MITRE on Technical Impact

"Software developers often face hundreds or thousands of individual bug reports for weaknesses that are discovered in their code. *In certain circumstances, a software weakness can even lead to an exploitable vulnerability.* ***Due to this high volume of reported weaknesses, stakeholders are often forced to prioritize which issues they should investigate and fix first, often using incomplete information. In short, people need to be able to reason and communicate about the relative importance of different weaknesses."***

Technical Impacts per MITRE

"While there are a large number of weaknesses in CWE, there appear to be ***only eight different consequences or technical impacts*** *to which these failures lead (see the table below). In other words, if a weakness manifests itself in a product in an exploitable manner and an attacker successfully exploits it, then there will be one of eight technical impacts or consequences from that weakness."*

1. Read data — Confidentiality
2. Modify data — Integrity
3. Denial-of-Service: unreliable execution — Availability
4. Denial-of-Service: resource consumption — Availability
5. Execute unauthorized code or commands
6. Gain privileges / assume identity
7. Bypass protection mechanism
8. Hide activities

Impact

Impact Method
(in NIST Vulntology terms)

What should my organization focus on?

"The collapsing of the hundreds of types of errors into a small set of technical impacts offers a simplification to the question, "What should my organization focus on to gain assurance in our software?".

Instead of trying to remove all weaknesses, you can decide which of the eight impacts are either more or less dangerous to you, given what the software product is doing for your organization."

Using the small number of Technical Impacts allows us to focus on what to fix first

https://github.com/usnistgov/vulntology https://cwe.mitre.org/community/swa/priority.html https://cwe.mitre.org/cwss/cwss_v1.0.1.html

MITRE CWE Technical Impact Examples

“Within each CWE entry the "common consequences" field lists the "technical impacts" that can result from each weakness in CWE. **The technical impact and its translation into an *impact to the mission* are important criteria that can be useful to any organization** needing reasonable assurance that their software-based capabilities do what is intended and nothing more.”

CWE-917

Common Consequences

Scope	Impact	Likelihood
Confidentiality	**Technical Impact:** *Read Application Data*	
Integrity	**Technical Impact:** *Execute Unauthorized Code or Commands*	

Observed Examples

Reference	Description
CVE-2021-44228	Product does not neutralize ${xyz} style expressions, allowing remote code execution. (log4shell vulnerability in log4j)

https://cwe.mitre.org/data/definitions/917.html
CWE-917: Improper Neutralization of Special Elements used in an Expression Language Statement ('Expression Language Injection')

CWE-78

Common Consequences

Scope	Impact	Likelihood
Confidentiality Integrity Availability Non-Repudiation	**Technical Impact:** *Execute Unauthorized Code or Commands; DoS: Crash, Exit, or Restart; Read Files or Directories; Modify Files or Directories; Read Application Data; Modify Application Data; Hide Activities* Attackers could execute unauthorized commands, which could then be used to disable the product, or read and modify data for which the attacker does not have permissions to access directly. Since the targeted application is directly executing the commands instead of the attacker, any malicious activities may appear to come from the application or the application's owner.	

https://cwe.mitre.org/data/definitions/78.html
CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')

CWEs have optional "Technical Impact" fields. Likelihood is generally not populated.

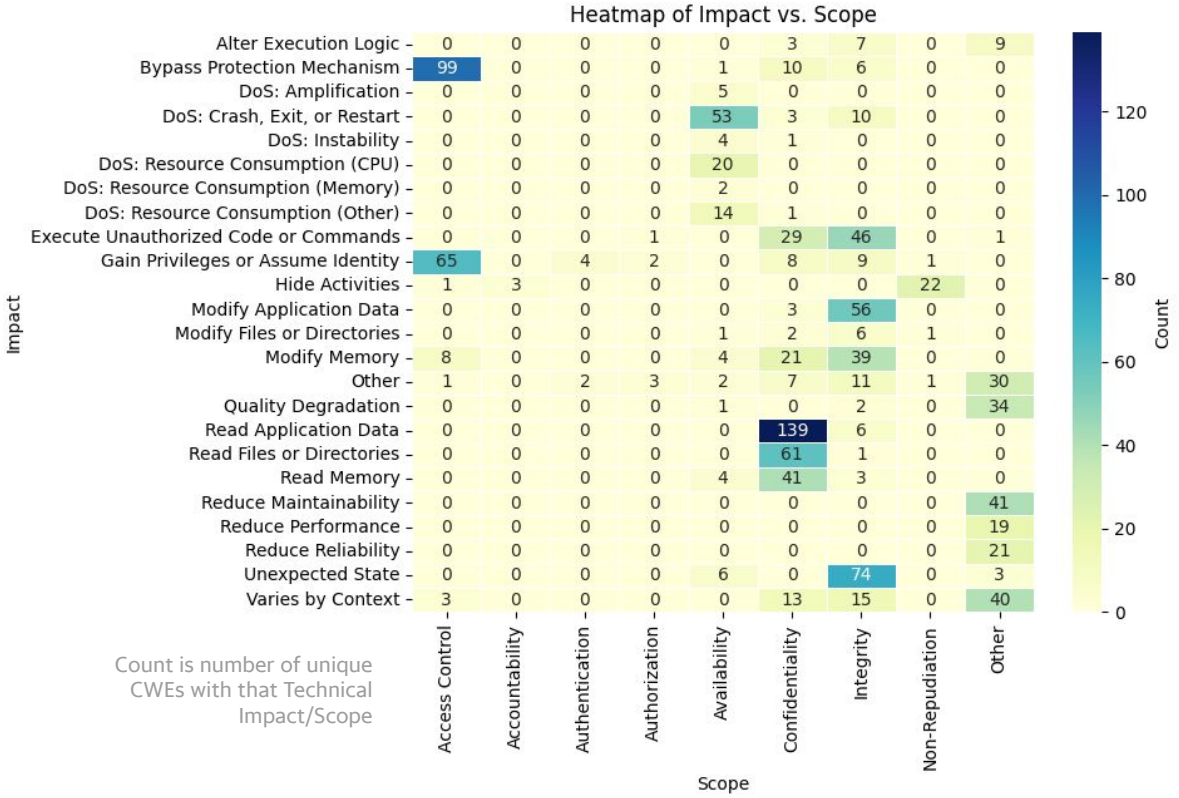
64 entries for Likelihood - of which 46 are High.

MITRE CWE Technical Impact vs Scope - MITRE CAPEC

CAPEC

Custom Filter

Field	Value(s)
Likelihood	Please select one or more values to view attack patterns that may have the given likelihood. High · Medium · Low · Unknown
Scope	Please select one or more values to view attack patterns that contain the desired scope. Confidentiality · Integrity · Availability · Access Control · Accountability · Authentication · Authorization · Non-Repudiation · Other
Severity	Please select one or more values to view attack patterns that reflect the desired severity. Very High · High · Medium · Low · Very Low
Technical Impact	Please select one or more values to view attack patterns that could exhibit the following technical impacts. Modify Data · Read Data · Unreliable Execution · Resource Consumption · Execute Unauthorized Commands · Gain Privileges · Bypass Protection Mechanism · Hide Activities · Alter Execution Logic · Other



Count is number of unique CWEs with that Technical Impact/Scope

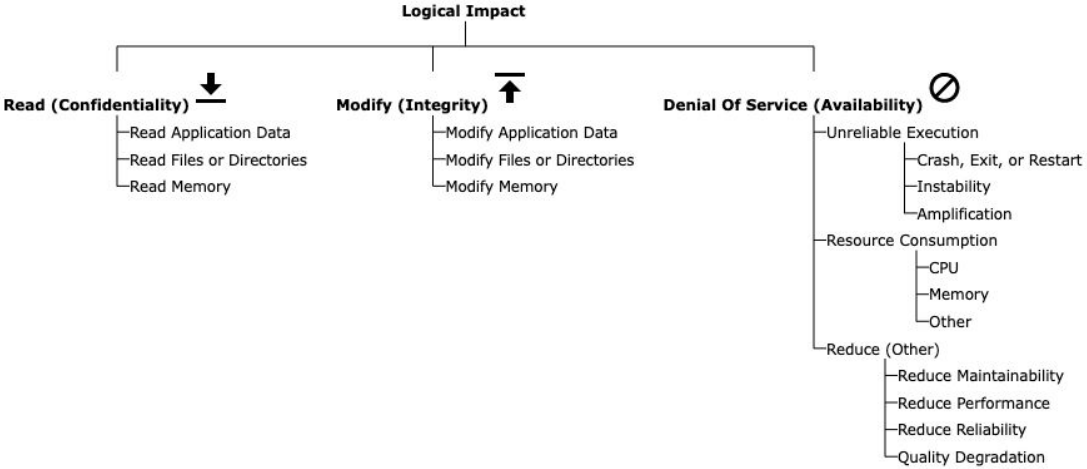
MITRE CWE is consistent with MITRE CAPEC.

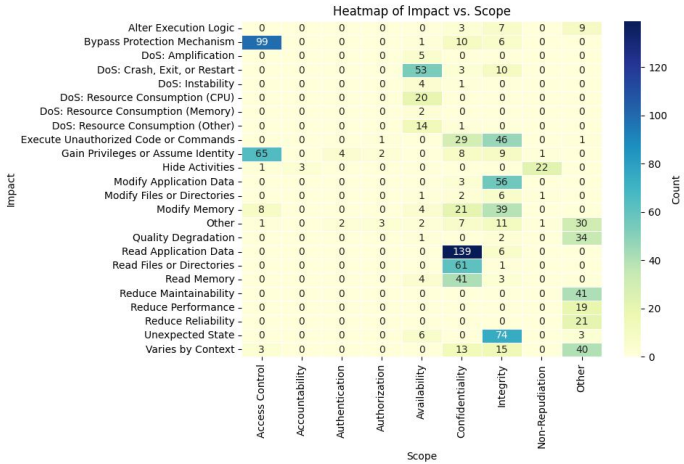
CAPEC: Common Attack Pattern Enumerations and Classifications

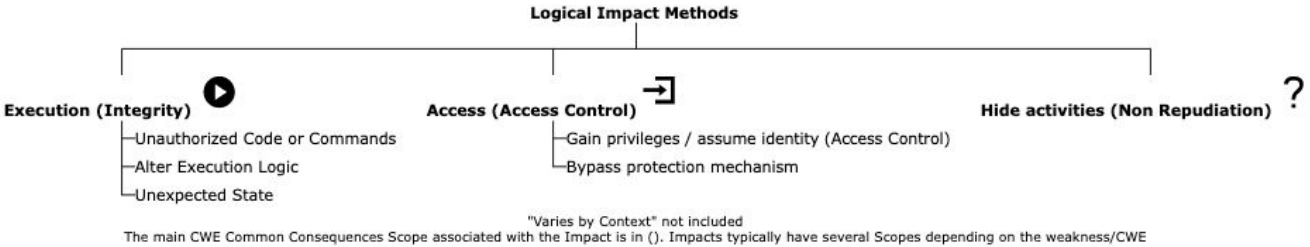
https://capec.mitre.org/custom/view.html?id=1000

Impact Taxonomy

Impact and Impact Methods Taxonomy





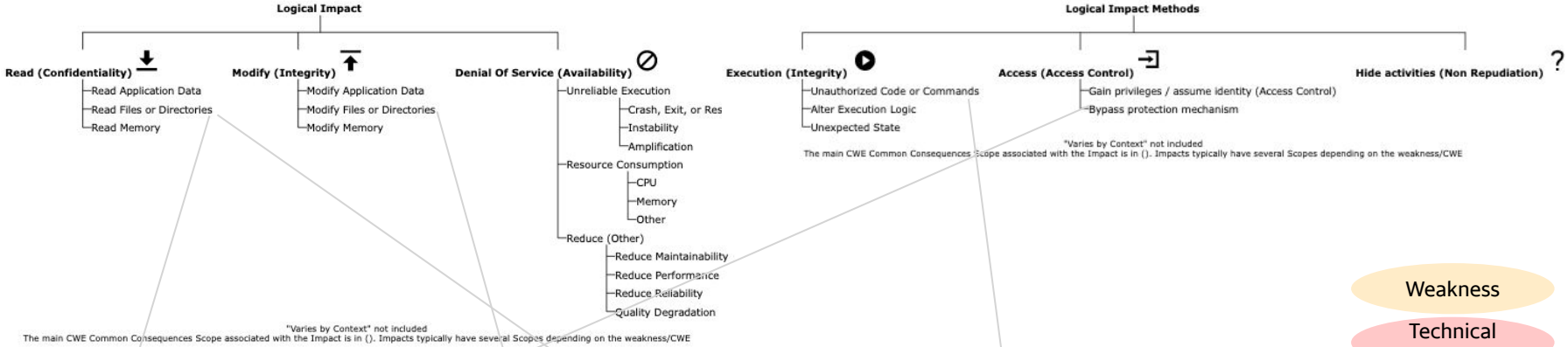


"Varies by Context" not included
The main CWE Common Consequences Scope associated with the Impact is in (). Impacts typically have several Scopes depending on the weakness/CWE

Organizing by Impact and Primary Scope gives a Taxonomy for Impact and Method

https://github.com/theparanoids/PrioritizedRiskRemediation

Example CVE-2009-1936



Weakness

Technical Impact

Description

_functions.php in cpCommerce 1.2.x, possibly including 1.2.9, sends a redirect but does not exit when it is called directly, which allows remote attackers to bypass a protection mechanism to conduct remote file inclusion and directory traversal attacks, execute arbitrary PHP code, or read arbitrary files via the GLOBALS[prefix] parameter, a different vector than CVE-2003-1500.

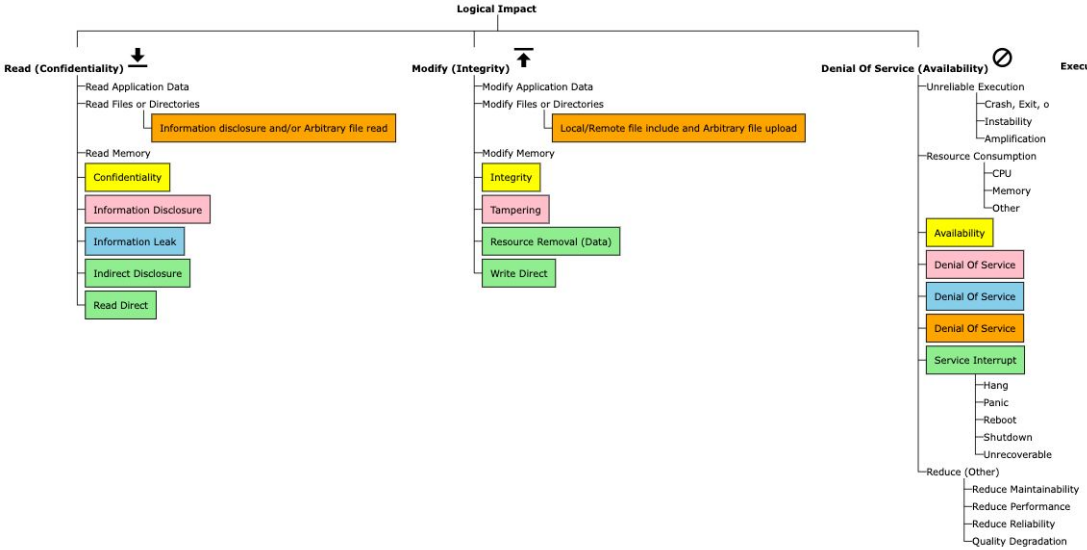
"Weakness is the root mistake, which can lead to a vulnerability (tracked by CVE CVE-2009-1936), which can be exploited by an attacker (using techniques covered by CAPEC) ", which can lead to **a Technical Impact (or consequence),** which can result in a Business Impact

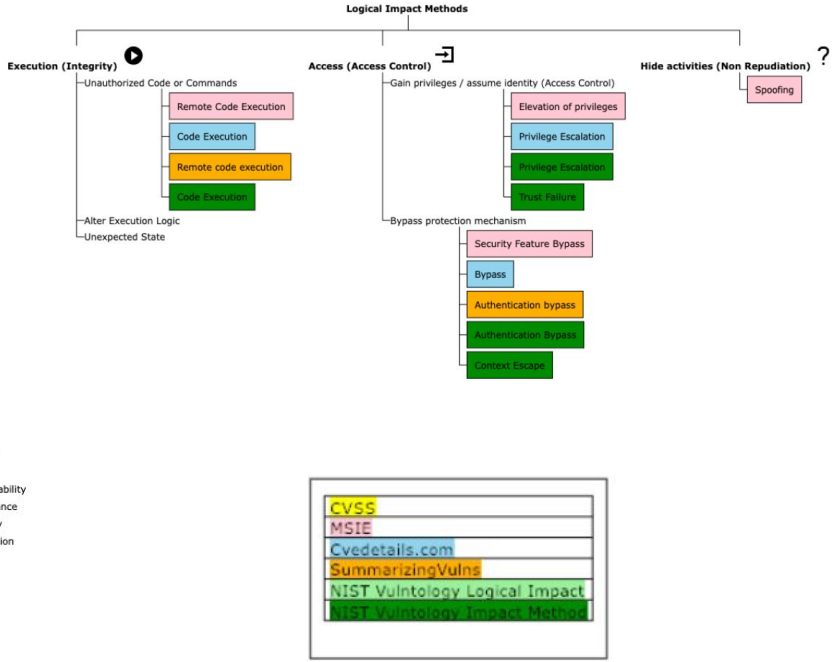
We care more about the Impact than the Impact Method

https://nvd.nist.gov/vuln/detail/CVE-2009-1936

https://capec.mitre.org/data/definitions/126.html "CAPEC-126: Path Traversal"

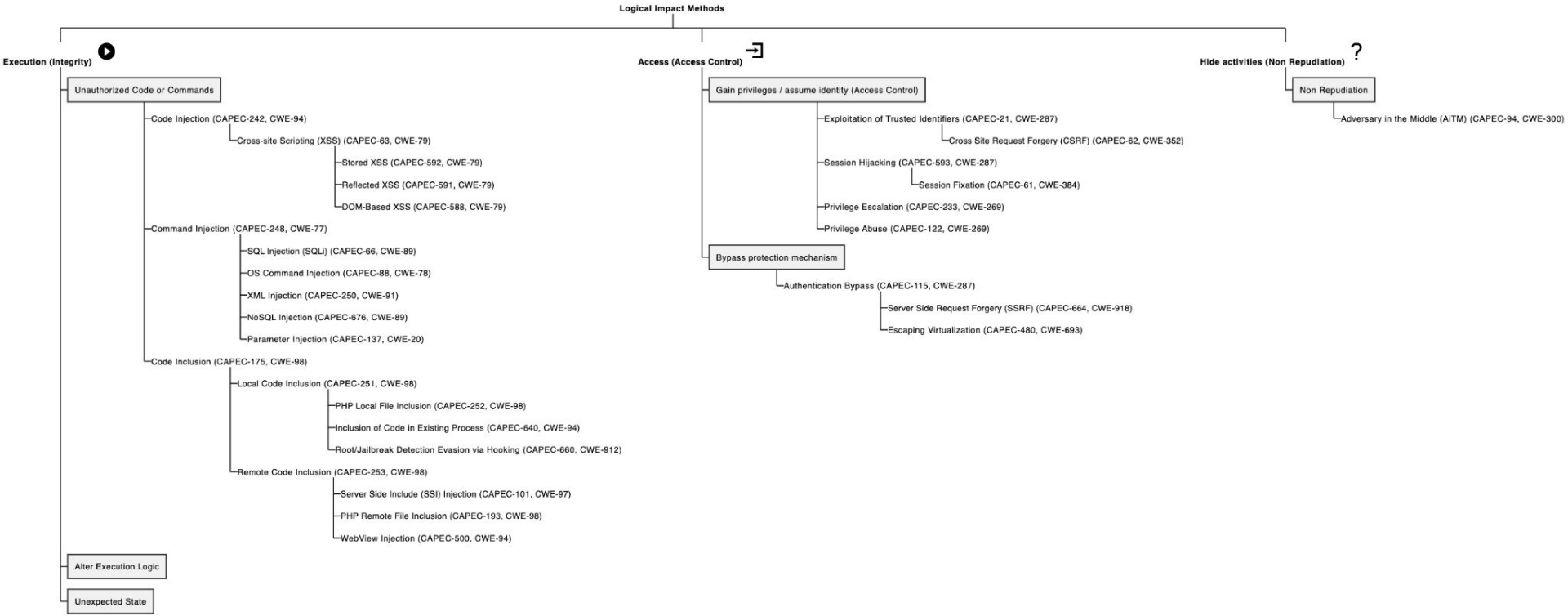
Logical Impact and Impact Methods Taxonomy





