Scientific Research Publishing

Journal of Information Security, 2023, 14, 330-342
https://www.scirp.org/journal/jis
ISSN Online: 2153-1242
ISSN Print: 2153-1234

Systematic Review: Analysis of Coding Vulnerabilities across Languages

Shreyas Sakharkar

Stevenson Ranch, USA
Email: shrey16sakharkar@gmail.com

How to cite this paper: Sakharkar, S. (2023) Systematic Review: Analysis of Coding Vulnerabilities across Languages. *Journal of Information Security*, **14**, 330-342.
https://doi.org/10.4236/jis.2023.144019

Received: August 15, 2023
Accepted: September 25, 2023
Published: September 28, 2023

Abstract

The boom of coding languages in the 1950s revolutionized how our digital world was construed and accessed. The languages invented then, including Fortran, are still in use today due to their versatility and ability to underpin a large majority of the older portions of our digital world and applications. Fortran, or Formula Translation, was a programming language implemented by IBM that shortened the apparatus of coding and the efficacy of the language syntax. Fortran marked the beginning of a new era of efficient programming by reducing the number of statements needed to operate a machine several-fold. Since then, dozens more languages have come into regular practice and have been increasingly diversified over the years. Some modern languages include Python, Java, JavaScript, C, C++, and PHP. These languages significantly improved efficiency and also have a broad range of uses. Python is mainly used for website/software development, data analysis, task automation, image processing, and graphic design applications. On the other hand, Java is primarily used as a client-side programming language. Expanding the coding languages allowed for increasing accessibility but also opened up applications to pertinent security issues. These security issues have varied by prevalence and language. Previous research has narrowed its focus on individual languages, failing to evaluate the security. This research paper investigates the severity and frequency of coding vulnerabilities comparatively across different languages and contextualizes their uses in a systematic literature review.

Keywords

CWE (Common Weakness Enumeration), Data Security, Coding Vulnerabilities

1. Introduction

Coding languages have been used for website building and application develop-

ment in the modern era. Security issues within these applications can cause significant problems worldwide, mainly due to organizational utilization, from Alphabet to Amazon Web Services (AWS). Since the coding languages have been used to build such websites differ, the security problems that arise are inherently different in type of risk and prevalence. Data compromises in the United States increased from 157 in 2005 to 1,802 compromises in 2022 as programming gained popularity [1].

A famous coding vulnerability example comes from the $100,000 Keying Error. Grete Fossbakk was trying to transfer $100,000 to her daughter via an online banking system. She entered her daughter's bank account number and added one extra digit in the middle. The value was, therefore, truncated incorrectly, and the money was sent to the wrong person. According to Kai A. Olsen, a check-sum mechanism, checking the number of digits in the bank account, was in place to catch such errors, but it was only effective 92% of the time [2]. This resulted in this money transfer error in cases like Fossbackk's and others.

While money transfer and security issues are concerning, the stakes are often even higher, considering machinery dealing with reactive materials with control systems managed by coding language algorithms. One example was Therac-25, a control system controlled radiation therapy machine used in patient cancer treatments. Six significant accidents in the system due to computer language failures from a past version of the system and improperly presented error codes resulted in a large overdose of radiation being delivered to the patients. Moreover, overflow errors would allow the software to bypass safety checks occasionally since the error was numeric, not a fixed flag value [3]. Medical malpractice cases uncovered the algorithmic errors underpinned by the language's security risks. The use of alternative languages would have circumvented this security risk. The crucial nature of language security shoring can be further exemplified in a multitude of other industries, an example of which is the space industry. Rocket Ariane 5 also malfunctioned due to software errors and had catastrophic consequences, exploding in the atmosphere seconds after launch. Once again, reusing old code that needed to be assessed for malfunctions and security risks proved problematic. Peter Ladkin said the failure was "caused by an 'Operand Error' in converting data in a subroutine from 64-bit floating point to 16-bit signed integer. One value was too large to be converted, creating the Operand Error" [4]. Ladkin goes on to blame the failure of using the coding language Ada for the mission. Ada does not accommodate the need for explicitly hard-coding basic low-level data conversion into a higher-level language. Using Ada was utterly unnecessary, but as a result of using it, the entire mission was a failure.

