M2 Project Deliverable 2

Data Processing:

Common software weaknesses in CISAs Known Exploitable Catalog

**Project Background:**

In my project, I analyze data from three major sources in the cybersecurity field: the Cybersecurity and Infrastructure Security Agency (CISA), the National Institute of Standards and Technology (NIST), and the Massachusetts Institute of Technology Research and Engineering (MITRE). To facilitate a comprehensive understanding, I will frequently refer to specific cybersecurity terms including Common Vulnerabilities and Exposures (CVE), Common Weakness Enumeration (CWE), and Common Vulnerability Scoring System (CVSS). These terms are essential for my analysis and are defined as follows:

* CVEs are managed by NIST and provide a public method of sharing information about specific cybersecurity vulnerabilities.
* The CWEs database is managed by MITRE and is a category system for types of hardware and software weaknesses and vulnerabilities.
* CVSS is a way to evaluate and rank reported vulnerabilities in a standardized and repeatable way, as determined by the Forum of Incident Response and Security Teams (FIRST).

The primary aim of my project is to explore how CISA's catalog of known exploited vulnerabilities correlates with the information published in NIST and MITRE databases. This catalog, a crucial collection of vulnerabilities frequently targeted by malicious cyber actors, holds significant importance for federal enterprises. Under operational directives, Federal Civilian Executive Branch agencies (FCEB) are mandated to address vulnerabilities listed in this catalog by prescribed due dates (Cybersecurity & Infrastructure Security Agency). To navigate and connect these disparate data sources, I employed several Python3 scripts, which will be discussed in the subsequent sections. Understanding the relationship between these databases is vital, as it could shed light on patterns and trends in cybersecurity threats, potentially guiding future defense strategies and policy-making efforts in the field.

**Data Processing:**

My data preparation process primarily involved Python scripts, Linux terminal commands, and Excel formulas. This approach, while unconventional for a typical data analyst, aligns with my cyber security engineering background. I developed three Python scripts (detailed in Appendix A) to extract and correlate data from the NIST and MITRE databases with CVEs in CISA's Known Exploitable Catalog. Specifically:

* *cwe-searcher.py*: This script correlated MITRE's CWEs with the term 'phishing'.
* *nist\_cvss2.py*: It matched CVEs with NIST's CVSSv2 and CVSSv3.1 base scores.
* *CVE\_CWE\_Matcher\_7.py*: This script correlated CVEs with CWEs based on NIST's reports.

These scripts, executed on a CentOS 9 Linux 5.14 server, iteratively evolved and improved, as indicated by their names, until the desired data mapping was achieved.

**Handling Negative timeToFix Values:**

In processing the data, I encountered negative values in the 'timeToFix' column. These were linked to emergency directives (ED) requiring prompt action, leading to retroactive CVE additions in CISA's catalog. For instance, CVE-2021-34527, added in November 2021 but mandated for fix in July 2021 via ED-21-04, exemplifies this (Cybersecurity & Infrastructure Security Agency. (n.d.). Emergency directive 21-04). Given this context, I manually adjusted the 'timeToFix' column to reflect the duration between NIST's CVE publication and CISA's mandated fix date, applying this correction to 15 similar CVEs. However, three records, related to Pulse Connect Secure, retained negative values post-correction, likely due to preemptive vendor notifications to CISA. These were excluded from the timeToFix-related analysis for accuracy.

**CVSS Score Normalization:**

Further data review revealed inconsistencies in CVSS base scores, with some using CVSSv2 and others CVSSv3.1. Lacking a standard method for direct conversion, I opted to translate CVSSv2 base scores into CVSSv3.1 severity categories. This decision stemmed from the distinct methodologies of the two versions, where CVSSv3.1 offers more nuanced scoring based on specific organizational environments. Using a script (*CVSS\_V\_Mapping.py* in Appendix B), I mapped these scores, ensuring a consistent severity rating framework across all data points.

**Challenges with Data Integration:**

The integration of the script results into my Excel spreadsheet proved unexpectedly challenging. Despite matching the number of script outputs with the spreadsheet rows, the data failed to paste correctly. This inconsistency, likely due to sorting discrepancies in Excel, necessitated manual matching of the script outputs to 165 rows containing CVSSv2 base scores. This labor-intensive process, though manageable given the data volume, significantly extended my processing time.

**CWE Category Analysis:**

A notable observation during the data analysis was the prevalence of 'none' as a category in the CWE column, hinting at incomplete or outdated mappings by NIST. To address this, I developed script\_for\_cwe\_manual\_review.py (Appendix C), targeting rows with 'none' in the CWE column. This script facilitated quick identification of applicable CWEs for each CVE, utilizing the 'vulnerability\_name' and 'shortDescription' fields. Although straightforward in some cases, like mapping "Privilege Escalation" to CWE-269, other instances required a more nuanced approach. Drawing on my cyber security engineering experience, I undertook the task of manually categorizing CVEs when detailed information was lacking from NIST or vendor descriptions. While my categorizations are best-effort approximations, they significantly reduced the number of 'none' categorizations from 90 to 13, enhancing the overall data quality and relevance.

Upon completion of the initial data categorization, I discovered a significant oversight. I had inadvertently used a prohibited CWE ID, CWE-264, in my manual CVE assessment. Since 2019, this ID has been deprecated due to its overlap with more specific CWEs. Fortuitously, all instances involving CWE-264 pertained to some form of privilege escalation. After replacing each occurrence of CWE-264 with CWE-268 (Improper Privilege Management), I re-executed all analyses and updated the related visualizations.