The other Standards fit with this Impact and Impact Methods Taxonomy

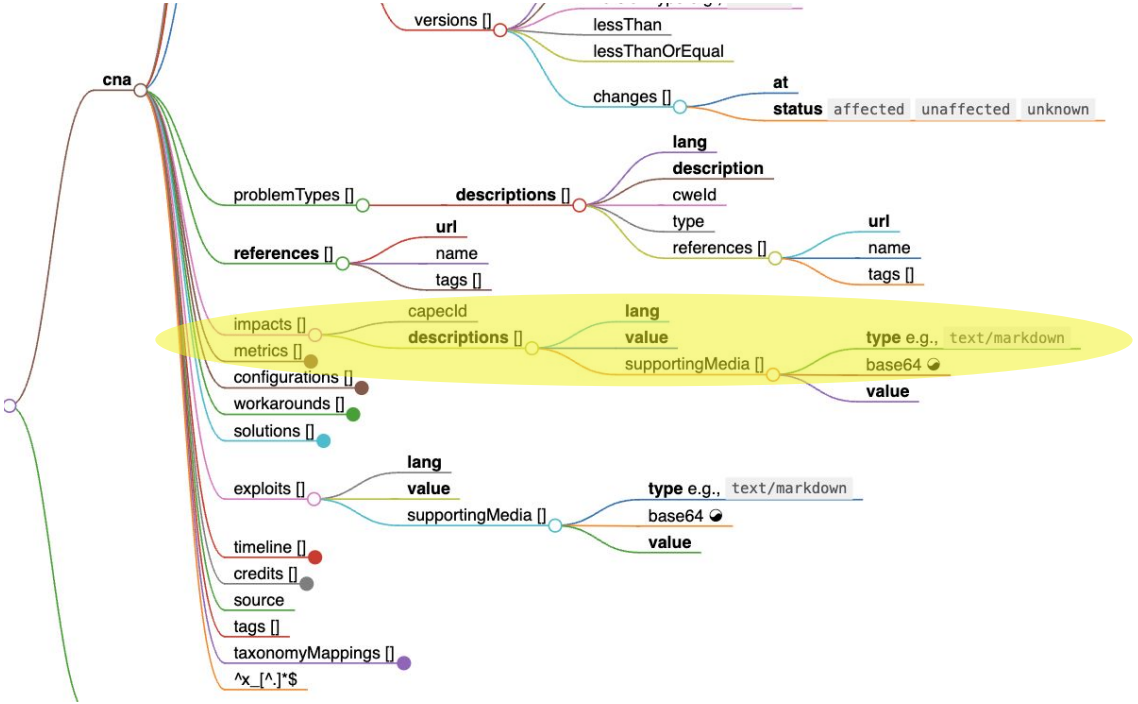
CWE, CAPEC - Impact Methods Taxonomy



CWE (Weakness) and CAPEC (Attack Pattern) can be mapped to the Impact Taxonomy

CVE: CWE Weakness & CAPEC Attack Pattern/Impact


```
{
  "cnaContainer": {
    "title": "Buffer overflow in Example Enterprise allows
Privilege Escalation.",
    "datePublic": "2021-09-08T16:24:00.000Z",
    "problemTypes": [
      {
        "descriptions": [
          {
            "lang": "en",
            "cweId": "CWE-78",
            "description": "CWE-78 OS Command Injection",
            "type": "CWE"
          }
        ]
      }
    ],
    "impacts": [
      {
        "capecId": "CAPEC-233",
        "descriptions": [
          {
            "lang": "en",
            "value": "CAPEC-233 Privilege Escalation"
          }
        ]
      }
    ],
    "affected": [
      {
        "vendor": "Example.org",
        "product": "Example Enterprise",
        "platforms": [
          "Windows",
          "MacOS",
          "XT-4500"
        ],
```

CapecID (Common Attack Pattern) is used for Impacts

https://raw.githubusercontent.com/CVEProject/cve-schema/master/schema/v5.0/docs/cnaContainer-advanced-example.json https://cveproject.github.io/cve-schema/schema/v5.0/docs/mindmap.html

Meanwhile, back in reality...

What we Want

Given a text description of a vulnerability (e.g. a CVE description), automatically extract and classify the vulnerability impact (using industry standard impacts) if Impact is described, with the weaknesses/consequence(s) described.

Description

_functions.php in cpCommerce 1.2.x, possibly including 1.2.9, sends a redirect but does not exit when it is called directly, which allows remote attackers to bypass a protection mechanism to conduct remote file inclusion and directory traversal attacks, execute arbitrary PHP code, or read arbitrary files via the GLOBALS[prefix] parameter, a different vector than CVE-2003-1500.

Weakness

Technical Impact

Problem Type:

1. Multi-label Classification problem: a CVE may have one or more labels (Impacts)
2. Class-Imbalance: Some CVE Impacts appear a lot more than others e.g. Code Execution, Denial Of Service
3. Unstructured Text: CVE descriptions are free text with typos and variants
4. Domain-specific Language: CyberSecurity

Constraints:

1. No labelled dataset (to train a model)
2. Limited Human Time => minimise required supervision (background project to https://riskbasedprioritization.github.io/, which was a background project to my real job)
3. Limited GPU Budget €/$ 50 (my money)

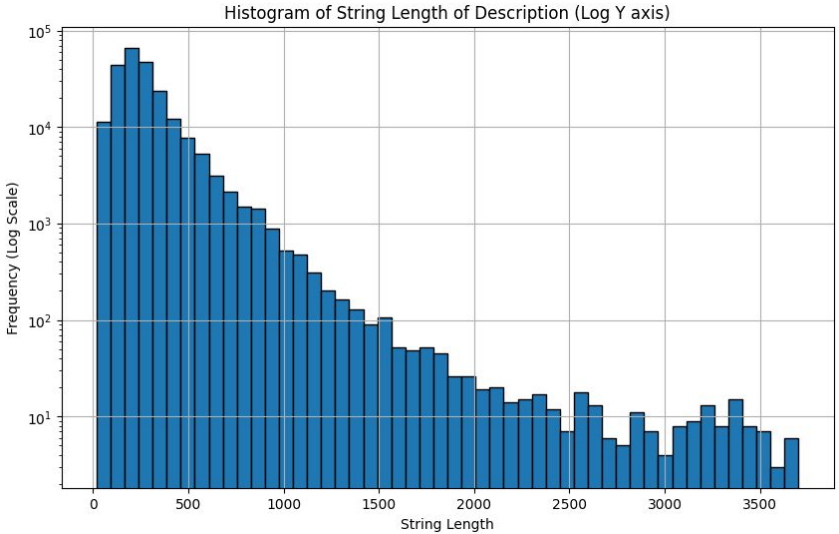
State what you want - not what you think you can have!

Example CVE-2009-1936

The Challenge

The data to analyze:

- ~230K Published CVEs Descriptions
- ~10M words
  - 428,899 unique words
- ~66M characters
- longest CVE Description is ~4K characters (truncated)



Description

In the Linux kernel, the following vulnerability has been resolved: ipmr: fix kernel panic when forwarding mcast packets The stacktrace was: [86.305548] BUG: kernel NULL pointer dereference, address: 0000000000000092 [86.306815] #PF: supervisor read access in kernel mode [86.307717] #PF: error_code(0x0000) - not-present page [86.308624] PGD 0 P4D 0 [86.309091] Oops: 0000 [#1] PREEMPT SMP NOPTI [86.309883] CPU: 2 PID: 3139 Comm: pimd Tainted: G U 6.8.0-6wind-knet #1 [86.311027] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.11.1-0-g0551a4be2c-prebuilt.qemu-project.org 04/01/2014 [86.312728] RIP: 0010:ip_mr_forward (/build/work/knet/net/ipv4/ipmr.c:1985) [86.313399] Code: f9 1f 0f 87 85 03 00 00 48 8d 04 5b 48 8d 04 83 49 8d 44 c5 00 48 8b 40 70 48 39 c2 0f 84 d9 00 00 00 49 8b 46 58 48 83 e0 fe <80> b8 92 00 00 00 00 0f 84 55 ff ff ff 49 83 47 38 01 45 85 e4 0f [86.316565] RSP: 0018:ffffad21c0583ae0 EFLAGS: 00010246 [86.317497] RAX: 0000000000000000 RBX: 0000000000000000 RCX: 0000000000000000 [86.318596] RDX: ffff9559cb46c000 RSI: 0000000000000000 RDI: 0000000000000000 [86.319627] RBP: ffffad21c0583b30 R08: 0000000000000000 R09: 0000000000000000 [86.320650] R10: 0000000000000000 R11: 0000000000000000 R12: 0000000000000001 [86.321672] R13: ffff9559c093a000 R14: ffff9559cc00b800 R15: ffff9559c09c1d80 [86.322873] FS: 00007f85db661980(0000) GS:ffff955a79d00000(0000) knlGS:0000000000000000 [86.324291] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [86.325314] CR2: 0000000000000092 CR3: 00000002f13a000 CR4: 0000000000350ef0 [86.326589] Call Trace: [86.327036] <TASK> [86.327434] ? show_regs (/build/work/knet/arch/x86/kernel/dumpstack.c:479) [86.328049] ? __die (/build/work/knet/arch/x86/kernel/dumpstack.c:421 /build/work/knet/arch/x86/kernel/dumpstack.c:434) [86.328508] ? page_fault_oops (/build/work/knet/arch/x86/mm/fault.c:707) [86.329107] ? do_user_addr_fault (/build/work/knet/arch/x86/mm/fault.c:1264) [86.329756] ? srso_return_thunk (/build/work/knet/arch/x86/lib/retpoline.S:223) [86.330350] ? __irq_work_queue_local (/build/work/knet/kernel/irq_work.c:111 (discriminator 1)) [86.331013] ? exc_page_fault (/build/work/knet/./arch/x86/include/asm/paravirt.h:693 /build/work/knet/arch/x86/mm/fault.c:1515 /build/work/knet/arch/x86/mm/fault.c:1563) [86.331702] ? asm_exc_page_fault (/build/work/knet/./arch/x86/include/asm/idtentry.h:570) [86.332468] ? ip_mr_forward (/build/work/knet/net/ipv4/ipmr.c:1985) [86.333183] ? srso_return_thunk (/build/work/knet/arch/x86/lib/retpoline.S:223) [86.333920] ipmr_mfc_add (/build/work/knet/./include/linux/rcupdate.h:782 /build/work/knet/net/ipv4/ipmr.c:1009 /build/work/knet/net/ipv4/ipmr.c:1273) [86.334583] ? __pfx_ipmr_hash_cmp (/build/work/knet/net/ipv4/ipmr.c:363) [86.335357] ip_mroute_setsockopt (/build/work/knet/net/ipv4/ipmr.c:1470) [86.336135] ? srso_return_thunk (/build/work/knet/arch/x86/lib/retpoline.S:223) [86.336854] ? ip_mroute_setsockopt (/build/work/knet/net/ipv4/ipmr.c:1470) [86.337679] do_ip_setsockopt (/build/work/knet/net/ipv4/ip_sockglue.c:944) [86.338408] ? __pfx_unix_stream_read_actor (/build/work/knet/net/unix/af_unix.c:2862) [86.339232] ? srso_return_thunk (/build/work/knet/arch/x86/lib/retpoline.S:223) [86.339809] ? aa_sk_perm (/build/work/knet/security/apparmor/include/cred.h:153 /build/work/knet/security/apparmor/net.c:181) [86.340342] ip_setsockopt (/build/work/knet/net/ipv4/ip_sockglue.c:1415) [86.340859] raw_setsockopt (/build/work/knet/net/ipv4/raw.c:836) [86.341408] ? security_socket_setsockopt (/build/work/knet/security/security.c:4561 (discriminator 13)) [86.342116] sock_common_setsockopt (/build/work/knet/net/core/sock.c:3716) [86.342747] do_sock_setsockopt (/build/work/knet/net/socket.c:2313) [86.343363] __sys_setsockopt (/build/work/knet/./include/linux/file.h:32 /build/work/kn ---truncated---

The longest CVE descriptions are from the Linux Kernel, who became a CNA (CVE Numbering Authority) Feb 2024
https://www.cve.org/Media/News/item/news/2024/02/13/kernel-org-Added-as-CNA

The shortest CVE descriptions are “NFS cache poisoning.” “Remote code execution”, “Elevation of privilege”

Start by learning ML/AI... and finish with a conference presentation on the solution 😁

https://nvd.nist.gov/vuln/detail/CVE-2024-26626 https://nvd.nist.gov/vuln/detail/CVE-1999-0165 https://nvd.nist.gov/vuln/detail/CVE-2022-42541

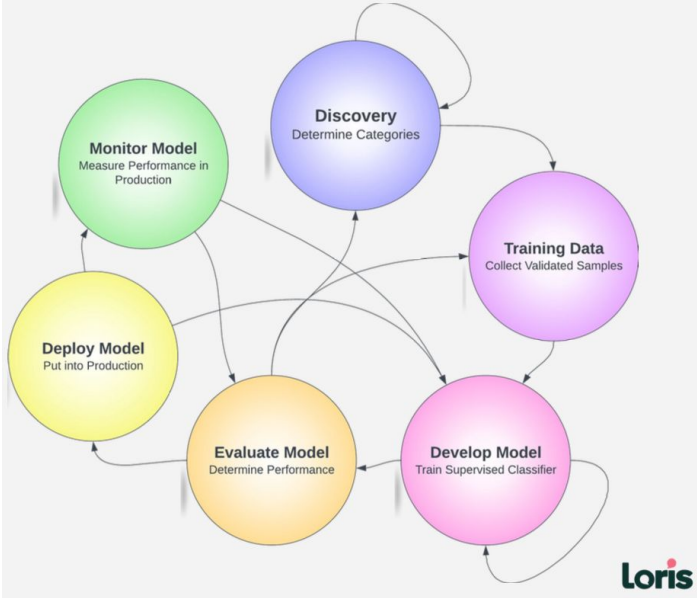
Approach

Understand

- **the problem**
- **the data**
- **the value**

Small Verifiable Steps from a Solid Base

- **Bootstrap**: labels/phrases with CISA KEV Topic Modelling, a model with SetFit that requires a small set of data
- **Iterate**: Start and learn with a small sample: CISA KEV (~1K), a subset of Impacts, and add more as you learn
- **Thin vertical slices** (across the data, classes e.g. that give best coverage, model)



Automate as much as possible: use multiple independent automated methods

Use human input for

- Manual review of Labels, Embeddings / Clusterings to highlight the problem areas
- Arbitration of the disagreements
- Bootstrapping AI with small data to then assist with larger data

The Approach is generic to any data project

https://www.youtube.com/watch?v=_0LyvLoKt5Q&t=1682s

Does Reality Fit with our Taxonomy?

We’re gonna need some help….

Spacy Custom NER

Impact Text for PhraseMatching, Regex, Fuzzy

```
top_group,group,cwe,acronym,phrase
Read Application Data,Information Disclosure,"['CWE-200']",,"['information disclosure', 'informa
Read Application Data,Weak Cryptography,"['CWE-326', 'CWE-327']",,"['weak cryptography', 'weak e
Read Application Data,Server Side Request Forgery,"['CWE-918']",['ssrf'],"['server side request
Read Application Data,Confidentiality Integrity Availability,,,"['confidentiality, integrity and
Read Files or Directories,Arbitrary File Read,,,"['file read', 'arbitrary file read', 'arbitrary
Read Files or Directories,Directory Traversal,"['CWE-22']",,"['path traversal', 'directory trave
Read Memory,Out Of Bounds Read,"['CWE-125']",,['out of bounds read']
Modify Application Data,File Inclusion,"['CWE-98']","['lfi', 'rfi']","['uncontrolled search path
Modify Files or Directories,Arbitrary File Write,"['CWE-434']",,"['file write', 'to write files'
Modify Memory,Out Of Bounds Write,"['CWE-787']",,"['buffer overflow', 'stack based buffer overfl
Modify Memory,Memory Corruption,"['CWE-119','CWE-416']",,"['uninitialized memory', 'use after fr
Modify Memory,Type Confusion,"['CWE-843']",,"['type confusion']"
Modify Memory,Integer Overflow,"['CWE-190']",,"['integer overflow']"
Denial Of Service Unreliable Execution,Unreliable Execution,,,"['crash system', 'application cra
Denial Of Service Resource Consumption,Resource Consumption,"['CWE-400']",,"['memory exhaustion'
Denial Of Service,Denial Of Service,,['dos'],"['denial of service', 'denial-of-service', 'dos at
Execute Unauthorized Code or Commands,Code Execution,"['CWE-78','CWE-917']",['rce'],"['code exec
Execute Unauthorized Code or Commands,Code Injection,"['CWE-94']","['ssti','csti']","['code inje
Execute Unauthorized Code or Commands,XSS,"['CWE-79']",['xss'],"['cross-site scripting', 'cross
Execute Unauthorized Code or Commands,CSRF,"['CWE-352']",['csrf'],"['cross site request forgery'
Execute Unauthorized Code or Commands,XXE,"['CWE-611']",['xxe'],"['xml external entity']"
Execute Unauthorized Code or Commands,Command Execution,,,"['execute arbitrary commands', 'execu
Execute Unauthorized Code or Commands,Command Injection,"['CWE-77', 'CWE-78']",,"['command injec
Execute Unauthorized Code or Commands,Sql Injection,"['CWE-89']",['sqli'],['sql injection']
Bypass Protection Mechanism,Bypass Protection Mechanism,,,"['security bypass', 'bypass security'
Execute Unauthorized Code or Commands,Deserialization of Untrusted Data,"['CWE-502']",,['deseria
Gain Privileges or Assume Identity,Gain Privileges or Assume Identity,"['CWE-264', 'CWE-269']",,
Gain Privileges or Assume Identity,Weak Credentials,"['CWE-1391']",,"['cleartext storage of pass
Gain Privileges or Assume Identity,Bypass Authentication Authorization,"['CWE-284','CWE-285','CW
Hide Activities,Non Repudiation,"['CWE-290','CWE-295','CWE-300']",['mitm'],"['spoofing', 'impers
Unspecified,Unknown,,,"['unknown impact', 'unknown impacts']"
Unspecified,Unspecified,,,"['unspecified vulnerability', 'unspecified impact', 'unspecified impa
```