Different coding languages have different inherent susceptibilities to coding security risks in the form of cyberattacks. These attacks can generally be classified into four main categories: cross-site scripting issues, SQL injection, command injection, and cryptographic issues. One type of attack is XSS or cross-site

scripting. In this injection attack, attackers try to incorporate malicious code into a credible website to attack a user. SQL injections are similarly injected malicious code to view backend data that is not meant to be accessible. Command injections exploit operating system (OS) vulnerabilities by executing arbitrary commands to exploit input validation issues. Cryptographic issues are the most diverse of these. Cryptographic attacks can be subclassified into three main aims: identity verification, confidentiality, and integrity. Each of these encompasses data security for certain users. Within each language, there is a different vulnerability that skews in the direction of one of these four cyberattack categories. 86% of applications utilizing general-purpose scripting languages, such as PHP, contain at least one cross-site scripting (XSS) vulnerability and are vulnerable to command injections [5]. However, there are differential security risks assigned to the specific languages. Languages like C and C++ are known to have more robust security against XSS, while languages such as Classic ASP and ColdFusion are known to have a greater prevalence of XSS issues. Lastly, cryptographic problems are more varied and challenging to protect against. Overall, cryptographic issues have become the most prevalent subtype of cyberattacks.

Different languages, thus, offer varying levels of security against attacks. However, delving into the specifics of each language's vulnerabilities that underpin these cyberattack data and validating this cyberattack distribution data by using a bottom-up approach is essential. This research paper will try to identify the following research questions: (1) What coding vulnerabilities/security risks come with using different languages? (2) Which coding languages are most vulnerable when faced with the most common cyberattacks?

2. Methodology

To answer the research aims, this paper will follow the methodology below. A systematic literature review of recent studies was conducted in order to provide a clear and comprehensive review of previous literature's data and interpretations. One database, Google Scholar, was utilized. Keywords that were used included "security risks", "coding vulnerabilities", "cyberattacks", "Python", "JavaScript" "Java", "C", "C++", "PHP" and "risks inherent in coding languages". Each paper was analyzed for security risks and vulnerabilities associated with each programming language. Furthermore, the uses and applications of each programming language were identified. By comparing the statistics of the severity and frequency of coding vulnerabilities in different languages, the risk of using each language can be assessed. A PRISMA 2020 model was followed. Systematic literature reviews, case studies, and meta-analyses were all included. As shown in **Figure 1**, thirty-seven studies were found to fit within the inclusion criteria. Of these, two were duplicative and removed. Thirty-five studies remained to conduct our literature review with.