**Data Analysis:**

After collating all necessary data, I began analyzing the results. I used a Python script called *analyze\_crosstab\_CWE5.py* (referenced in Appendix D), I extracted the most prevalent CWEs, leading vendors, and their associations with specific CWEs and products. The following charts, Figures 2 and 3, illustrate the distribution and ranking of CWEs in the CISA catalog.

A graph with numbers and lines

Description automatically generated

Figure 1: CWE Distribution in the Enhanced CISA Catalog, Nov 2023

A graph of a number of items

Description automatically generated with medium confidence

Figure 2: Top Ten CWEs in the Enhanced CISA Catalog, Nov 2023

My analysis confirmed the hypothesis: certain CWEs are indeed more prevalent in CISA's known exploitable catalog. Notably, Improper Privilege Management, Input Validation, and Memory Buffer Overflow emerged as the top three software weaknesses leading to CVEs in the catalog. This finding aligns with MITRE’s “Stubborn Weaknesses in the CWE Top 25” list for 2023, which annually tracks the most significant CWEs in published CVEs (MITRE. (2023). 2023 Stubborn Weaknesses).

A chart with colorful lines and numbers

Description automatically generated with medium confidence

Figure 3: Comparison of Top 10 CWEs in CISA’s Catalog with MITRE’s Top 25, Nov 2023

While the top CWEs in my enriched catalog don’t entirely mirror MITRE’s Top 25, similarities in trends are evident. Both lists predominantly feature weaknesses related to improper data processing (like Input Validation) and inadequate memory management (such as Buffer Overflows). Interestingly, three CWEs—CWE-269 (Improper Privilege Management), CWE-94 (Code Injection), and CWE-77 (Command Injection)—rank highly in the CISA catalog but are absent from MITRE’s list. These can be broadly categorized under improper data processing and insufficient privilege management.

A diagram of a network

Description automatically generated with medium confidence

Figure 4: Correlation of Top CWEs, Vendor-Specific CWEs, and MITRE's Top 25 CWEs, Nov 2023

This analysis indicates a particular vulnerability in Microsoft’s products concerning privilege management. A significant 40% of Microsoft-associated CVEs relate to this issue, compared to just 5% for Apple, the second-most represented vendor in the catalog.

A graph with blue squares and numbers

Description automatically generated with medium confidence

Figure 5: Leading Vendors and Their Corresponding Top CWEs, Nov 2023

**Conclusion:**

This research substantiates the hypothesis that some software weaknesses, defined by CWEs, are more prevalent in CISA's catalog of known exploitable vulnerabilities. Notably, Microsoft emerges as a frequent vendor with a significant share of issues relating to improper privilege management. This could be attributed to its wide adoption as the operating system of choice among many Federal Civilian Executive Branch agencies (FCEBs), rendering it a prime target for privilege escalation attacks.

From a cybersecurity standpoint, these insights suggest that mitigation efforts should focus on the most common CWEs, particularly in products from commonly represented vendors like Microsoft, Apple, and Cisco. The distinct correlation between certain CWEs and specific software types implies that some venders’ software is inherently more susceptible to particular weaknesses. This knowledge can inform more targeted and effective mitigation strategies. For instance, FCEBs predominantly using Microsoft products should, at the very least, implement zero-trust zones, enforce the principle of least privilege rigorously, and maintain a strict separation of privileges. Such strategies could significantly diminish the risk associated with CVEs classified under CWE-269 (Improper Privilege Management). Similarly, for agencies leveraging Apple products, strategies should include redundancy in availability and integrity to ensure critical operations continue seamlessly and sensitive information remains secure, even in the event of vulnerabilities exploiting CWE-787. For Cisco product users, limiting access exclusively to authorized entities could effectively reduce exposure to harmful inputs. While these suggestions may be untenable for some organizations, they serve as a solid starting point for beginning their mitigation strategies.

**Work Cited**

Cybersecurity & Infrastructure Security Agency. (n.d.). Known exploited vulnerabilities catalog.CISA. Retrieved from <https://www.cisa.gov/known-exploited-vulnerabilities-catalog>

Cybersecurity & Infrastructure Security Agency. (n.d.). Emergency directive 21-04: Mitigate Windows Print Spooler Service Vulnerability. CISA. Retrieved from <https://www.cisa.gov/news-events/directives/ed-21-04-mitigate-windows-print-spooler-service-vulnerability>

MITRE. (2023). 2023 Stubborn Weaknesses. Retrieved from <https://cwe.mitre.org/top25/archive/2023/2023_stubborn_weaknesses.html>

National Institute of Standards and Technology. (n.d.). CVSS v3.1: Specification document. NIST. Retrieved from <https://nvd.nist.gov/vuln-metrics/cvss>

**Appendixes:**

**Appendix A: Scrape and Parse from the web**

CVE\_CWE\_Matcher\_7.py: <https://github.com/blugo2/AIT-664/blob/main/CVE_CWE_Matcher_7.py>

cwe-searcher.py: <https://github.com/blugo2/AIT-664/blob/main/cwe-searcher.py>

nist\_cvss2.py: <https://github.com/blugo2/AIT-664/blob/main/nist_cvss2.py>

**Appendix B: Convert CVSSv2 base scores to CVSSv3.1 severity ranking**

CVSS\_V\_Mapping.py: <https://github.com/blugo2/AIT-664/blob/main/CVSS_V_Mapping.py>

**Appendix C: Assess ‘none’ CWEs**

Script\_for\_cwe\_manual\_review.py: <https://github.com/blugo2/AIT-664/blob/main/script_for_cwe_manual_review.py>

**Appendix D: Analyze results**

Analyze\_crosstab\_CWE.py: <https://github.com/blugo2/AIT-664/blob/main/analyze_crosstab_CWE.py>