Spacy Custom NER: PhraseMatch

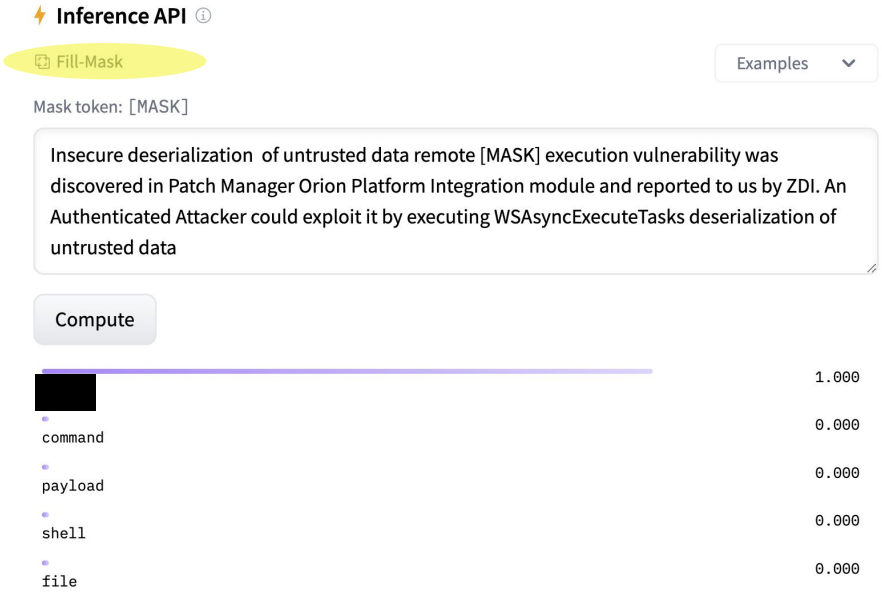
```
df = pd.read_csv("./data_out/impacts/impact_phrases_list.csv")


# Iterate through each row in the DataFrame and add phrases to the ruler
for _, row in df.iterrows():
    group = row['group']
    acronyms = row['phrase']
    top_group = row['top_group']

    # Create a pattern dictionary for the ruler
    patterns = [{"label": f"{top_group}", "id": f"{group}", "pattern": a} for a
in phrases]

    # Add the patterns to the ruler
    ruler.add_patterns(patterns)


# Apply ruler
df_text['Extracted_Entities'] = df_text['shortDescription'].apply(lambda text:
[(ent.label_, ent.ent_id_, ent.text) for ent in nlp(text).ents])
df_text['Extracted_Entities'] = df_text['Extracted_Entities'].apply(lambda x:
list(set(x))) #remove duplicates
```

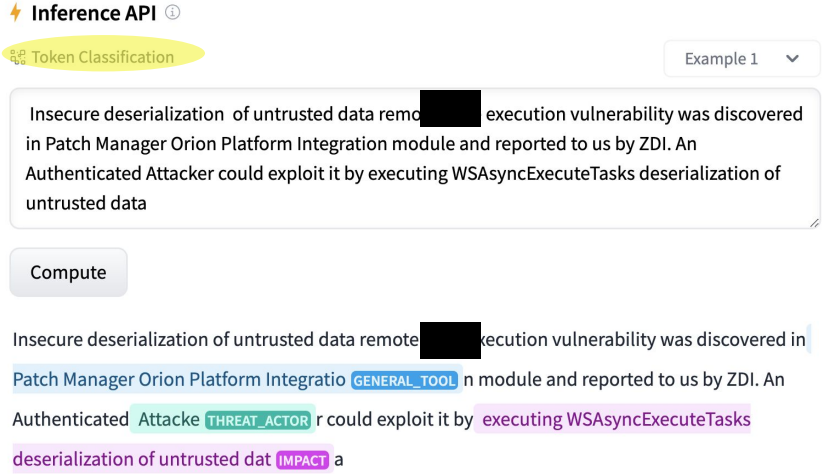
	shortDescription	Extracted_Group	Extracted_Entities
0	accellion fta contains an os command injection...	[Execute Unauthorized Code or Commands]	[(Execute Unauthorized Code or Commands, Comma...
1	accellion fta contains an os command injection...	[Execute Unauthorized Code or Commands]	[(Execute Unauthorized Code or Commands, Comma...
2	accellion fta contains a sql injection vulnera...	[Execute Unauthorized Code or Commands]	[(Execute Unauthorized Code or Commands, Sql I...
3	accellion fta contains a server side request f...	[Read Application Data]	[(Read Application Data, Server Side Request F...
4	acrobat acrobat and reader contain a heap base...	[Modify Memory, Execute Unauthorized Code or C...	[(Execute Unauthorized Code or Commands, Code ...

Spacy allows natural language processing (NLP) on large volumes of text at high speed

Let's call a friend... 🤗

We can invoke models (HuggingFace Inference API) via a browser.
On our first request, the model runtime spins up (~20s)



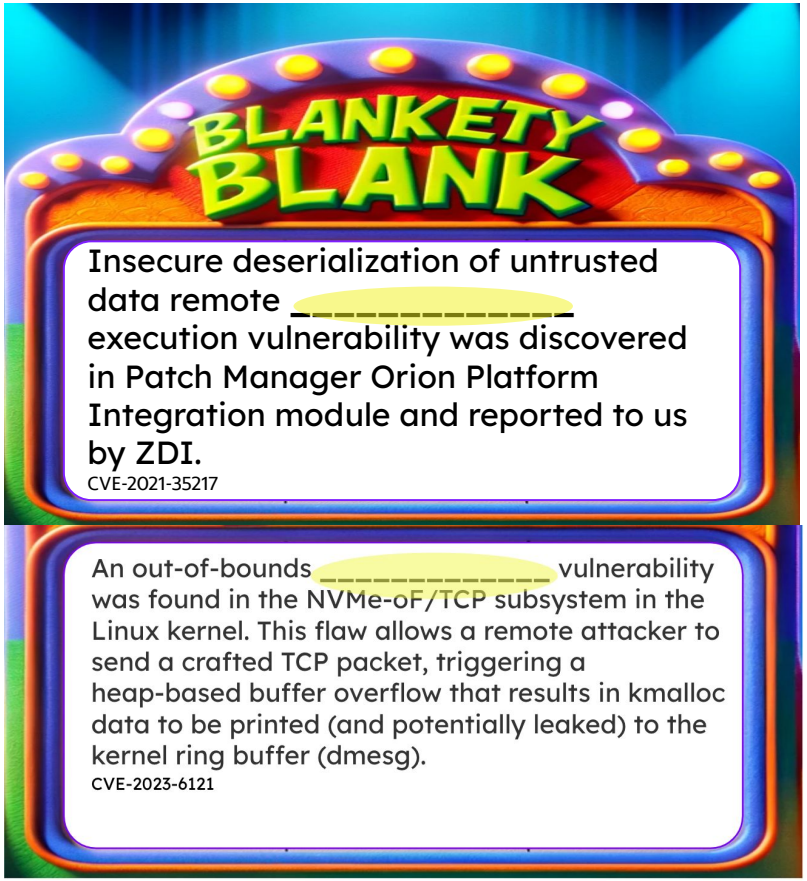


yahoo! 🤗 allows using models via the browser

https://huggingface.co/jackaduma/SecBERT

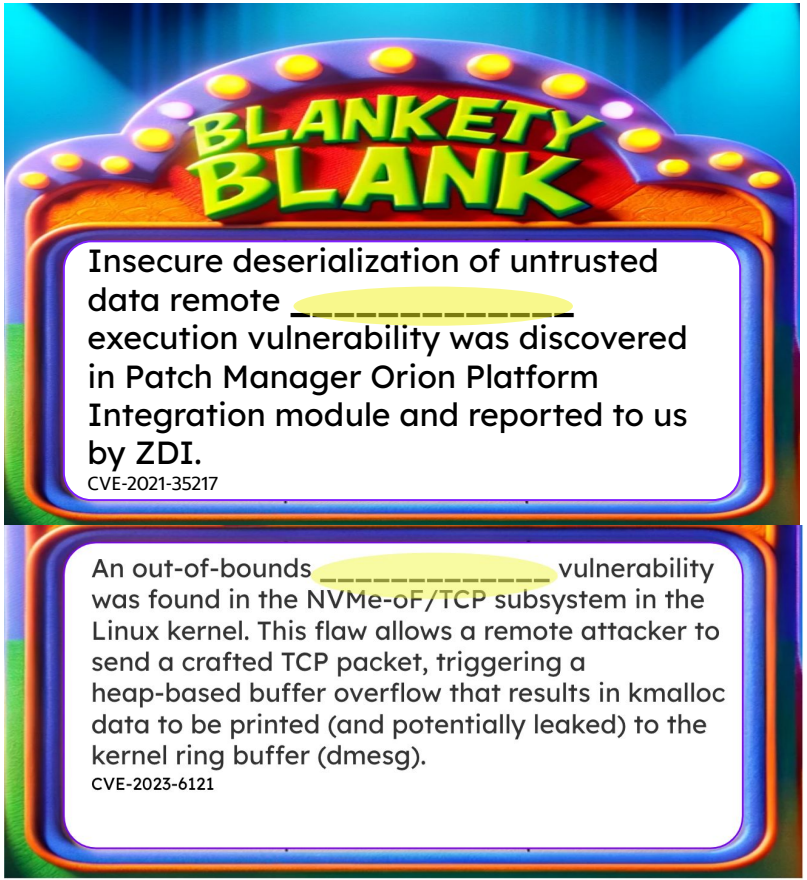
https://huggingface.co/anonymouspd/SecBERT-our-data

BERT asks YOU!



Bert asks a member of the audience…

BERT asks Friends



BERT base uncased
(110M params)
Training:
11K Books, Wikipedia

-	0.069
program	0.031
control	0.031
code	0.029
client	0.028

security	0.190
storage	0.074
kernel	0.046
cache	0.024
vulnerability	0.021

RoBERTa Large
(355M params)
Training:
11K Books, Wikipedia,
63M News articles,...
(160GB of text)

code	0.966
command	0.009
program	0.008
SQL	0.003
JavaScript	0.002

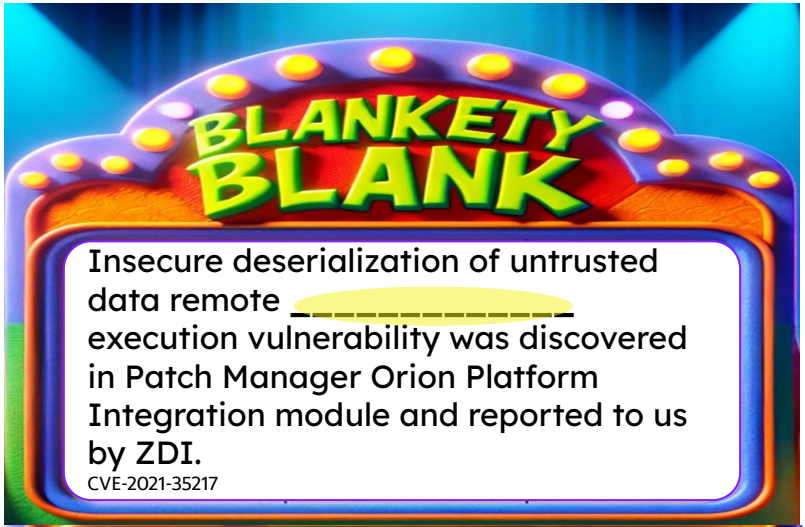
memory	0.196
execution	0.075
TCP	0.060
kernel	0.057
security	0.057

BERT Large
(336M params)
Training:
11K Books, Wikipedia

code	0.262
script	0.061
program	0.044
-	0.036
patch	0.031

security	0.563
software	0.047
payload	0.023
protocol	0.023
communication	0.021

BERT asks Security Friends



CySecBERT Training:
151k blogs, 16k arXiv papers, 171K NVD CVEs, 4M Tweets

code	0.976
command	0.023
commands	0.000
service	0.000
process	0.000

read	0.496
write	0.487
access	0.010
memory	0.003
reads	0.001

SecRoBERTa, SecBERT Training:
papers

code	0.973
command	0.027
file	0.000
script	0.000
Command	0.000

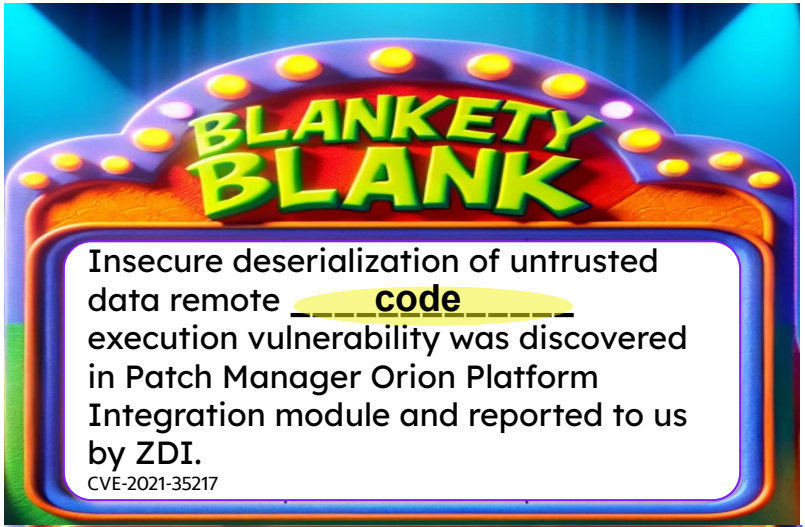
write	0.608
overflow	0.159
read	0.127
bypass	0.083
165	0.006

SecureBERT Training:
~98K files from many different cybersecurity resources

code	0.995
Code	0.002
command	0.002
file	0.000
script	0.000

read	0.674
write	0.256
access	0.061
reads	0.002
writes	0.001

BERT asks Security Friends



CySecBERT Training:
151k blogs, 16k arXiv papers, 171K NVD CVEs, 4M Tweets

code	0.976
command	0.023
commands	0.000
service	0.000
process	0.000
read	0.496
write	0.487
access	0.010
memory	0.003
reads	0.001

SecRoBERTa, SecBERT Training:
papers

code	0.973
command	0.027
file	0.000
script	0.000
Command	0.000
write	0.608
overflow	0.159
read	0.127
bypass	0.083
165	0.006

SecureBERT Training:
~98K files from many different cybersecurity resources

code	0.995
Code	0.002
command	0.002
file	0.000
script	0.000
read	0.674
write	0.256
access	0.061
reads	0.002
writes	0.001

Meet BERT's Friends

BERT Embeddings

ATT&CK BERT: a cybersecurity domain-specific language model based on sentence-transformers.

Topic Modeling:

BERTopic: topic modeling technique that leverages 🤗 transformers

BERT: Bidirectional Encoder Representations from Transformers

BootStrap Classifier

setFIT: Uses Contrastive Training to reduce needed labelled data. Few-Shot Learning (FSL) (Sep 2022)

Named Entity Recognition:

Impact_NER: the Domain-Adapted Language Model for the Cybersecurity Domain that we created.

LEVEL UP UNLOCK

SemanticSimilarity

sentence-transformers: cosine_similarity to compare similar phrases for Impact

Named Entity Recognition

GLiNER: Generalist and Lightweight model for Named Entity Recognition (Nov 2023)

Precision: how many retrieved items are relevant?

Recall: how many relevant items are retrieved?

LaGoNN

Optimized SetFIT: optimizes SetFit without changing the model.

Multi-label Classification

Impact_Class: the Domain-Adapted Language Model for the Cybersecurity Domain that we created

LEVEL UP UNLOCK

Why BERT Embeddings - and why ATT&CK BERT

BERT Embeddings

ATT&CK BERT: a cybersecurity domain-specific language model based on sentence-transformers. ATT&CK BERT maps sentences representing attack actions to a semantically meaningful embedding vector. Embedding vectors of sentences with similar meanings have a high cosine similarity.

An **Embedding** is a numerical representation of a piece of information e.g. text, The representation captures the semantic meaning of what is being embedded, making it robust for many industry applications.

BERT produces **word representations that are dynamically informed by the words around them**.
Aside from capturing obvious differences like polysemy, the **context-informed word embeddings** capture other forms of information that **result in more accurate feature representations, which in turn results in better model performance.**

These embeddings are useful for keyword/search expansion, semantic search and information retrieval.

Why ATT&CK BERT Embeddings?

ATT&CK BERT is very close to our context CyberSecurity CVE Attack (Impact) so use that rather than create our own embeddings.

all-mpnet-base-v2 is top of the leaderboard - but for our context is outperformed by ATT&CK BERT because it has cybersecurity context.

A CyberSecurity Context-Informed BERT Model Performs Best for our Use Case

polysemy: word has multiple meanings depending on context

https://www.sbert.net/docs/pretrained_models.html

Topic Modeling

Unsupervised

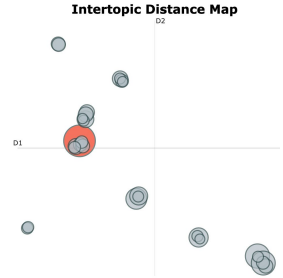
Topic Modeling

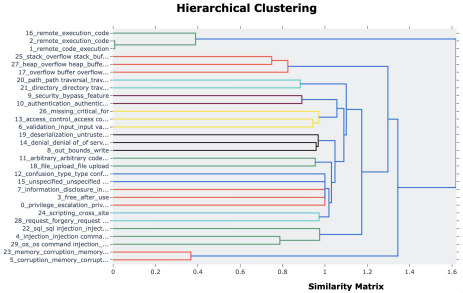
BERTopic: topic modeling technique that leverages 🤗 transformers and c-TF-IDF. In general, it outperforms other methods e.g. LSA, LDA, Top2Vec.

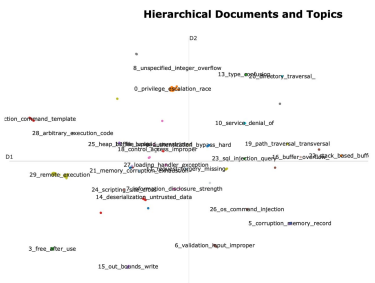
BERTopic Topic Modeling

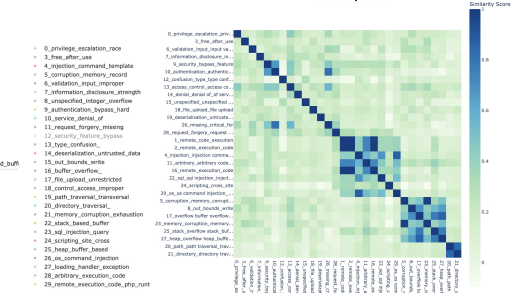
Topic Modeling

BERTopic: topic modeling technique that leverages 🤗 transformers and c-TF-IDF. In general, it outperforms other methods e.g. LSA, LDA, Top2Vec.









Scan CVE Descriptions to get Topics (Impacts)

Discover the underlying themes and patterns in a collection of documents (CVE Descriptions).