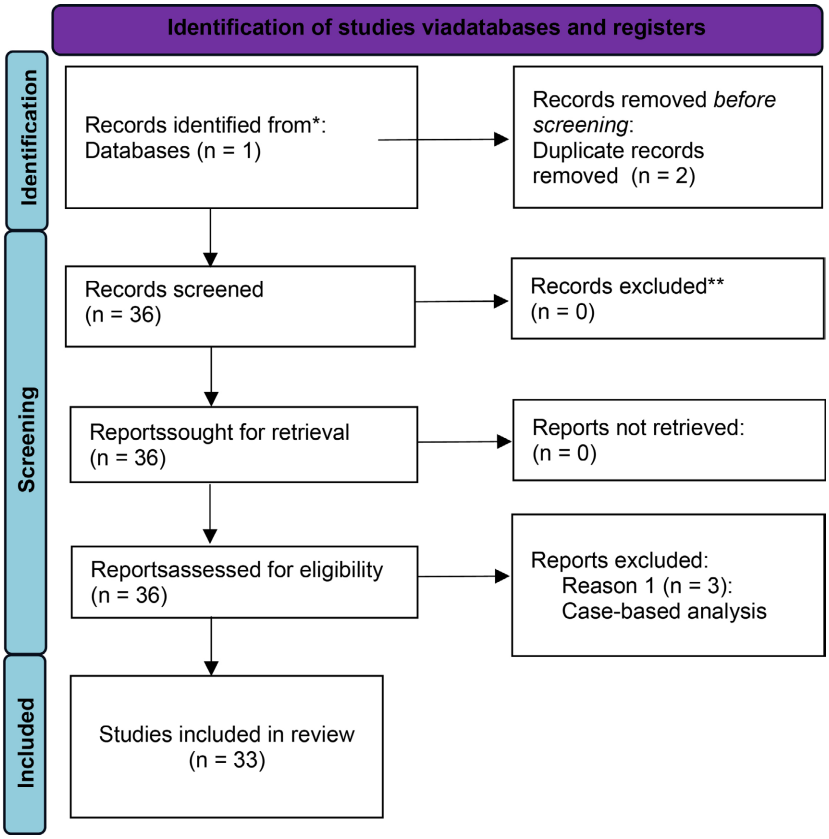

Figure 1. PRISMA.

3. Results

3.1. Python

In recent years, Python has experienced an explosion in popularity. According to the TIOBE Index, as of August 2022, Python overtook C to become the most popular programming language with a rating of 15.42% and is likely to keep growing. [6] Its popularity is mainly attributed to its simple and easy-to-read syntax.

Since Python is easy to code and also user-friendly, it is much easier on an individual quality analyst level to catch mistakes and errors in programs that could cause security issues. According to mend.io, only 6% of all reported open-source vulnerabilities are from websites written in Python. Furthermore, on average, only 15% of Python's severity vulnerabilities in the past five years are considered high risk, which is relatively low compared to other major coding languages [7].

However, Python also has some security vulnerabilities that are worth considering. There are four principal security vulnerabilities according to Python's CWE (Common Weakness Enumeration) list. In order from most frequent to least, Input Validation (CWE-20), Permissions, Privileges, and Access Control (CWE-264), Cross-Site Scripting (XSS) (CWE-79), and Information Leak/Disclosure (CWE-200) affect Python language-based scripts [7].

In further analysis, Python security risks are primarily a result of insufficient

user input validation [8]. Input validation is how a program can ensure the data being entered by a user is not malicious, incorrectly formatted, or wrong in other ways. A program can avoid leaking important information by sanitizing the user's input.

Additionally, Python is a type-safe language. This means it checks for type errors when the program compiles, known as compile time. On the other hand, languages that are not type-safe check for type errors at runtime. This is advantageous because it leads Python to be less error prone. Languages that are not type-safe, such as C and C++, are susceptible to error due to not having type safety.

3.2. JavaScript

JavaScript is a popular language, and according to the TIOBE Index, JavaScript ranks seventh on the list of the most popular programming languages in the world, with a rating of 2.33% as of August 2022 [6]. However, Stack Overflow, a popular programming website, posted a developer survey in 2021 that claimed JavaScript was the most popular language. 64.96% of respondents and 68.62% of professionals chose JavaScript as their most heavily favored and used development language [9]. JavaScript underpins nearly 95+% of websites accessible today, emphasizing the widespread nature and havoc cyberattacks could wreak by exploiting a singular vulnerability [10].

Despite being so popular, JavaScript has more vulnerabilities than Python. According to mend.io, 11% of all reported open-source vulnerabilities are from websites incorporating JavaScript. Mend.io also states that, on average, 31% of vulnerabilities identified in websites incorporating JavaScript contained high-risk vulnerabilities over the last five years, more than double that of Python.

According to mend.io, JavaScript is vulnerable to three main CWEs: Cryptographic Issues, Path Traversal, and Cross-Site Scripting (XSS), the first two relatively uncommon among other top programming languages. Additionally, Michael Hollander claims that SQL injection and sensitive cookie exposure are among the top vulnerabilities of JavaScript [10].

3.3. Java

Java is extremely popular as well. According to the TIOBE Index, which is a measure of relative programming language popularity, Java is ranked third in popularity at 12.40% [6]. Also, Java is a type-safe language, which is beneficial in minimizing type errors. Furthermore, Java technology is utilized by six million developers across billions of devices [11]. As a result, any significant vulnerability found in Java could potentially result in billions of devices being at risk of being hacked. Therefore, it is crucial to examine whether Java is secure.