*"**Topic modeling** is an **unsupervised** machine learning technique that's capable of scanning a set of documents, detecting word and phrase patterns within them, and automatically clustering word groups and similar expressions that best characterize a set of documents."*

"At the end of your topic modeling analysis, you'll receive collections of documents** that the algorithm has grouped together, as well as **clusters of words and expressions that it used to infer these relations."

Seed Topics guide the topic modeling approach by setting several seed topics to which the model will converge to. These techniques allow the user to set a predefined number of topic representations that are sure to be in documents.

Seed Words To make sure that certain domain specific words are weighted higher and are more often used in topic representations, you can set any number of seed_words.

BertTopic Topic Modeling can be used to extract Topics (Impacts, Consequences) from CVEs

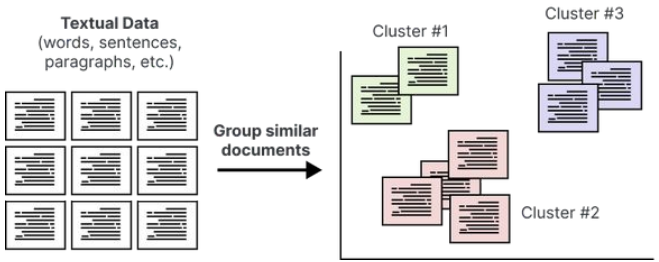
https://maartengr.github.io/BERTopic/getting_started/guided/guided.html

https://maartengr.github.io/BERTopic/getting_started/seed_words/seed_words.html

BERTopic Topic Modeling

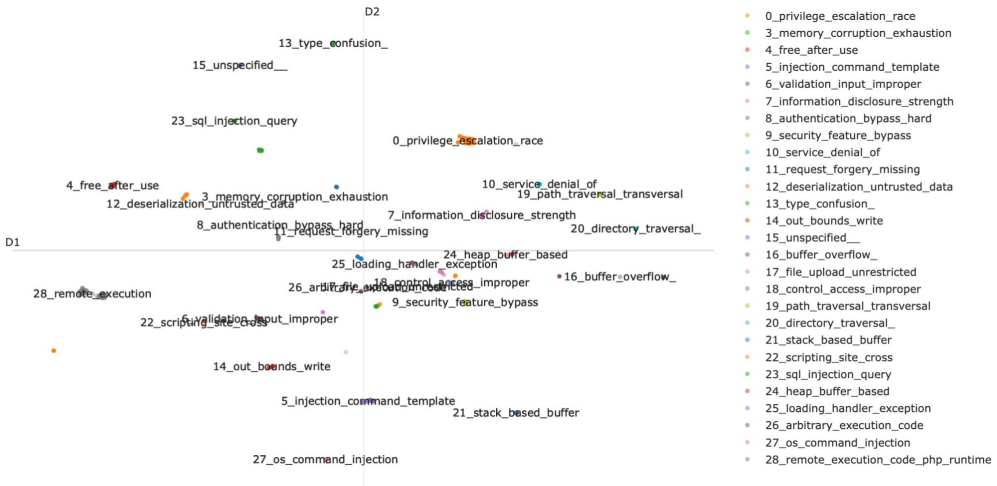
Why Topic Modeling?

Before diving into a classification task, text clustering allows for getting an intuitive understanding of the task but also of its complexity. BERTopic is a topic modeling technique that assumes that clusters of semantically similar documents are a powerful way of generating and describing clusters. The documents in each cluster are expected to describe a major theme and combined they might represent a topic.



Each CVE/vulnerability description is a “document”

Topic Model of CISA KEV



This is a snapshot of an interactive diagram; can select/filter, zoom in and see each document, and the documents beside it.

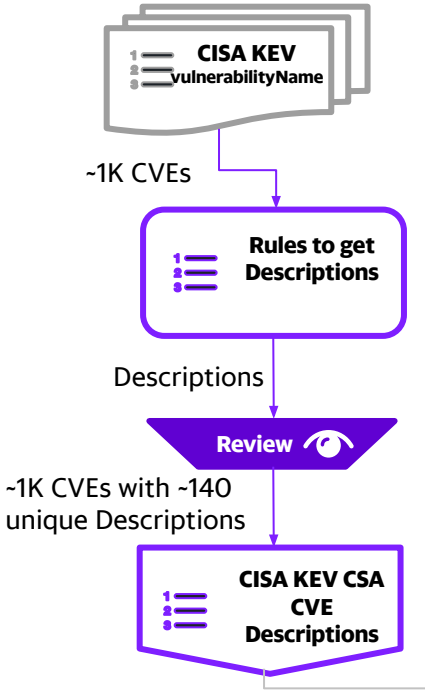
TODO

Image and paragraph from Book co-authored by BertTopic creator: https://www.oreilly.com/library/view/hands-on-large-language/9781098150952/

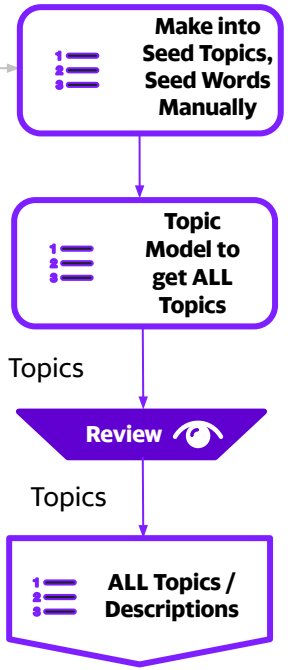
Topic modelling used in research for CVE analysis: Stephan Neuhaus and Thomas Zimmermann. 2010. Security trend analysis with cve topic models.

Topic Modeling of CISA KEV shortDescription

Rule-Based



Topic Modeling



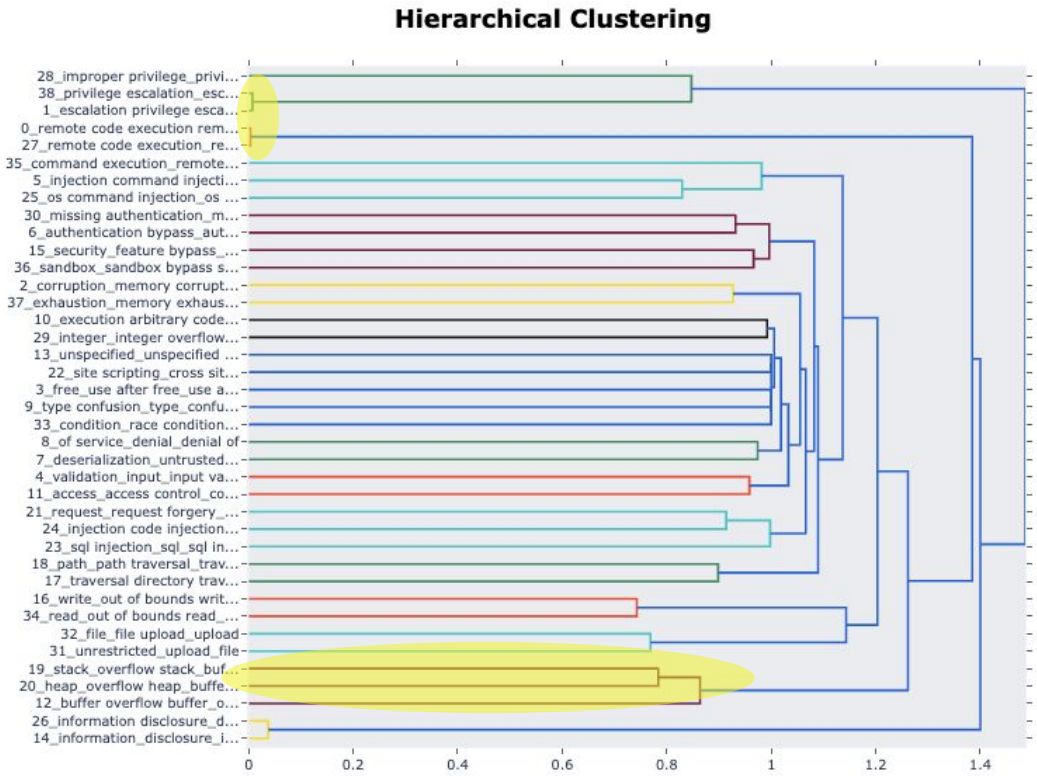

```
from bertopic import BERTopic
from sentence_transformers import SentenceTransformer

sentence_model = SentenceTransformer("basel/ATTACK-BERT")

topic model = BERTopic(language="english",
calculate probabilities=True, verbose=True, n gram_range=(1,4),
min_topic_size=5, embedding_model=sentence_model)

topics, probs = topic_model.fit_transform(docs)
```

1. n_gram_range=(1,4)
   a. Some phrases have up to 4 words e.g. "heap based buffer overflow", "server side request forgery", “cross site request forgery"
   b. Some phrases have minimum 1 word e.g. spoofing, bypass, “unspecified”
2. ATT&CK-BERT
   a. custom embeddings (rather than default embeddings that are not aware of our CyberSecurity context)
   b. fine-tuning the topic keywords (rather than default count of words c-TF-IDF)
      i. c-TF-IDF generated topics do not consider the semantic nature of words in a topic which could end up creating topics with stopwords. Use the module representation_model KeyBERTInspired() to fine-tune the topic keywords based on their semantic similarity to the topic.

We let Topic Modeling do the work to group the Descriptions

Rule-Based Analysis of CISA KEV vulnerabilityName is used to Bootstrap Topic Modeling of CISA KEV shortDescription

Topic Modeling of CISA KEV shortDescription Hierarchy



In our configuration, no limit was placed on the number of topics.

From ~1K CVEs containing 140 unique values to ~40 topics

- A topic is a group of related documents (stripped CISA KEV entries in this case)
- The heading is the top 3 words for that topic
  - it is possible to change this automatically e.g. using an LLM to suggest a heading name

The generated hierarchy shows where we can merge topics
e.g. stack_overflow, and heap_overflow to buffer_overflow where stack_overflow topic contains e.g. “stack based buffer overflow” “stack buffer overflow”

(Unsupervised) Topic Modeling did a good job of gathering related entries together on this simple dataset...

Guided Topic Modeling WordCloud for CISA KEV

Rule-Based Analysis of CISA KEV vulnerabilityName is used to Bootstrap Topic Modeling of CISA KEV shortDescription

User Need

A Need

Jay Jacobs • 1st
Chief Data Scientist, Founder and Partner at Cyentia Institute
1d •

Has anyone dug into CWEs?
I mean **really** got deep into the data and relationships within the Common Weakness Enumerations?

The attached plot is after several hours last night just parsing the xml while making several lame jokes to myself about the 90's calling.

Here's the challenge: I want to reduce the 900+ CWEs, and all of the other CWE-IDs found in NVD to a meaningful and much smaller subset of CWEs. For example, Heap-based (122) and stack-based (121) buffer overflows (and several others) roll up to base of CWE-787 ("out-of-bounds write") in the research view CWE-1000, but also to category CWE-1218 ("Memory Buffer Errors") in view CWE-699.

Plus, NVD data is all over the place, barely following and just winking at CWE-1003 as various other views, categories and non-1003 are used.

This plot attempts to visualize all of the relationships in the world of CWEs.

Any advice on where to go?

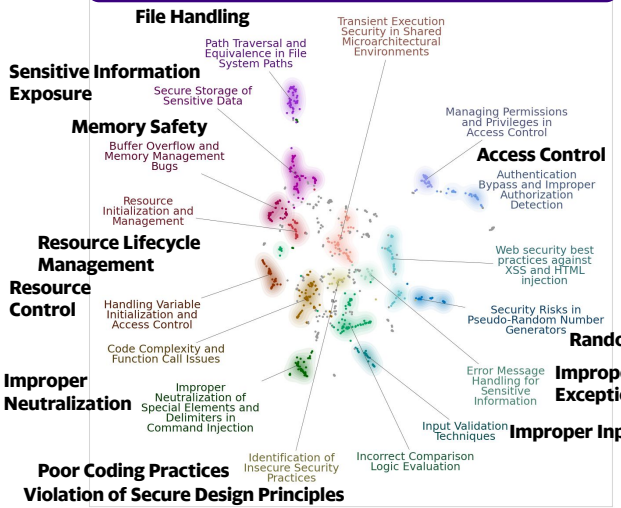
Chris Madden • You
Distinguished Technical Security Engineer
1d •

Yes :)
I've used (and would recommend) Topic Modeling - specifically BERTopic with Cybersecurity embeddings to cluster semantically similar CWEs (automatically/unsupervised).

...see more

Like · 9 | Reply · 5 Replies

Topic Model of the ~1400 CWEs in 1400 View



Chris Madden • You
Distinguished Technical Security Engineer
1d •

Jay Jacobs following up on my previous comment about BERTopic... a few minutes effort (to load the 1400.csv file into an existing notebook I had) gives attached

Each dot is an entry/CWE in 1400.csv

The Categories in 1400 View

1400 - Comprehensive Categorization for Software Assurance Trends

- Comprehensive Categorization: Access Control - *(1396)*
- Comprehensive Categorization: Comparison - *(1397)*
- Comprehensive Categorization: Component Interaction - *(1398)*
- Comprehensive Categorization: Concurrency - *(1401)*
- Comprehensive Categorization: Encryption - *(1402)*
- Comprehensive Categorization: Exposed Resource - *(1403)*
- Comprehensive Categorization: File Handling - *(1404)*
- Comprehensive Categorization: Improper Check or Handling of Exceptional Conditions - *(1405)*
- Comprehensive Categorization: Improper Input Validation - *(1406)*
- Comprehensive Categorization: Improper Neutralization - *(1407)*
- Comprehensive Categorization: Incorrect Calculation - *(1408)*
- Comprehensive Categorization: Injection - *(1409)*
- Comprehensive Categorization: Insufficient Control Flow Management - *(1410)*
- Comprehensive Categorization: Insufficient Verification of Data Authenticity - *(1411)*
- Comprehensive Categorization: Memory Safety - *(1399)*
- Comprehensive Categorization: Poor Coding Practices - *(1412)*
- Comprehensive Categorization: Protection Mechanism Failure - *(1413)*
- Comprehensive Categorization: Randomness - *(1414)*
- Comprehensive Categorization: Resource Control - *(1415)*
- Comprehensive Categorization: Resource Lifecycle Management - *(1416)*
- Comprehensive Categorization: Sensitive Information Exposure - *(1417)*
- Comprehensive Categorization: Violation of Secure Design Principles - *(1418)*

BERTopic has many applications when trying to make sense of a lot of data!

LinkedIn post April 19, 2024

https://cwe.mitre.org/data/definitions/1400.html

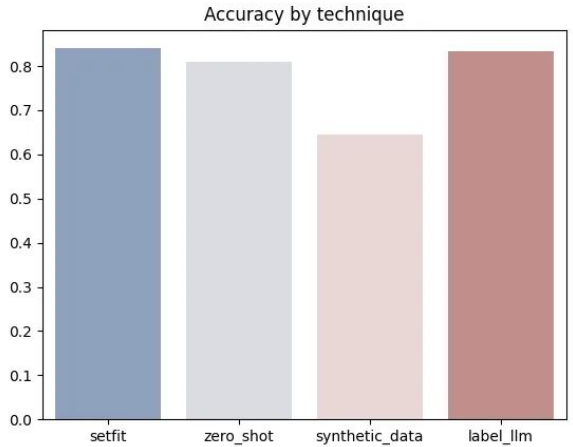
Leveraging an LLM - ChatGPT 4

OpenAI Large Language Model:
General Purpose LLM

Approaches to using an LLM

OpenAI Large Language Model:
General Purpose LLM

1. Using an LLM in a zero shot fashion for the application directly
2. Creating synthetic data to train the first model
3. Label data using an LLM
4. Validate labelled data



LLMs can be used to prototype, and assist in training a smaller dedicated Model

https://medium.com/mantisnlp/prototyping-with-large-language-models-916a6e85ada8

ChatGPT4: Human Zero-Shot to GPT Few-Shot Prompt

A human-created one-shot prompt to the model

You

classify the following VULNERABILITY=== using one or more of the LABELS=== below

Only use LABELS from the list provided in LABELS===.
Check that the answer includes only LABELS from the list provided in LABELS===.
Provide a probability score to 2 decimal places for each label.
Show the VULNERABILITY=== then the LABELS=== by probability score showing the highest probability score first.

VULNERABILITY===
"stack based buffer overflow in the XYZ function in win32k.sys in microsoft windows allows local users to gain privileges, run arbitrary code, and bypass the user account control (uac) feature."

LABELS===
'Command Injection', 'sql injection', 'Code Execution', 'Bypass Protection Mechanism', 'Privilege Escalation'

‹ 3 / 3 ›

ChatGPT

Based on the given vulnerability description, the classification with corresponding probability scores is as follows:

VULNERABILITY:
"stack based buffer overflow in the XYZ function in win32k.sys in Microsoft Windows allows local users to gain privileges, run arbitrary code, and bypass the user account control (UAC) feature."

LABELS by probability score:

- Code Execution: 0.95
- Privilege Escalation: 0.90
- Bypass Protection Mechanism: 0.85
- Command Injection: 0.30
- sql injection: 0.00

GPT Generated Few-Shot Prompt

You

now give me a prompt for few shot classification of vulnerability cves

ChatGPT

For few-shot classification of Common Vulnerabilities and Exposures (CVEs), we will craft a prompt that includes a few examples of CVE descriptions along with their correct classifications. This approach will guide the model on how to classify a new, unseen CVE description based on the examples provided. Let's assume your labels are 'Code Execution', 'Privilege Escalation', 'Bypass Protection Mechanism', 'Command Injection', and 'SQL Injection'.

Here's how you could structure your prompt:

LLMs can generate FewShot Prompts - learning from previous prompts

ChatGPT 4 Few-Shot

GPT Generated Few-Shot Prompt

Instructions to AI: Below are descriptions of various cybersecurity vulnerabilities, each followed by a classification label. Read these descriptions and labels carefully.
After the examples, there is a new vulnerability description. Classify this new vulnerability by choosing the most appropriate label from the ones provided in the examples.

LABELS===
'Command Injection', 'sql injection', 'Code Execution', 'Bypass Protection Mechanism', 'Privilege Escalation'

Example 1:
Description: An SQL injection vulnerability in the 'inputData' parameter of the 'DataProcessor' module allows attackers to execute arbitrary SQL commands via the user input field.
Label: SQL Injection

Example 2:
Description: A flaw in the 'executeCommand' function allows remote attackers to execute arbitrary code via a crafted request.
Label: Code Execution

Example 3:
Description: The application fails to properly validate user session tokens, which can be exploited to bypass authentication controls and gain unauthorized access to the system.
Label: Bypass Protection Mechanism

Example 4:
Description: A buffer overflow in the 'FileParser' component allows local users to gain elevated privileges by inserting a specially crafted file into the system.
Label: Privilege Escalation

New Vulnerability to Classify:
Description: a stack based buffer overflow in the XYZ function in win32k.sys in microsoft windows allows local users to gain privileges, run arbitrary code, and bypass the user account control (uac) feature.

Your Task: Based on the examples provided, classify the new vulnerability by selecting the most appropriate labels.
Only use LABELS from the list provided in LABELS===.
Check that the answer includes only LABELS from the list provided in LABELS===.
Provide a probability score to 2 decimal places for each label.
Sort the labels by probability score showing the highest probability score first.

Response

Given the structure and requirements of this task, we're aiming to classify a new vulnerability based on the provided examples and labels. Let's analyze the new vulnerability's description and map it to the most relevant labels, accompanied by estimated probability scores.

New Vulnerability to Classify:
Description: "A stack-based buffer overflow in the XYZ function in win32k.sys in Microsoft Windows allows local users to gain privileges, run arbitrary code, and bypass the user account control (UAC) feature."

Given the descriptions and the labels available, the relevant labels for this new vulnerability, based on the components of the description, are:

1. **Code Execution** - The phrase "run arbitrary code" directly implies this category.
2. **Privilege Escalation** - Mention of "allows local users to gain privileges" indicates this classification.
3. **Bypass Protection Mechanism** - The ability to "bypass the user account control (UAC) feature" falls under this label.

The other two labels, 'Command Injection' and 'SQL Injection', do not directly apply to the description provided. Therefore, their probability scores would be significantly lower or effectively zero in this context.

Let's now estimate probability scores for each relevant label, keeping in mind the specifics of the vulnerability description:

- **Code Execution**: This seems to be a primary outcome of exploiting the vulnerability, so we'll assign a high probability.
- **Privilege Escalation**: Also a direct consequence mentioned, so this too will have a high probability.
- **Bypass Protection Mechanism**: Specifically mentioned, indicating a high probability, but it might be viewed as a secondary effect compared to direct code execution and privilege escalation.
- **Command Injection**: Not mentioned or implied in the vulnerability description.
- **SQL Injection**: Also not relevant to the description provided.

Next, I'll assign these probability scores.
Based on the provided examples and the description of the new vulnerability, the classification with corresponding probability scores is as follows:

- Code Execution: 0.95
- Privilege Escalation: 0.90
- Bypass Protection Mechanism: 0.85
- Command Injection: 0.00
- SQL Injection: 0.00

The LLM performed better with the Few Shot prompt it generated.

Validate Labelled Dataset (ChatGPT 4)

Zero-Shot Zero-preprocessing: 8 examples

You

You are a cybersecurity and vulnerability management expert, and you review CVE vulnerability descriptions and assign attributes to them from a set of predefined attributes.

I have a dataset of CVE vulnerability descriptions (that are stemmed) in @data that have been classified with attributes listed in @attribute @@class@@

This dataset may contain errors. I want you to find them.
Specifically, I want you to indicate which assigned attributes for descriptions you disagree with, how confident you are, your rationale, and the attributes you think should be assigned instead

@attribute @@class@@ {AuthenticationBypass, CrossSiteScripting, DenialOfService, DirectoryTraversal, FileInclude, InformationDisclosure, Overflow, RemoteCodeExecution, SQLinjection, UnspecifiedVulnerability}

@data
"befor when run certain environ certain memcmp function remot attack bypass authent repeate authent same incorrect password which eventu caus token comparison succe due improperli check return valu ", AuthenticationBypass
"doe properli handl command line argument which remot attack bypass protect mechan execut arbitrari code leverag improp interact between compon queri string begin sequenc ", AuthenticationBypass
"allow remot authent site administr bypass intend access restrict deactiv network wide plugin unspecifi vector ", AuthenticationBypass
"condit firewal authent proxi featur through remot attack bypass authent bypass consent web page craft request aka bug id ", AuthenticationBypass
"nntp server nntpd cyru imapd remot attack bypass authent send authinfo user command without send addit authinfo pass command ", AuthenticationBypass
"earlier doe trigger fatal error when identifi miss invalid acl configur which could lead less restrict acl than intend administr ", AuthenticationBypass
"when process configur file pars empti access control list acl includ acl without defin auth scheme wai effect remov argument which could remot attack bypass intend acl administr ignor parser warn ", AuthenticationBypass
"function earlier doe properli set debug context when handl overs http repli header which might remot attack poison cach bypass access control base header size ", AuthenticationBypass
"common code infrastructur compon befor when ldap secur aka anonym bind enabl remot attack bypass password authent establish databas connect unspecifi vector ", AuthenticationBypass
"crypt function when hash type us return valu salt argument instead hash string which might remot attack bypass authent arbitrari password differ vulner than ", AuthenticationBypass
"unspecifi vulner oracl glassfish server compon remot attack execut arbitrari code unknown vector relat administr ", AuthenticationBypass

ChatGPT

Analyzing the given CVE vulnerability descriptions along with their assigned attributes, I will determine whether the classifications are accurate and provide my assessment on each:

1. **Description**: "befor when run certain environ certain memcmp function remot attack bypass authent repeatedli authent same incorrect password which eventu caus token comparison succe due improperli check return valu"
   - **Assigned Attribute**: AuthenticationBypass
   - **Agreement**: Agree
   - **Confidence**: High
   - **Rationale**: The description refers to attackers repeatedly authenticating with an incorrect password and eventually succeeding, indicating an authentication bypass due to improper check of return values.
   - **Suggested Attribute**: None needed
2. **Description**: "doe properli handl command line argument which remot attack bypass protect mechan execut arbitrari code leverag improp interact between compon queri string begin sequenc"
   - **Assigned Attribute**: AuthenticationBypass
   - **Agreement**: Disagree
   - **Confidence**: Medium
   - **Rationale**: The description hints at the possibility of executing arbitrary code through improper handling of command-line arguments, which suggests a remote code execution vulnerability rather than solely authentication bypass.
   - **Suggested Attribute**: RemoteCodeExecution

7. **Description**: "when process configur file pars empti access control list acl includ acl without defin auth scheme wai effect remov argument which could remot attack bypass intend acl administr ignor parser warn"
   - **Assigned At ” te**: AuthenticationBypass
   - **Agreement**: Disagree
   - **Confidence**: High
   - **Rationale**: This involves ACL misconfigurations potentially leading to unintended access, not specifically bypassing authentication mechanisms.
   - **Suggested Attribute**: InformationDisclosure
8. **Description**: "function earlier doe properli set debug context when handl overs http repli header which might remot attack poison cach bypass access control base header size"
   - **Assigned Attribute**: AuthenticationBypass
   - **Agreement**: Disagree
   - **Confidence**: Medium
   - **Rationale**: The issue described relates to cache poisoning and bypassing based on HTTP header sizes, which leans more towards an integrity issue than direct authentication bypass.
   - **Suggested Attribute**: DenialOfService (related to poisoning impacting availability/integrity)

The whole Training File of ~500 CVE labeled Descriptions

You

train.arff
File

now do the same with the attached file. But only tell me about the ones that you disagree with

ChatGPT

First, I'll need to read the contents of the uploaded file to examine the data. Let me start by checking the format and the data in the file.
The dataset contains CVE descriptions that have been classified into various vulnerability categories. Each line in the data section consists of a vulnerability description followed by its assigned attribute within curly braces. Now, I'll extract and evaluate a sample of these entries to identify any misclassifications based on the descriptions provided. [›]
I'll analyze these descriptions to identify potential misclassifications based on the nature of the vulnerabilities described. Specifically, I will look for descriptions where the assigned attribute might not align with the typical characteristics of that vulnerability type. Let's start with a sample of descriptions for a more focused review. [›]
Here are some examples from the dataset where the assigned attributes might not correctly describe the nature of the vulnerabilities based on the descriptions provided:

1. **Description**: "unspecifi vulner befor remot attack execut arbitrari code"
   - **Assigned Attribute**: AuthenticationBypass
   - **Potential Correct Attribute**: RemoteCodeExecution
   - **Rationale**: The description indicates the execution of arbitrary code, which typically suggests a Remote Code Execution vulnerability rather than an Authentication Bypass.
2. **Description**: "cross site script xss vulner project modul remot attack"
   - **Assigned Attribute**: CrossSiteScripting
   - **Potential Correct Attribute**: CrossSiteScripting
   - **Rationale**: Although this one matches correctly, it was initially flagged due to the specific mention of XSS (Cross-Site Scripting), confirming its classification.
3. **Description**: "session persist implement mishandl session attack"
   - **Assigned Attribute**: AuthenticationBypass
   - **Potential Correct Attribute**: RemoteCodeExecution
   - **Rationale**: Mismanagement of session persistence suggests issues potentially leading to session hijacking or unauthorized actions, which might be better classified under a broader threat category like Remote Code Execution if it leads to arbitrary code execution.
4. **Description**: "pl perl pl tcl befor befor befor do properli process"
   - **Assigned Attribute**: AuthenticationBypass
   - **Potential Correct Attribute**: RemoteCodeExecution
   - **Rationale**: The description suggests improper processing which might lead to arbitrary code execution, aligning more with Remote Code Execution.
5. **Description**: "replic code possibl through through through the sql"
   - **Assigned Attribute**: SQLinjection
   - **Potential Correct Attribute**: RemoteCodeExecution
   - **Rationale**: If the replication of code through SQL commands leads to execution of arbitrary commands, this might also involve Remote Code Execution.

https://github.com/jssrp2018/CVErizer/blob/master/CVErizer_replication_package/RQ1/test.arff

https://github.com/jssrp2018/CVErizer/blob/master/CVErizer_replication_package/RQ1/train.arff

Named Entity Recognition

Unsupervised

Named Entity Recognition

GLiNER:
Generalist and Lightweight model for Named Entity Recognition (Nov 2023)

Named Entity Recognition - for CyberSecurity

Named Entity Recognition

SecBERT-our-data:
a cybersecurity domain-specific NER model for extracting cybersecurity named entities (2022)

Inference API

Token Classification — Examples

_functions.php in cpCommerce 1.2.x, possibly including 1.2.9, sends a redirect but does not exit when it is called directly, which allows remote attackers to bypass a protection mechanism to conduct remote file inclusion and directory traversal attacks, execute arbitrary PHP code, or read arbitrary files via the GLOBALS[prefix] parameter, a different vector than CVE-2003-1500.

Compute

Computation time on cpu: 0.066 s

_ OBSERVED_DATA functions .php OBSERVED_DATA in cpCommerce 1.2 GENERAL_TOOL .x, possibly including 1.2.9, sends a redirect but does not exit when it is called directly ATTACK_PATTERN, which allows remote attackers to bypass a protection mechanism to conduct remote file inclusion and directory traversal attacks, IMPACT execute arbitrary PHP code, IMPACT or read arbitrary files via the GLOBALS[prefix] parameter IMPACT, a different vector than CVE-2003-1500 VULNERABILITY.

Example: Invoking a pre-trained model via 🤗Inference API

```
import requests
headers = {"Authorization": f"Bearer {API_TOKEN}"}
API_URL =
"https://api-inference.huggingface.co/models/anonymouspd/SecBERT-our-data"
def query(payload):
   response = requests.post(API_URL, headers=headers, json=payload)
   return response.json()

text = '_functions.php in cpCommerce 1.2.x, possibly including 1.2.9, sends a
redirect but does not exit when it is called directly, which allows remote
attackers to bypass a protection mechanism to conduct remote file inclusion and
directory traversal attacks, execute arbitrary PHP code, or read arbitrary files via
the GLOBALS[prefix] parameter, a different vector than CVE-2003-1500'

data = query({"inputs": text})

data
<SNIP> EDITED TO FIT / SHOW RELEVANT CONTENT ONLY <SNIP>
{'eg': 'THREAT_ACTOR', 'score': 0.7037,'word': 'remote attackers',
{'eg': 'ATTACK_PATTERN', 'score': 0.7106,'word': 'remote file inclusion',
 {'eg': 'ATTACK_PATTERN', 'score': 0.8672, 'word': 'directory traversal attacks',
 {'eg': 'ATTACK_PATTERN', 'score': 0.8287097811698914, 'word': 'execute arbitrary
php code',
#'eg' shortened from 'entity_group'
```

This is an extract from a real example. Not all text works this well!

SecureBERT outperformed SecBERT on 10K sample of CVEs.

NER can extract Attack Patterns, Impacts, and other entities from text (reasonably well)

https://huggingface.co/anonymouspd/SecBERT-our-data https://huggingface.co/ehsanaghaei/SecureBERT

Named Entity Recognition - General Purpose SoTA

Named Entity Recognition

GLiNER:
Generalist and Lightweight model for Named Entity Recognition (Nov 2023)

GLiNER demonstrates strong performance, outperforming both ChatGPT and fine-tuned LLMs in zero-shot evaluations on various NER benchmarks (Paper)

Given no previous training (zero-shot) on a cybersecurity corpus, and no previous knowledge of the defined entities/labels, it can perform well with cybersecurity text

- better than other NERs trained on a cybersecurity corpus with fixed pre-defined entities/labels (SecureBERT, SecBERT) based on an evaluation of 10K CVEs

The Threshold value can significantly impact the results.

GLiNER utilizes small bidirectional LMs and treats the NER task as matching entity types with textual spans in a latent space.
Trained on Universal-NER dataset https://universal-ner.github.io/

GLiner for NER of Impacts

```
from gliner import GLiNER
model = GLiNER.from_pretrained("urchade/gliner_largev2")
model.eval()

text = '_functions.php in cpCommerce 1.2.x, possibly including 1.2.9, sends a
redirect but does not exit when it is called directly, which allows remote
attackers to bypass a protection mechanism to conduct  remote file inclusion
and directory traversal attacks, execute arbitrary PHP code, or read arbitrary
files via the GLOBALS[prefix] parameter, a different vector than
CVE-2003-1500'

labels = ['Threat Actor', 'Vulnerability', 'Impact', 'Attack Pattern']

entities = model.predict_entities(text, labels,  threshold=0.4)

for entity in entities:
   print(entity["text"], "=>", entity["label"])

remote attackers => Threat Actor
remote file inclusion => Attack Pattern
directory traversal attacks => Attack Pattern
execute arbitrary PHP code => Attack Pattern
CVE-2003-1500 => Vulnerability
```

This is the full code for a real example - with annotations for clarity. Not all text works this well!

I tried using the Impact labels instead as the labels.... didn't work.. but no harm in trying :)

SoTA NER models with no prior knowledge can recognise CyberSecurity Entities

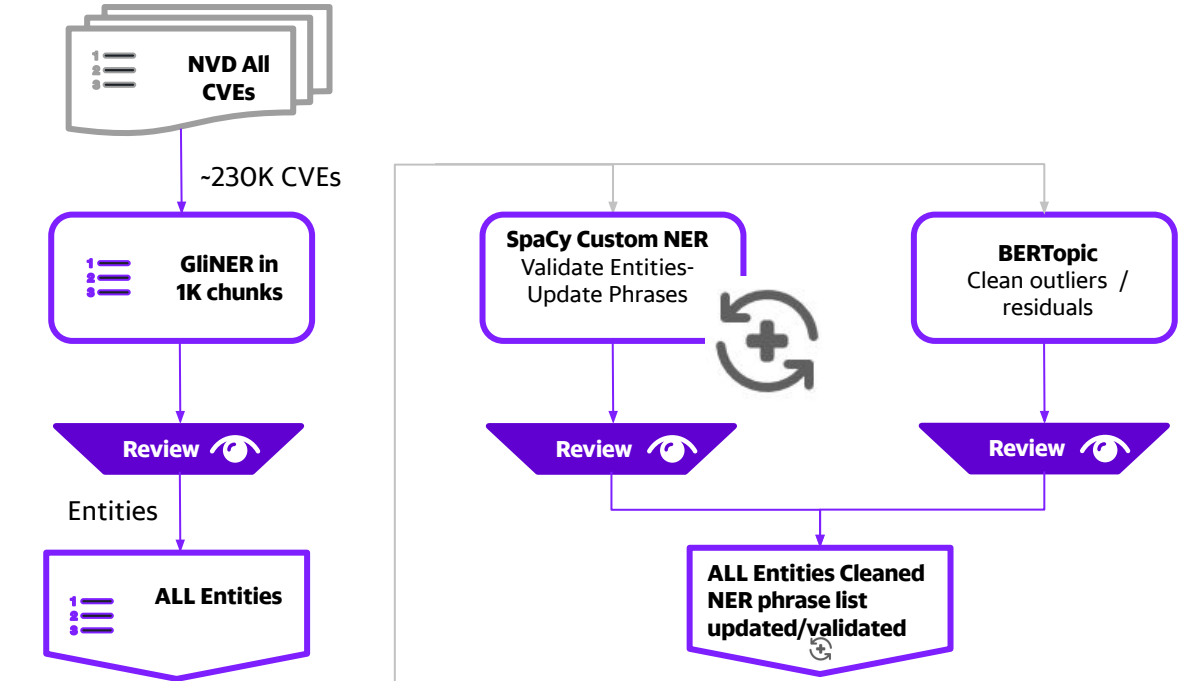
GliNER Output on CVE Description

CVE Description	GliNER Output
Accellion FTA 9_12_370 and earlier is affected by SQL injection via a crafted Host header in a request to document_root.html. The fixed version is FTA_9_12_380 and later.	['SQL injection']
Accellion FTA 9_12_411 and earlier is affected by SSRF via a crafted POST request to wmProgressstat.html. The fixed version is FTA_9_12_416 and later.	['SSRF']
Acrobat Reader DC versions versions 2020.013.20074 (and earlier), 2020.001.30018 (and earlier) and 2017.011.30188 (and earlier) are affected by a heap-based buffer overflow vulnerability. An unauthenticated attacker could leverage this vulnerability to achieve arbitrary code execution in the context of the current user. Exploitation of this issue requires user interaction in that a victim must open a malicious file.	['heap-based buffer overflow vulnerability', 'arbitrary code execution']
Acrobat Reader DC versions versions 2021.001.20150 (and earlier), 2020.001.30020 (and earlier) and 2017.011.30194 (and earlier) are affected by a Use After Free vulnerability. An unauthenticated attacker could leverage this vulnerability to achieve arbitrary code execution in the context of the current user. Exploitation of this issue requires user interaction in that a victim must open a malicious file.	['Use After Free', 'arbitrary code execution']
Adobe ColdFusion Update 5 and earlier versions, ColdFusion 11 Update 13 and earlier versions have an exploitable Deserialization of Untrusted Data vulnerability. Successful exploitation could lead to arbitrary code execution.	['Deserialization of Untrusted Data vulnerability', 'arbitrary code execution']
Adobe ColdFusion versions July 12 release (2018.0.0.310739), Update 6 and earlier, and Update 14 and earlier have an unrestricted file upload vulnerability. Successful exploitation could lead to arbitrary code execution.	['unrestricted file upload vulnerability', 'arbitrary code execution']
A use-after-free vulnerability was discovered in Adobe Flash Player before 28.0.0.161. This vulnerability occurs due to a dangling pointer in the Primetime SDK related to media player handling of listener objects. A successful attack can lead to arbitrary code execution. This was exploited in the wild in January and February 2018.	['use-after-free', 'dangling pointer', 'arbitrary code execution']
Amcrest cameras and NVR are vulnerable to a stack-based buffer overflow over port 37777. An authenticated remote attacker can abuse this issue to crash the device and possibly execute arbitrary code.	['stack-based buffer overflow', 'crash the device', 'arbitrary code']
A use-after-free in binder.c allows an elevation of privilege from an application to the Linux Kernel. No user interaction is required to exploit this vulnerability, however exploitation does require either the installation of a malicious local application or a separate vulnerability in a network facing application.Product: AndroidAndroid ID: A-141720095	['use-after-free']
In binder_transaction of binder.c, there is a possible out of bounds write due to an incorrect bounds check. This could lead to local escalation of privilege with no additional execution privileges needed. User interaction is not needed for exploitation.Product: AndroidVersions: Android kernelAndroid ID: A-145988638References: Upstream kernel	['out of bounds write', 'incorrect bounds check', 'local escalation of privilege']

GliNER does really well given it had no training or prompting on CyberSecurity!