According to mend.io, Java comprises 11% of all open-source security vulnerabilities, placing it in the top 3 in this category. However, the number of websites coded with Java containing high-severity security vulnerabilities averages

roughly 19% but has been declining since 2015. Additionally, Java shares Python's top four CWEs: Information Leak/Disclosure, Input Validation, Cross-Site Scripting, and Permissions, Privileges, & Access Control [7]. Java and Python have several similarities, which provide a reason why their top security vulnerabilities are the same.

However, being listed among the top CWEs does not indicate a critical vulnerability. Static application security testing (SAST) helps identify code vulnerabilities. The most common code vulnerability identified through this process was the Unpatched Libraries vulnerability [12]. Furthermore, S. Rahaman *et al.* concluded that on a cumulative evaluation of Android applications, these applications draw their vulnerabilities largely from packaged libraries of code [13]. Unpatched libraries occur when versions of coding languages are updated to patch vulnerabilities found in libraries. Lists of these vulnerabilities can be found in databases such as the National Vulnerability Database or the CVE database. Coding languages can be vulnerable to attackers who know these vulnerabilities when not updated.

3.4. C

C is a relatively old language still in use. According to the TIOBE Index, C ranks second in popularity to Python and ranked first before August 2022. However, according to a survey by Stack Overflow, only 21.01% of respondents have done extensive development work with C [6].

Despite its broad applicability, there are three significant components to this need for more familiarity with C. 1) C is not very user-friendly and is challenging to master due to syntax requirements and hard-coding practices. 2) C contains security flaws and infrastructure weaknesses that are easy to exploit [14]. 3) C is not type-safe, making C increasingly liable to type errors. Mend.io states that C has the most vulnerability out of the above languages, encompassing 50% of all reported vulnerabilities in a decade. However, this last type-safety argument may be unreliable, as it is worth mentioning that the length of time that C has been around and the mass of code that relies on C may skew this data improperly [7].

C's top CWE is the Improper Restriction of Operations within the Bounds of a Memory Buffer, scoring 75.56 out of 100. Out-of-bounds Read is the second-most common CWE with a score of 26.5, far less common than the prior [15].

3.5. C++

C++ ranks fourth on the TIOBE Index, with a rating of 10.17% [6]. However, according to the Stack Overflow Developer Survey, 24.31% of respondents have done extensive development using C++, which ranks 10th on this survey. This points to the improvement in user accessibility and ease of use.

Similar to the above, in terms of security, C++ is not type-safe, so it is vulnerable to type errors. Notably, C++ has two significant unique security risks. First,

C++ has the highest % of high-severity security vulnerabilities in this group at 36% [7]. Secondly, C++'s top CWE is Buffer Errors, which accounts for over 50% of all CWEs. According to mend.io, these Buffer errors, referred to by CWE code 19, have been incredibly prevalent within C. As explained by the lead-time fallacy of C, it is now becoming apparent that they are not unique to C. C++ has seen a sharp spike in these errors since 2017 [7]. Furthermore, as shown in **Figure 2**, C and C++ are uniquely responsible for a significant portion of CWE vulnerabilities [16].

3.6. PHP

According to the TIOBE Index, PHP is ranked tenth with a popularity rating of 1.39% [6]. 21.98% of respondents to the Stack Overflow 2021 Survey state that they have done extensive development with PHP [9]. PHP is especially popular in website development, with continuous growth and applicability [17]. This is primarily due to web developers' use of PHP in creating dynamic content and utilizing strong database support systems. Overall, this can provide a user-friendly experience [18].

Of security, according to mend.io, 16% of websites coded with PHP in the last five years were found to contain high-severity vulnerabilities. PHP has one clear top CWE: Cross-Site Scripting, or XSS. While we have explored the prevalence of XSS CWEs in depth, PHP is the only language with prominent SQL Injection (CWE-89) vulnerabilities [7].