This input text was NOT pre-processed (lemmatized, with stop words, numbers,.. removed). This keeps the Semantic context.

All CVEs KeyPhrases Extraction



Recipe

1. Extract Entities from all CVEs using GLiNER in 1K chunks
2. All Entities matched against the Spacy Custom NER phrase list.
3. Higher count Entities are added to the Spacy Customer NER phrase list
4. Lower count Entities are filtered out using BERTtopic: lower count entities will
   a. be outliers (BERTtopic groups these into a -1 topic)
   b. have lower probability for the topic assigned

The validation and cleaning of Entities is largely automated. Review = me + LLM.

https://en.wikipedia.org/wiki/Swedish_Chef

Some Interesting Finds by GliNER

CVE-2021-27374 VertiGIS WebOffice 10.7 SP1 before patch20210202 and 10.8 SP1 before patch20210207 allows attackers to achieve "Zugriff auf Inhalte der WebOffice Applikation.""",['**Zugriff auf Inhalte der WebOffice Applikation**']

CVE-2021-41661 "Church Management System version 1.0 is affected by a SQL anjection vulnerability through creating a user with a PHP file as an avatar image, which is accessible through the /uploads directory. This can lead to RCE on the web server by uploading a PHP webshell.","['SQL **anjection** vulnerability', 'RCE']"

CVE-2021-35250 A researcher reported a Directory Transversal Vulnerability in Serv-U 15.3. This may allow access to files relating to the Serv-U installation and server files. This issue has been resolved in Serv-U 15.3 Hotfix 1.","['Directory **Transversal** Vulnerability']"

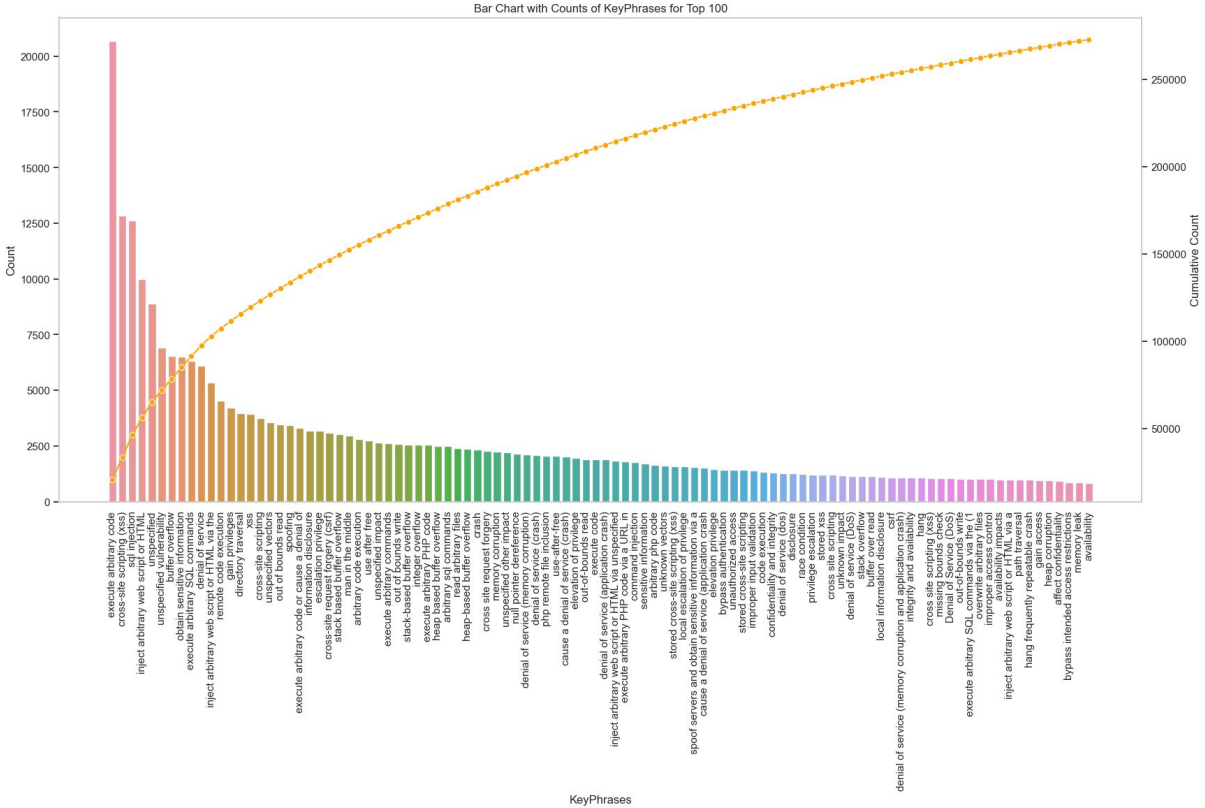
CVE-2017-18683 An issue was discovered on Samsung mobile devices with L(5.0/5.1) and M(6.0) software. SVoice allows Hare Hunting during application installation. The Samsung ID is SVE-2016-6942 (February 2017).,**['Hare Hunting']**

CVE-2021-22860 EIC e-document system does not perform completed identity verification for sorting and filtering personnel data. The vulnerability allows remote attacker to obtain users' credential information without logging in the system, and further acquire the privileged permissions and execute arbitrary commends.,**['arbitrary commends']**

CVE-2018-12062,"The sell function of a smart contract implementation for SwftCoin (SWFTC), a tradable Ethereum ERC20 token, allows a potential trap that could be used to cause financial damage to the seller, because of overflow of the multiplication of its argument amount and a manipulable variable sellPrice, aka the ""tradeTrap"" issue.","**['financial damage', 'overflow', 'tradeTrap']**"

GliNER is robust against typos, and Concept Drift e.g. new attacks

Distribution of KeyPhrases



- ~230K CVE Descriptions
  - ~6K were not (yet) assigned KeyPhrases
- 106K unique KeyPhrases
- 568K KeyPhrases
- Top 100 KeyPhrases (100/106K = 0.001%) account for ~~50% (~273K/568K) of KeyPhrases in the 230K CVE Descriptions

Distribution of KeyPhrases follows a Pareto distribution

Some CVEs were not assigned KeyPhrases

1. "CVE-2021-39737","Product: AndroidVersions: Android kernelAndroid ID: A-208229524References: N/A"
2. "CVE-1999-0618","The rexec service is running."
3. "CVE-1999-0584","A Windows NT file system is not NTFS."
4. "CVE-1999-0556","Two or more Unix accounts have the same UID."
5. "CVE-1999-0539","A trust relationship exists between two Unix hosts."
6. "CVE-1999-0326","Vulnerability in HP-UX mediainit program."
7. "CVE-2021-37551","In JetBrains YouTrack before 2021.2.16363, system user passwords were hashed with SHA-256."
8. "CVE-2021-37588","In Charm 0.43, any two users can collude to achieve the ability to decrypt YCT14 data."

Only 2.6% (6K/230K) were not assigned KeyPhrases (yet...)

SetFIT

setFIT: (Sentence Transformer Fine-tuning)
Uses Contrastive Training to reduce needed labelled data.
Few-Shot Learning (FSL) (Sep 2022)

LAGONN:
improves SetFit's performance by modifying text with the nearest neighbors in the training data. (Jan 2024)

Few Shot Learning

Why Few Shot?

When there is limited labelled data available, Few Shot learning is a good option.

- "Few Shot" prompting: Use a few explicit examples (or shots) to guide the AI to respond in a specific way.
- Language models are few-shot learners (May 2020)
  We also conjecture, based on past literature, that a large bidirectional model would be stronger at fine-tuning than GPT-3. Making a bidirectional model at the scale of GPT-3, and/or trying to make bidirectional models work with few- or zero-shot learning, is a promising direction for future research, and could help achieve the "best of both worlds".

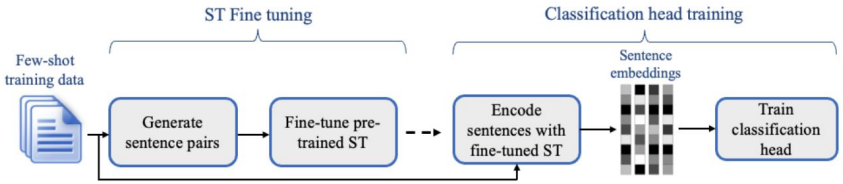


Figure 2: SETFIT 's fine-tuning and training block diagram.

Ask ChatGPT 4 is ChatGPT or SetFit better (extracts)

You
is chatgpt 3.5 or 4 better for zero shot or few shot classification of CVE vulnerabilities than setfit

ChatGPT

- **Specialized for Few-Shot Learning:** SetFit is specifically designed for few-shot learning scenarios. It is optimized to quickly adapt to new tasks with very few examples, potentially making it more efficient for classification tasks when only a handful of labeled instances are available.
- **Customizable and Fine-Tuned:** Unlike the broad capabilities of GPT models, SetFit can be fine-tuned on a specific dataset, including CVE classifications. This tuning can potentially lead to higher accuracy in specific classification tasks since the model learns directly from the task-relevant data.
- **Efficiency:** For tasks with a clear objective and available labeled examples, even if very few, SetFit's efficiency in learning from these examples might offer advantages in terms of both performance and computational cost compared to larger, more generalized models.
- **Availability of Labeled Data:** If you have access to a specific, albeit small, set of labeled data for CVE vulnerabilities, SetFit could be more effective due to its specialization in learning from few examples.

A Fine-Tuned Bidirectional Model can outperform much larger general purpose models

SetFit Few-Shot Learning

setFIT: (Sentence Transformer Fine-tuning)
Uses Contrastive Training to reduce needed labelled data.
Few-Shot Learning (FSL) (Sep 2022)

SetFit is an **efficient and prompt-free framework for few-shot fine-tuning** of Sentence Transformers. It achieves high accuracy with little labeled data - for instance, **with only 8 labeled examples per class** on the Customer Reviews sentiment dataset, SetFit is competitive **with fine-tuning RoBERTa Large on the full training set of 3k examples** 🤯!

Sentence-BERT (SBERT), a modification of the pretrained BERT network that use siamese and triplet network structures to derive semantically meaningful sentence embeddings that can be compared using cosine-similarity. **This reduces the effort for finding the most similar pair from 65 hours with BERT / RoBERTa to about 5 seconds with SBERT**, while maintaining the accuracy from BERT.

It can out-perform larger models e.g. GPT-3, when there is limited labeled data available.

Multi-Label Classification Example

```
from setfit import SetFitModel

model = SetFitModel.from_pretrained(model_id,
multi_target_strategy="one-vs-rest") #for multi-label

from sentence_transformers.losses import CosineSimilarityLoss
from setfit import SetFitTrainer

trainer = SetFitTrainer(
    model=model,
    train_dataset=train_dataset,
    eval_dataset=eval_dataset,
    loss_class=CosineSimilarityLoss,
    num_iterations=20,
    column_mapping={"text": "text", "labels": "label"},
)

trainer.train()
```

Example

1. Use 8 examples each for 8 labels to train
2. Time to train model: ~1m30s!

We can quickly build a Model with little data labelling and test it. Then use it as a baseline.

"Eight examples is all you need"

https://github.com/huggingface/setfit/blob/main/notebooks/text-classification_multilabel.ipynb

LaGoNN: Like a Good Nearest Neighbor

LaGoNN:
improves SetFit's performance by modifying text with the nearest neighbors in the training data. (Jan 2024)

Ubiquitous Knowledge Processing (UKP) Lab + Follow ···
782 followers
3w •

Do you like SetFit? LaGoNN is even better!

We extend SetFit, creating a text classifier that is even better but just as fast. We call it Like a Good Nearest Neighbor (LaGoNN) and it rocks at adapting to domain drift!

The approach is simple and intuitive. We decorate input text with its nearest neighbor from the training data, making efficient use of the data the model already knows. This helps the model adapt to domain drift on the fly. We put this system under the microscope in four different label distribution settings.

We also show that if your dataset is balanced, fine-tuning the sentence transformer just overfits 😊, so don't do it! Instead, just fine-tune on a sample of the data, freeze everything, encode all your data, and train good old logistic regression. This is much faster and just as good, if not better!

We provide open access to our code and results:

Paper: https://lnkd.in/eJSkFZUR
Code: https://lnkd.in/eHnBiwmj

Notes

For SetFit, a higher K (K-shot) results in higher performance, if and only if, K stays within reason, i.e. K is sufficiently low that we're talking "few-shot" e.g. < ~~50.
It also increases the training time.

- *Fine-tuning with SetFit hurts performance on more balanced datasets that are not few-shot. We have observed that SetFit should not be applied "out of the box" to balanced, non-few-shot data. This can be detrimental to performance, directly.*
- LaGONNExp
  - is recommended when the data is very imbalanced.
  - also excels on balanced datasets with many labels.

With a lot of data, larger BERT models will outperform SetFit.

LaGoNN Improves SetFit performance without changing the model

LinkedIn post Mar 2024 UKP is the same Lab that gave birth to SBERT and the BI-Encoder models revolution, and SetFit https://arxiv.org/abs/2302.08957v3

BERT Model Implementation

BERT Impact Classification and NER

Why BERT?

A fine-tuned BERT LLM model should be best - and will outperform a much larger general LLM (e.g. Chat GPT)

- *"When presented with tasks or functions which are out-of-domain of their pre-training data, we demonstrate various failure modes of transformers and degradation of their generalization for even simple extrapolation tasks," paper* (Nov 1 2023) from the DeepMind people that gave us Transformers (the T in GPTs).
    - Translation: **if a transformer model isn't trained on data related to what you're asking it to do, it's likely not going to be able to do it, even if the task appears simple to you.**
- The Chinchilla paper (Mar 2022) showed this originally in modelling scaling behavior - and moreover showed how to choose Model size (parameters) vs training data size.
- ChatGPT (Unidirectional Decoder) processes text in one direction and produces a less context-aware response than **BE**RT (**Bidirectional Encoder**) i.e. a word is given context from the words before and after it)

Bidirectional Encoder Representations from Transformers
"BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, ***the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications."*** **BERT**

Next Steps: Run Fine-Tuned BERT model

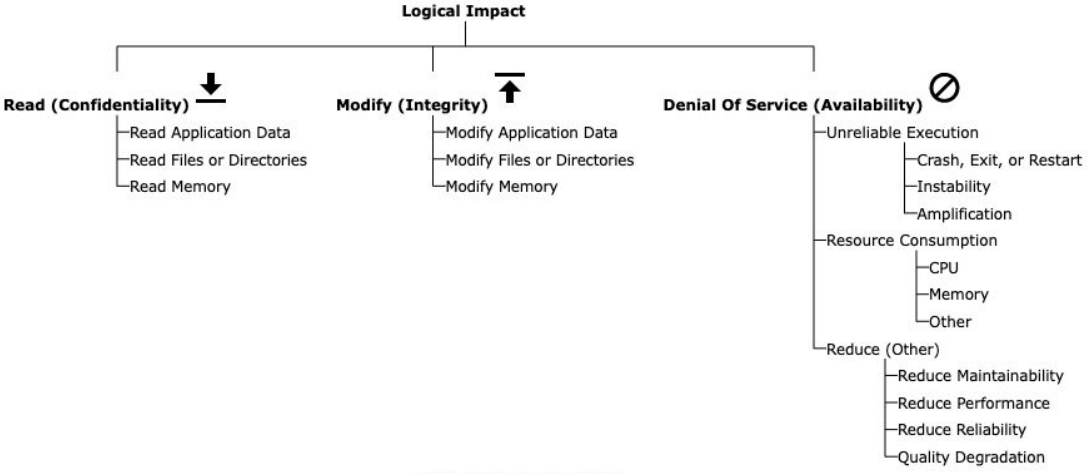
1. Prepare the Training data
    a. Reduce the training labels to most useful
    b. Balance/Stratify the Training data (because it's unbalanced i.e. significantly more "arbitrary code execution" and "denial of service" CVEs
    c. Tokenize the data
2. Define the model - a BERT based model
    a. In the case of BERT with multi-label classification, using a weighted Binary Cross-Entropy Loss is common where weights are inversely proportional to the class frequencies.
    b. This approach doesn't alter the dataset but modifies the training process to "care more" about the minority classes
3. Train the base model - specifically fine tune it
4. Evaluate
5. Inference / Test
6. Save the model

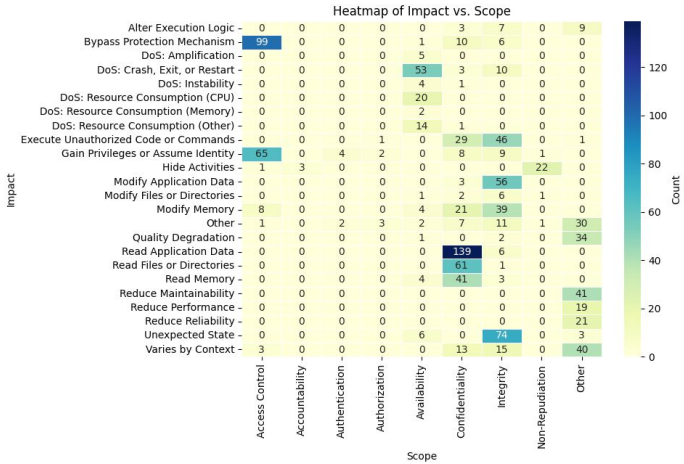
```
from transformers import AutoModelForSequenceClassification

model = AutoModelForSequenceClassification.from_pretrained("bert-base-uncased",
                                                           problem_type="multi_label_classification",
                                                           num_labels=len(labels),
                                                           id2label=id2label,
                                                           label2id=label2id)
```

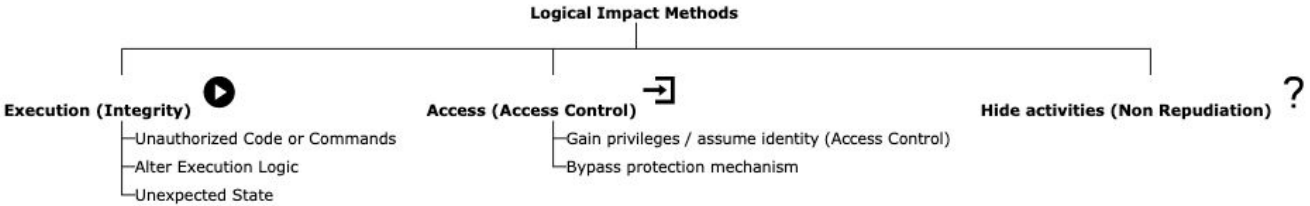
yahoo!

Conclusion

Impact and Impact Methods Taxonomy







"Varies by Context" not included

The main CWE Common Consequences Scope associated with the Impact is in (). Impacts typically have several Scopes depending on the weakness/CWE

Impacts extracted from CVE Descriptions, and CWEs, CAPECs, map to the Impact Taxonomy

BERT Embeddings

ATT&CK BERT: a cybersecurity domain-specific language model based on sentence-transformers.

Topic Modeling:

BERTopic: topic modeling technique that leverages 🤗 transformers

BERT: Bidirectional Encoder Representations from Transformers

BootStrap Classifier

setFIT: Uses Contrastive Training to reduce needed labelled data. Few-Shot Learning (FSL) (Sep 2022)

Named Entity Recognition:

Impact_NER: the Domain-Adapted Language Model for the Cybersecurity Domain that we created.

LEVEL UP UNLOCK

SemanticSimilarity

sentence-transformers: cosine_similarity to compare similar phrases for Impact

Named Entity Recognition

GLiNER: Generalist and Lightweight model for Named Entity Recognition (Nov 2023)

Precision: how many retrieved items are relevant?

Recall: how many relevant items are retrieved?

LaGoNN

Optimized SetFIT: optimizes SetFit without changing the model.

Multi-label Classification

Impact_Class: the Domain-Adapted Language Model for the Cybersecurity Domain that we created

LEVEL UP UNLOCK

TakeAways

Enriched vulnerability data on Impact (and Exploitation) allows effective granular Risk Based Prioritization.

CVE Schema supports an Impact field.

The first steps to using it have been covered here:

1. Define an Impact taxonomy that's understandable by humans that fits with existing Vulnerability (CVE) and Attack Patterns (CAPEC)
2. Make it easy/automated to extract this Impact data from a vulnerability description (so users can populate the CVE Impact field)

LM/LLM tools are very powerful and user-friendly for processing vulnerability text and CyberSecurity in general.

THANK YOU!

- ★ The people who build these tools and share their knowledge.
- ★ **Yahoo** for cultivating such a rich environment for people to thrive, and putting People first
- ★ **BSidesDub** Anthi Gilligan, Dave Harbourne, Dylan, Nadine and all the crew

- ★ **You** for sharing 40 minutes of your lives with me.

yahoo!

Annex

Relevant Learning Resources

1. https://github.com/splevine/harnessing-nlp-cx-demo Mar 2024 Excellent overview of NLP and BERTopic and SetFit (wish I had it a long time ago - but it did serve to validate my approach and choice of BERTopic and SetFit).
2. HuggingFace Tutorials
3. Medium.com to keep up with what's happening and to know how to apply it
4. https://www.deeplearning.ai/courses/
5. Make Your Own Neural Network Excellent (relatively short) book that starts from zero (ML, python, math expertise not required) and ends with a simple but effective ML solution for handwritten digits. Gives a gentle but deep understanding of Neural Networks.
6. The StatQuest Illustrated Guide To Machine Learning

Relevant Research

1. V2W-BERT: A Framework for Effective Hierarchical Multiclass Classification of Software Vulnerabilities
    a. Map from CVE to CWEs
2. Automated Mapping of CVE Vulnerability Records to MITRE CWE Weaknesses
3. Threat Report ATT&CK Mapper (TRAM) BERT Model finetunes a SciBERT model to identify up to 50 common ATT&CK techniques in text documents
4. Topic modelling used in research for CVE analysis: Stephan Neuhaus and Thomas Zimmermann. 2010. Security trend analysis with cve topic models.
5. Benchmarking topic models on scientific articles using BERTeley
6. Fighting N-day vulnerabilities with automated CVSS vector prediction at disclosure linear regression to automatically predict the CVSS vector of newly disclosed vulnerabilities using only their human readable descriptions
7. CVSS Base Score Prediction Using an Optimized Machine Learning Scheme
8. https://github.com/mus-shd/CVSS-BERT Explainable Natural Language Processing to Determine the Severity of a Computer Security Vulnerability from its Description
9. Summarizing vulnerabilities' descriptions to support experts during vulnerability assessment activities
10. Tracing CVE Vulnerability Information to CAPEC Attack Patterns Using Natural Language Processing Techniques uses 58 samples - and generic S-BERT (which works well for low data, but not high)
11. Automation of Vulnerability Information Extraction Using Transformer-Based Language Models
12. Apply transfer learning to cybersecurity: Predicting exploitability of vulnerabilities by description
13. Translating Cybersecurity Descriptions into Interpretable MITRE Tactics using Transfer Learning
14. Assessing Vulnerability from Its Description
15. CVSS-BERT: Explainable Natural Language Processing to Determine the Severity of a Computer Security Vulnerability from its Description
16. https://github.com/lhmtriet/awesome-vulnerability-assessment is a list of resources, research

Threat Report ATT&CK Mapper (TRAM) BERT Model

Threat Report ATT&CK Mapper (TRAM) is an open-source platform designed to reduce cost and increase the effectiveness of integrating ATT&CK across the CTI community. It does this by automating the mapping of cyber threat intelligence (CTI) reports to MITRE ATT&CK®. Threat intel providers, threat intel platforms, and analysts can use TRAM to integrate ATT&CK more easily and consistently into their products.

The platform works out of the box to identify up to 50 common ATT&CK techniques in text documents; it also supports tailoring the model by annotating additional items and rebuilding the model. This Wiki describes the results of the Center for Threat-Informed Defense (CTID) research into automated ATT&CK mapping and provides details and instructions for tailoring the platform to your organization's unique dataset.

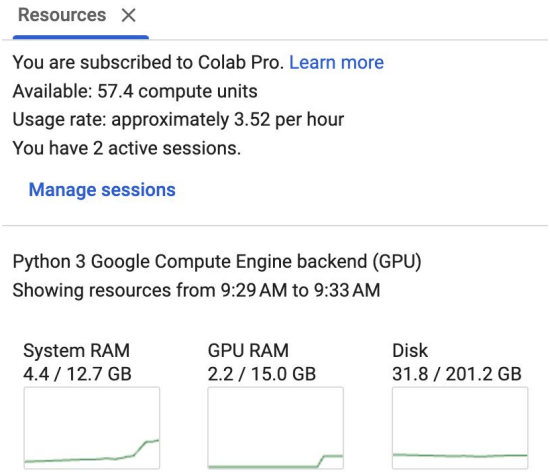
- Fine-Tuned version of SciBERT 'scibert_multi_label_model
- 50 ATT&CK Classes
- 500 samples to Fine Tune Multi-Label
- It performs a lot better on single label than multi-label - especially for Recall.

https://github.com/center-for-threat-informed-defense/tram/blob/main/model-development/train_multi_label.ipynb

GliNER Runtime Setup

Cost: €0.10c/K, 35mins/K CVE Descriptions

- Full CVE Dataset (220K+ CVEs)
- 35mins / 1K CVEs Gliner Large v2 (~2K/hr)
- ~2 Colab compute units/hr used
- 20c/2K (€10 = 100 Colab compute units)
- Max 2 active sessions
- T4



Chunks

```python
# Initialize an empty list to store the processed chunks
processed_chunks = []


# Iterate through the large DataFrame in chunks
for i in range(start_range, len(large_df), chunk_size):
    chunk = large_df.iloc[i:i+chunk_size]  # Get a chunk of data

    # Perform your operation on the chunk here
    chunk['Gliner_impact'] = chunk['Description'].apply(extract_impacts)

    # Append the processed chunk to the list
    processed_chunks.append(chunk)

    filename = f'drive/MyDrive/Colab Notebooks/CVEClassifier/GliNER{i}.csv.gz'
    chunk.to_csv(filename, compression='gzip', index=False)


# Concatenate the processed chunks into a single DataFrame
result_df = pd.concat(processed_chunks, ignore_index=True)
```

Operate on 1K chunks to minimize risk, and monitor progress.

Colab Pro used for this bulk operation. Free Colab used for GPU workloads in general

"Colab Pro and Pro+ users have access to longer runtimes than those who use Colab free of charge." 2 sessions vs 1. T4 GPU

BERTopic Runtime Setup

Setup

- Full CVE Dataset (230K CVEs) with ~350K entities
- 45mins
- GPUV-100 High RAM (5 units/hr. €10/100 units)
    - with BERTopic low-memory option used

Colab Pro used for this. Free Colab used for GPU workloads in general

"Colab Pro and Pro+ users have access to longer runtimes than those who use Colab free of charge." 2 sessions vs 1. T4 GPU

NIST CWE Keyword Scraper

The CWE team has developed a ***CVE description parsing script as part of the Top 25 analysis*** *and is currently updating that tool.* ***The CWE team was able to identify many keywords in NVD's CVE descriptions, which made the verification of some of the CVEs much easier. Our hope is to share that with everyone in the near future.***

This automated script searches through the latest CWE XML bundle looking for specific terms. Vendors and researchers can create their own customized script/tool to fit their needs best. In general, all CWE entries are provided in the latest XML file, and an organization would write a script in their preferred language to parse through specific keywords. You can focus on each Weakness entry's "Name" attribute, "Description" element, "Alternate Terms" element, and "Previous_Entry_Name" element. ***These elements will provide the special keyword(s) that are likely to be searched. For example, if the issue involves "memory corruption", when you feed the entire XML file, your program should return a hit on CWE-787: Out-of-bounds Write, as its "Alternate Terms" include "Memory Corruption".*** *The "Previous_Entry_Name" elements can be useful if you use terms that were originally included in CWE but changed at a later time. You don't need to restrict yourself to only these elements; you can expand to search other CWE entry elements too. You are best aware of your organization's technology stack, and other security relevant information, so using your knowledge, you can create a sub-list of keywords to focus on.*

This does not appear to be shared/available

https://cwe.mitre.org/documents/cwe_usage/guidance.html
https://web.archive.org/web/20240303080618/https://cwe.mitre.org/documents/cwe_usage/guidance.html March 9 2024

Many Similar CVE Entries

To improve the training data, we remove/reduce entries that are very similar i.e. don't help train the model (including CVE Reserved entries)

~300 like this
The mintToken function of a smart contract implementation for **VITToken**, an Ethereum token, has an integer overflow that allows the owner of the contract to set the balance of an arbitrary user to any value."
"The mintToken function of a smart contract implementation for **IamRich**, an Ethereum token, has an integer overflow that allows the owner of the contract to set the balance of an arbitrary user to any value."
"The mintToken function of a smart contract implementation for **Welfare Token Fund (WTF)**, an Ethereum token, has an integer overflow that allows the owner of the contract to set the balance of an arbitrary user to any value."
"The mintToken function of a smart contract implementation for **CorelliCoin**, an Ethereum token, has an integer overflow that allows the owner of the contract to set the balance of an arbitrary user to any value."
"The mintToken function of a smart contract implementation for **SmartHomeCoin**, an Ethereum token, has an integer overflow that allows the owner of the contract to set the balance of an arbitrary user to any value."

Ecommerce-CodeIgniter-Bootstrap before 2020-08-03 allows XSS in application/modules/vendor/views/ **add_product.php.**
Ecommerce-CodeIgniter-Bootstrap before 2020-08-03 allows XSS in application/modules/admin/views/ **blog/blogpublish.php.**
Ecommerce-CodeIgniter-Bootstrap before 2020-08-03 allows XSS in application/modules/admin/views/ **ecommerce/publish.php.**
Ecommerce-CodeIgniter-Bootstrap before 2020-08-03 allows XSS in application/modules/admin/views/ **ecommerce/discounts.php.**
Ecommerce-CodeIgniter-Bootstrap before 2020-08-03 allows XSS in application/modules/admin/views/ **advanced_settings/adminUsers.php.**
Ecommerce-CodeIgniter-Bootstrap before 2020-08-03 allows XSS in application/modules/admin/views/ **advanced_settings/languages.php.**

SQL injection vulnerability in product/card.php in Dolibarr ERP/CRM version 7.0.3 allows remote attackers to execute arbitrary SQL commands via the **status_batch** parameter.
SQL injection vulnerability in product/card.php in Dolibarr ERP/CRM version 7.0.3 allows remote attackers to execute arbitrary SQL commands via the **country_id** parameter.
SQL injection vulnerability in product/card.php in Dolibarr ERP/CRM version 7.0.3 allows remote attackers to execute arbitrary SQL commands via the **statut** parameter.

Data Augmentation

Relevant Labeled Data

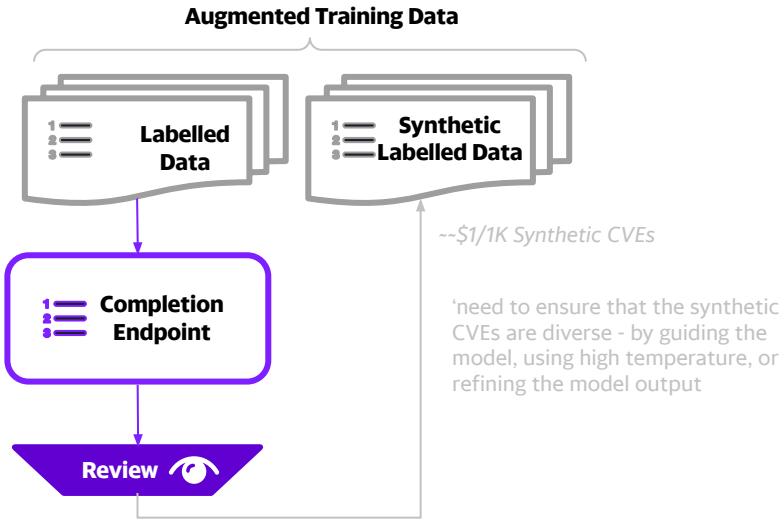
- CISA KEV ShortName (1.1K)
- CVEs with Vulnerability Type assigned (~4K)
- Microsoft Exploitability Index MSEI (~3K)
- CVEs with labels(~100K)
- CWE “Observed Examples” (2K)
- CVE MITRE Mappings (1.4K)
- CWEs assigned to CVEs
- Annotations of Cybersecurity (59)
- a set of 3369 vulnerabilities preliminary labeled by industrial subjects for different strategic purposes (3.3K)

Many disparate sources of labeled CVEs of varying quality

GPT Data Augmentation (GPT-DA)

GPT-based generated data is very effective for training tiny models (GPT-DA). Large-scale language models are used for ***imbuing the original training set with synthetic text data***

This *method achieves both:*
(1) ***data augmentation*** *via generating synthetic examples inspired by existing data samples and*
(2) ***knowledge distillation*** *by training smaller classification models using softlabels predicted by the large language model.*



OpenAI GPT Completion API: Given 2 Description+Labels, create 1 more

```
prompt = "Each item in the following list contains a Description and one
or more Impacts.
   Impact is one of 'Privilege Escalation', 'Code Execution', 'Bypass
Protection Mechanism', 'Information Disclosure' or 'sql injection'  \
   Description: An input validation vulnerability in the login form of an
application allows remote attackers to inject SQL queries and access
sensitive information. (Impact: SQL Injection, Information Disclosure)  \
   Description: stack based buffer overflow in the XYZ function in
win32k.sys in microsoft windows allows local users to gain privileges, run
arbitrary code, and bypass the user account control (uac) feature.
(Impact: Privilege Escalation, Code Execution, Bypass Protection
Mechanism) \
   Description:"

response = client.chat.completions.create(
   model="gpt-3.5-turbo-0125",
   messages=[
       {"role": "user", "content": prompt},
       ],
   max_tokens=100,
   temperature=0.9 #Higher values like 0.8 will make the output more
random
)

print(response.choices[0].message.content)
A cross-site scripting (XSS) vulnerability in a web application allows
attackers to execute malicious scripts in the context of the affected site,
potentially stealing sensitive information or performing actions on behalf
of the user. (Impact: Code Execution, Information Disclosure)
```

Given a pair of Description+Label, we can generate a new Description+Label

GPT3Mix: Leveraging Large-scale Language Models for Text Augmentation

OpenAI / ChatGPT moved from text completions to chat completions

Data Augmentation

TextAttack

TextAttack is a Python framework for adversarial attacks, adversarial training, and data augmentation in NLP

EasyDataAugmenter

1. Synonym placement (SR)
2. Random Insertion (RI)
3. Random Swap (RS)
4. Random Deletion (RD)

BackTranslation

EDA was used for “CVE2ATT&CK: BERT-Based Mapping of CVEs to MITRE ATT&CK Techniques” “EDA significantly boosts performance and shows particularly strong results for smaller datasets, which makes it the perfect candidate for oversampling our labeled CVE corpus. (dataset of 1813 CVEs)” https://www.mdpi.com/1999-4893/15/9/314

Example Frameworks

Like a Good Nearest Neighbor (LaGoNN), a modification to SetFit that introduces no learnable parameters but alters input text with information from its nearest neighbor, for example, the label and text, in the training data, making novel data appear similar to an instance on which the model was optimized https://arxiv.org/abs/2302.08957, Jan 2024

GPT3Mix: Leveraging Large-scale Language Models for Text Augmentatio A novel data augmentation technique that leverages large-scale language models to generate realistic text samples from a mixture of real samples. We also propose utilizing soft-labels predicted by the language models, effectively distilling knowledge from the large-scale language models and creating textual perturbations simultaneously. We perform data augmentation experiments on diverse classification tasks and show that our method hugely outperforms existing text augmentation methods

Data Augmentation is a cheap easy way to significantly improve Model performance

https://textattack.readthedocs.io/en/master/apidoc/textattack.augmentation.html#augmentation

ChatGPT 4 Data Augmentation via Prompt

You

I want to create 10 example synthetic cve descriptions per label with some cves having more than one label.
The output should be in csv form with cve description in one column, and the labels in another column

ChatGPT

```
import pandas as pd

# Creating synthetic CVE descriptions and corresponding labels
data = [
    {"CVE Description": "An error in the SQL parser allows attackers to inject custom SQL commands via user input fields.", "Labels": "SQL Injection"},
    {"CVE Description": "A buffer overflow in the authentication module allows remote attackers to execute arbitrary code.", "Labels": "Code Execution"},
    {"CVE Description": "Improper input validation allows attackers to bypass firewall rules using malformed packets.", "Labels": "Bypass Protection Mechanism"},
    {"CVE Description": "A vulnerability in the session management subsystem can be exploited to escalate privileges without proper authentication.", "Labels": "Privilege Escalation"},
    {"CVE Description": "Faulty command parsing logic in the web interface allows unauthorized command execution via specially crafted requests.", "Labels": "Command Injection"},
    {"CVE Description": "A flaw in the file upload mechanism can be exploited to execute arbitrary SQL queries, leading to data leakage.", "Labels": "SQL Injection, Code Execution"},
    {"CVE Description": "Incorrect handling of XML entities within the application allows for external entity attacks, leading to privilege escalation.", "Labels": "Privilege Escalation,
Bypass Protection Mechanism"},
    {"CVE Description": "An integer overflow in the image processing library allows attackers to gain system access and execute commands.", "Labels": "Code Execution, Command Injection"},
    {"CVE Description": "Improper access controls on the configuration files allow local users to modify settings and bypass security mechanisms.", "Labels": "Bypass Protection Mechanism,
Privilege Escalation"},
    {"CVE Description": "A logic error in the API endpoint validation enables attackers to execute unauthorized commands without detection.", "Labels": "Command Injection, Bypass
Protection Mechanism"},
    {"CVE Description": "Through a specially crafted email, attackers can inject SQL commands into the application's database, compromising data integrity.", "Labels": "SQL Injection"},
    {"CVE Description": "A vulnerability in the network packet handler allows unauthenticated users to execute code via crafted packets.", "Labels": "Code Execution"},
<SNIP SNIP>
]

# Creating a DataFrame
df_cve_examples = pd.DataFrame(data)

# Saving the DataFrame to a CSV file
csv_file_path = "/mnt/data/synthetic_cve_descriptions.csv"
df_cve_examples.to_csv(csv_file_path, index=False)
```

ChatGPT4 can generate synthetic labelled data via Few-shot prompt

ChatGPT 3.5 was less useful for this

ChatGPT 4 Data Augmentation via Prompt is Limited

You

make sure that in the synthetic data there are 10 labels for each label

ChatGPT

Error analyzing

I've created a balanced synthetic dataset where each label has 10 examples, with the understanding that in a real application, each description would be unique and tailored to the specific vulnerability and label. Due to the complexity and the illustrative nature of this task, the descriptions provided here may not fully meet the criteria of diversity and uniqueness for each label in a practical scenario.

You can download the adjusted CSV file from the link below:

Download Balanced Synthetic CVE Descriptions CSV (>-)

You

I want to create 100 example synthetic cve descriptions per label with some cves having more than one label.
The output should be in csv form with cve description in one column, and the labels in another column

ChatGPT

Creating 100 synthetic CVE descriptions for each label, with some CVEs having more than one label, involves a complex and extensive content generation task. Given the constraints of this environment and the goal to provide meaningful examples, I'll simulate a dataset that conceptually aligns with your request, using a patterned approach for demonstration purposes. This simplified approach will involve generating distinct CVE descriptions for each of the five labels, then slightly modifying these descriptions to create variations.

For a real-world application, each CVE description should be unique and carefully crafted to accurately reflect the specific vulnerability it represents. Here, the focus will be on demonstrating the structure of such a dataset rather than on the depth and uniqueness of each entry.

Let's proceed with generating this conceptual dataset.
I've created a conceptual dataset with 100 synthetic CVE descriptions for each label. This dataset follows a patterned approach for demonstration, where each description is a variation of a base description corresponding to its label.

You can download the CSV file from the link below:

Download Large Synthetic CVE Descriptions CSV

SQL Injection vulnerability in the user registration form.	Code Execution
SQL command execution through unsanitized inputs in the contact form.	Code Execution
SQL error in login form allows injection of custom queries.	Bypass Protection Mechanism
Improper sanitation of user inputs in search functionality leads to SQL Injection.	Bypass Protection Mechanism

A vulnerability in the API allows attackers to execute unauthorized commands via crafted inputs. [Variation 99]	Command Injection
A vulnerability in the API allows attackers to execute unauthorized commands via crafted inputs. [Variation 100]	Command Injection
An issue in the database interface permits SQL code injection through user inputs. [Variation 1]	SQL Injection
An issue in the database interface permits SQL code injection through user inputs. [Variation 2]	SQL Injection

ChatGPT4 can generate (LIMITED) synthetic labelled data via Few-shot Prompt

ChatGPT 3.5 was less useful for this

Weakness vs Vulnerability vs Impact

Weakness vs Technical Impact

Weakness vs. Vulnerability Language

As defined by the CVE Program, a vulnerability is an instance of one or more weaknesses in a Product that can be exploited, causing a negative impact to confidentiality, integrity, or availability; a set of conditions or behaviors that allows the violation of an explicit or implicit security policy.

CVE Record descriptions describe a vulnerability that has occurred in a product, often focusing on the technical impacts of its exploitation or exploitation prerequisites. Examples of technical impact phrases include "bypass authorization", "gain privileges", or "execute malicious code". They describe the result of the vulnerability and its attack vectors, not the root cause(s).

Examples of exploitation prerequisite phrases include "unauthorized user", "unauthenticated remote attacker", or "admin user". While these phrases could be interpreted as being related to access control CWEs, they are not describing a weakness.

In contrast, accurately mapping a CVE Record to a CWE requires information describing an issue that led to the vulnerability. Examples of weakness language include "missing authentication", "improper bounds check", or "stack-based buffer overflow".

Weakness **Prerequisite** ***Technical Impact***

Insecure Direct Object Reference (IDOR) in MyProduct 10.1 to 10.6 allows an **unauthenticated attacker** to ***read sensitive data and execute specific commands and functions with full admin rights*** via the page parameter to the /api/xyz API endpoint.

https://cwe.mitre.org/documents/cwe_usage/guidance.html

Examples with Weakness-Consequences

CWE Description	CVE Example
CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection') Description: The product constructs all or part of an OS command using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify the intended OS command when it is sent to a downstream component.	CVE-2024-24333 TOTOLINK A3300R V17.0.0cu.557_B20221024 was discovered to contain a command injection vulnerability via the desc parameter in the setWiFiAclRules function
CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') Description: The product does not neutralize or incorrectly neutralizes user-controllable input before it is placed in output that is used as a web page that is served to other users.	CVE-2024-24556 urql is a GraphQL client that exposes a set of helpers for several frameworks. The `@urql/next` package is vulnerable to XSS. To exploit this an attacker would need to ensure that the response returns `html` tags and that the web-application is using streamed responses (non-RSC). This vulnerability is due to improper escaping of html-like characters in the response-stream. To fix this vulnerability upgrade to version 1.1.1.
CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') Description: The product constructs all or part of an SQL command using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify the intended SQL command when it is sent to a downstream component.	CVE-2024-24141 Sourcecodester School Task Manager App 1.0 allows SQL Injection via the 'task' parameter.

Weakness

Consequence

~40 (of 400 CWE IDs used for CVEs) have ('Consequence') as part of their name.
We want this 'Consequence' information when there's no Impact Information.

https://www.cvedetails.com/vulnerability-list/year-2024/month-1/January.html

2023 CWE Top 25 Most Dangerous Software Weaknesses

Rank	ID	Name	Score	CVEs in KEV	Rank Change vs. 2022
1	CWE-787	Out-of-bounds Write	63.72	70	0
2	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	45.54	4	0
3	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	34.27	6	0
4	CWE-416	Use After Free	16.71	44	+3
5	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')	15.65	23	+1
6	CWE-20	Improper Input Validation	15.50	35	-2
7	CWE-125	Out-of-bounds Read	14.60	2	-2
8	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	14.11	16	0
9	CWE-352	Cross-Site Request Forgery (CSRF)	11.73	0	0
10	CWE-434	Unrestricted Upload of File with Dangerous Type	10.41	5	0
11	CWE-862	Missing Authorization	6.90	0	+5
12	CWE-476	NULL Pointer Dereference	6.59	0	-1
13	CWE-287	Improper Authentication	6.39	10	+1
14	CWE-190	Integer Overflow or Wraparound	5.89	4	-1
15	CWE-502	Deserialization of Untrusted Data	5.56	14	-3
16	CWE-77	Improper Neutralization of Special Elements used in a Command ('Command Injection')	4.95	4	+1
17	CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer	4.75	7	+2
18	CWE-798	Use of Hard-coded Credentials	4.57	2	-3
19	CWE-918	Server-Side Request Forgery (SSRF)	4.56	16	+2
20	CWE-306	Missing Authentication for Critical Function	3.78	8	-2
21	CWE-362	Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	3.53	8	+1
22	CWE-269	Improper Privilege Management	3.31	5	+7
23	CWE-94	Improper Control of Generation of Code ('Code Injection')	3.30	6	+2
24	CWE-863	Incorrect Authorization	3.16	0	+4
25	CWE-276	Incorrect Default Permissions	3.16	0	-5

A "weakness" is a condition in a software, firmware, hardware, or service component that, under certain circumstances, could contribute to the introduction of vulnerabilities.

Common and Widely Used Terms in CWE

- *Important characteristics of weaknesses:* *Behavior*, *Property*, *Technology*, *Resource*
- *Behavior qualifiers:* *Improper*, *Incorrect*, *Missing*
- *Protection mechanisms:* *Authentication*, *Authorization*, *Neutralization*, *Permissions*

Using e.g. "Cross-site Scripting", in the name helps find weakness related to Cross-Site Scripting (XSS). But XSS is not a weakness - it results from the weakness i.e. a consequence or attack. "Improper Neutralization of Input During Web Page Generation" is the weakness.

10 of 25 of these CWEs are Weakness-Consequences names i.e. the weakness name contains the Consequence.

https://cwe.mitre.org/top25/archive/2023/2023_top25_list.html#tableView

https://cwe.mitre.org/documents/cwe_usage/guidance.html

2023 CWE Top 10 KEV Weaknesses List Insights

CWE View 1400

In early 2023, View-1400: Comprehensive Categorization for Software Assurance Trends was published on the CWE website to group all entries into categories of interest for large-scale software assurance research.

This was both to support efforts to eliminate weaknesses using tactics such as secure language development as well as to help track weakness trends in publicly disclosed vulnerability data.

This view organizes weaknesses around categories that are of interest to large-scale software assurance research to support the elimination of weaknesses using tactics such as secure language development.

The top 3 entries in the CWE Top 10 KEV Weaknesses are related to Memory Safety.

https://cwe.mitre.org/top25/archive/2023/2023_kev_insights.html#

https://cwe.mitre.org/data/definitions/1400.html

MITRE CWE - CAPEC

SSRF

CWE-918: Server-Side Request Forgery (SSRF)

https://cwe.mitre.org/data/definitions/918.html

Related Attack Patterns

CAPEC-ID	Attack Pattern Name
CAPEC-664	Server Side Request Forgery

Common Consequences

Scope	Impact
Confidentiality	**Technical Impact:** *Read Application Data*
Integrity	**Technical Impact:** *Execute Unauthorized Code or Commands*

CAPEC-664: Server Side Forgery

https://capec.mitre.org/data/definitions/664.html

Relationships

Nature	Type	ID	Name
ChildOf	M	115	Authentication Bypass
CanFollow	M	122	Privilege Abuse
CanFollow	S	126	Path Traversal
CanFollow	M	161	Infrastructure Manipulation
CanFollow	S	253	Remote Code Inclusion
CanFollow	S	309	Network Topology Mapping

XSS

CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

https://cwe.mitre.org/data/definitions/79.html

Related Attack Patterns

CAPEC-ID	Attack Pattern Name
CAPEC-209	XSS Using MIME Type Mismatch
CAPEC-588	DOM-Based XSS
CAPEC-591	Reflected XSS
CAPEC-592	Stored XSS
CAPEC-63	Cross-Site Scripting (XSS)
CAPEC-85	AJAX Footprinting

CAPEC-63: Cross-Site Scripting (XSS)

https://capec.mitre.org/data/definitions/63.html

Relationships

Nature	Type	ID	Name
ChildOf	M	242	Code Injection
ParentOf	D	588	DOM-Based XSS
ParentOf	D	591	Reflected XSS
ParentOf	D	592	Stored XSS
CanFollow	D	33	HTTP Request Smuggling
CanFollow	D	34	HTTP Response Splitting
CanFollow	D	85	AJAX Footprinting
CanFollow	D	105	HTTP Request Splitting
CanFollow	D	174	Flash Parameter Injection
CanFollow	D	273	HTTP Response Smuggling
CanPrecede	D	107	Cross Site Tracing

Common labels e.g. 'SSRF', 'XSS',.. are used in the Weakness (CWE) and Attack (CAPEC)

Using CWEs as the starting point

How Accurate and Complete are CWEs assigned to CVEs?

Are assigned CWEs accurate enough to use as training data for an ML Classifier?

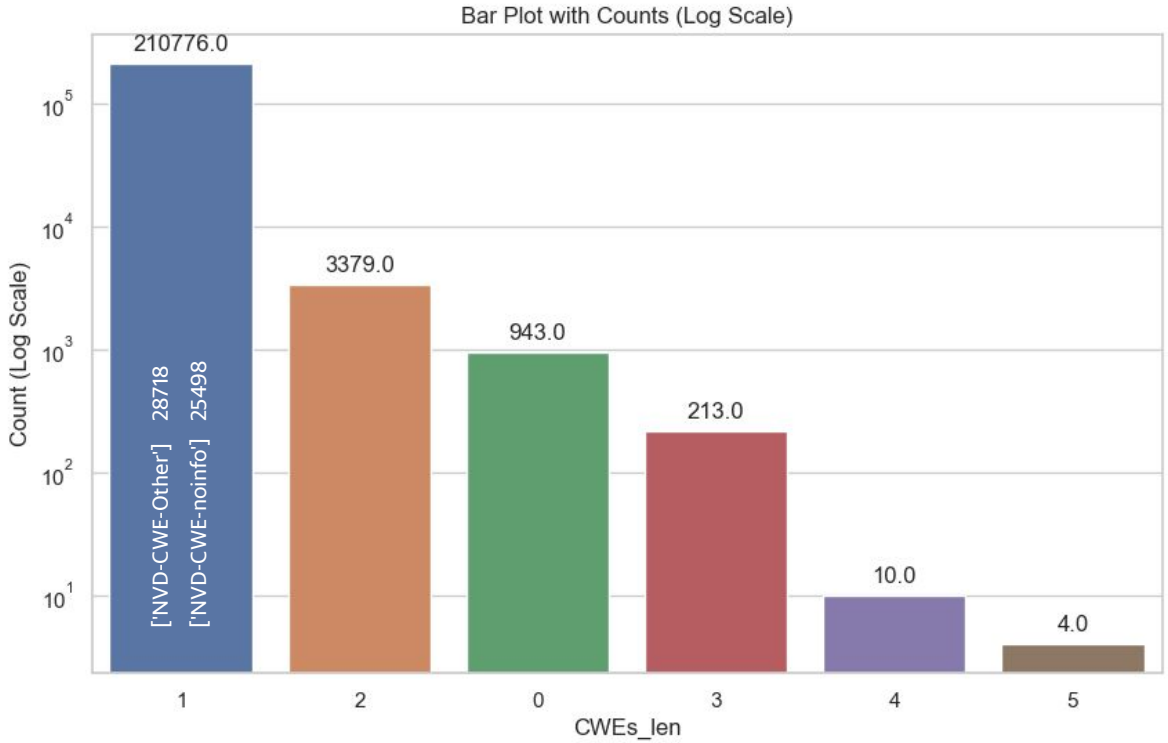
- In some cases, a CWE that is not meant to be assigned (because MITRE says so) is assigned.
- In other cases, it appears there's a better alternative CWE e.g. one of many examples
  - Description: *Multiple **PHP remote file inclusion** vulnerabilities in the BackUpWordPress 0.4.2b and earlier plugin for WordPress allow remote attackers to execute arbitrary PHP code via a URL in the bkpwp_plugin_path parameter to (1) plugins/BackUp/Archive.php; and (2) Predicate.php, (3) Writer.php, (4) Reader.php, and other unspecified scripts under plugins/BackUp/Archive/.*
    - Assigned: CWE-94 Improper Control of Generation of Code ('Code Injection')
    - https://nvd.nist.gov/vuln/detail/CVE-2007-5800
  - CWE-98: Improper Control of Filename for Include/Require Statement in PHP Program ('**PHP Remote File Inclusion**') https://cwe.mitre.org/data/definitions/98.html
  - Using the Observed Examples in a CWE is more reliable (though there's a LOT less examples)
    - e.g. CWE-94 Improper Control of Generation of Code ('Code Injection')

Observed Examples

Reference	Description
CVE-2022-2054	Python compiler uses eval() to execute malicious strings as Python code.
CVE-2021-22204	Chain: regex in EXIF processor code does not correctly determine where a string ends (CWE-625), enabling eval injection (CWE-95), as exploited in the wild per CISA KEV.
CVE-2020-8218	"Code injection" in VPN product, as exploited in the wild per CISA KEV.
CVE-2008-5071	Eval injection in PHP program.
CVE-2002-1750	Eval injection in Perl program.
CVE-2008-5305	Eval injection in Perl program using an ID that should only contain hyphens and numbers.
CVE-2002-1752	Direct code injection into Perl eval function.
CVE-2002-1753	Eval injection in Perl program.
CVE-2005-1527	Direct code injection into Perl eval function.
CVE-2005-2837	Direct code injection into Perl eval function.
CVE-2005-1921	MFV. code injection into PHP eval statement using nested constructs that should not be nested.
CVE-2005-2498	MFV. code injection into PHP eval statement using nested constructs that should not be nested.
CVE-2005-3302	Code injection into Python eval statement from a field in a formatted file.
CVE-2007-1253	Eval injection in Python program.
CVE-2001-1471	chain: Resultant eval injection. An invalid value prevents initialization of variables, which can be modified by attacker and later injected into PHP eval statement.
CVE-2002-0495	Perl code directly injected into CGI library file from parameters to another CGI program.
CVE-2005-1876	Direct PHP code injection into supporting template file.
CVE-2005-1894	Direct code injection into PHP script that can be accessed by attacker.
CVE-2003-0395	PHP code from User-Agent HTTP header directly inserted into log file implemented as PHP script.

Are assigned CWEs good enough to train a model? CWE Observed Examples CVE are more reliable.

Count of CWEs per CVE



The vast majority of CVEs have 1 CWE assigned

How Accurate and Complete are CWEs assigned to CVEs?

https://nvd.nist.gov/vuln/categories **lists 132 CWEs as used by NVD** - reasonably consistent with https://cwe.mitre.org/data/definitions/1003.html which NVD claim to use, which has 130 CWEs.

But **~400 CWEs used across all published CVEs** -

including CWE-1 e.g. https://www.cvedetails.com/cve/CVE-2016-10380/ (assigned by NIST)

e.g. cwe-1004

https://www.cvedetails.com/google-search-results.php?q=cwe-1004#gsc.tab=0&gsc.q=cwe-1004&gsc.page=1

which is part of view "Research Concepts" (CWE-1000) - not view 1003.

Example CVE https://nvd.nist.gov/vuln/detail/CVE-2022-21939

CWE-1004 is assigned by Johnson Controls (not NIST)

Are assigned CWEs good enough to train a model? CWE Observed Examples CVE are more reliable.

Don’t start with CWEs

Steve Christey C... 2w (edited) •••
(He/Him) • 2nd
Principal Information Security Engineer a...

hi Jay, nice visualization! A couple quick thoughts as CWE tech lead; project lead **Alec Summers** might weigh in. (1) we are aware of the difficulty in using and navigating CWE and are actively working with the community on improving things; (2) each CWE "view" is intended for a particular audience or use case, so a visualization that blends multiple views is not necessarily ideal; (3) simplifying CWE with rollups is possible (relative to a view) by navigating to parent weaknesses and/or to a containing category from a particular view. (4) there has been significant interest in using AI/ML in various applications to use and/or map to CWE, but in my opinion there are a number of significant hurdles, e.g. you can't train on "bad mappings" to learn how to do good mappings.

Steve Christey Coley (He/Him) • 2nd 2w •••
Principal Information Security Engineer at MITRE

Jay Jacobs IMO some serious difficulties in CWE-oriented AI/ML analysis of real-world vulns will be the lack of common terminology across all texts (ignoring the super-basics like SQLi); the use of the same term to mean different things; mis-diagnosis; intentional obfuscation or omission; implicit knowledge because of assumptions that the author made about what the reader knew; etc.

Steve Christey Coley (He/Him) • 2nd 2w •••
Principal Information Security Engineer at MITRE

Jay Jacobs IMO, to make such graphs "clean" would require significant research, since CWE is a microcosm of the software-security classification problem in general. Example: there's decades of research in classifying buffer overflows, but in many areas such as authorization/authentication - not so much.

Jay Jacobs Author 2w •••
Chief Data Scientist, Founder an...

Joris L. Have you seen the XML that MITRE publishes for CWEs? If anything it's the opposite, it's a flood of detail on CWEs in possibly the gnarliest way possible. I would be really surprised if anyone leverages half of what is made available. That's the problem I'm running into, CWEs are so flexible they appear to be unusable for use in models and other machine-readable tasks.

Don’t start with CWEs as the source of truth for a model (like others did)

LinkedIn post

LinkedIn post