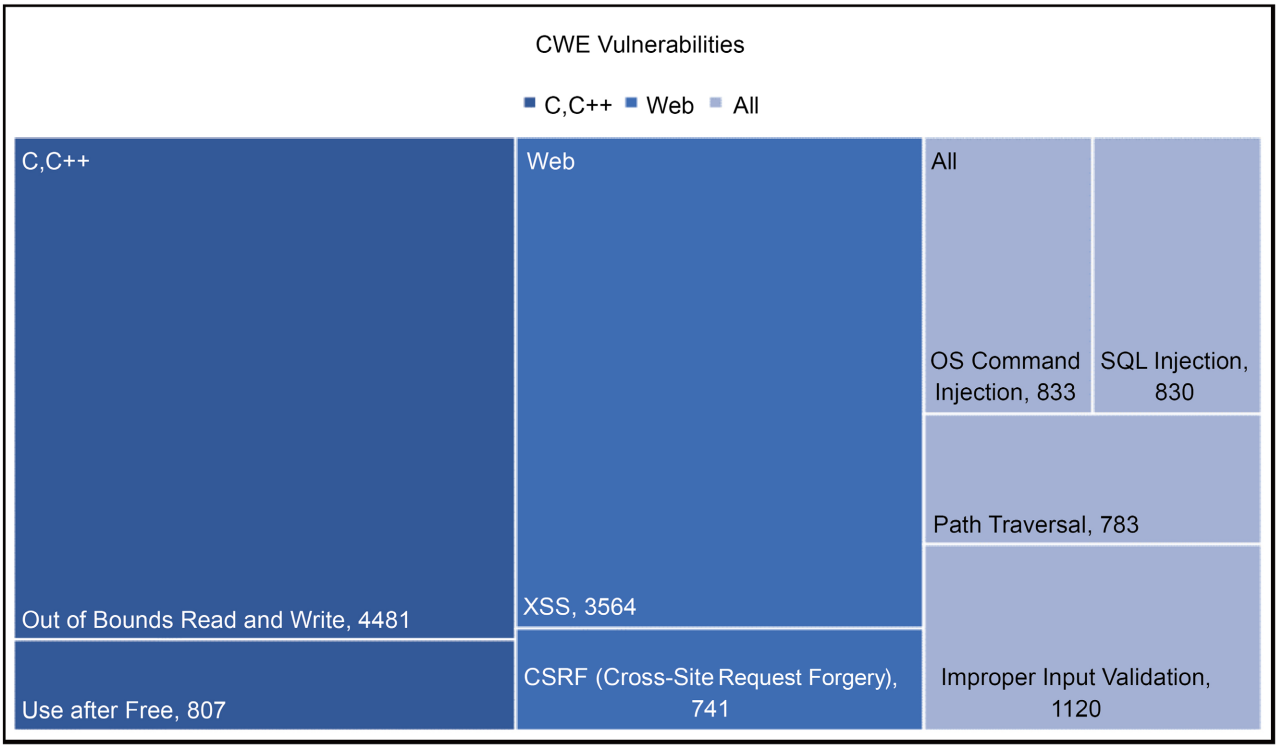

Figure 2. Number of CWE vulnerabilities for 2021 by language and type.

4. Discussion

In order to correctly interpret this research, it is also important to realize that each of the languages discussed earlier has different purposes. Despite their vulnerabilities, languages may still be used solely to execute a specific task that no other language can do better.

4.1. Applications of Python

Python has a relatively low number of vulnerabilities, which is a positive feature as it is one of the most used programming languages. Most of its applications involve image processing and graphic design applications [19]. Furthermore, Python can be used in website/software development, data analysis, and task automation. Since Python is very user-friendly, easy to learn, and has a simple syntax, many accountants and scientists can also utilize it for their job requirements [20]. It is often heralded as a beginner-friendly language for new web developers and computer science students.

4.2. Applications of JavaScript

JavaScript is a client-side programming language mainly used in web development, specifically to make websites dynamic and interactive [21]. 98.0% of all websites use JavaScript as a client-side programming language [22]. JavaScript vulnerabilities can be high-stakes as one of the more popular languages underpinning some of the heaviest trafficked web pages without significant failures or security risks. Examples include Google, YouTube, Facebook, Wikipedia, and Amazon. As a result, any vulnerability found in JavaScript can be used cross-applied across several websites that are not only important but can be financially catastrophic.

4.3. Applications of Java

Java can be used to construct applications on several platforms, particularly for the recent video gaming boom [23]. However, Java is not limited to mobile and computer applications. One of the more novel security risks that have emerged from using Java comes from the National Aeronautics and Space Administration (NASA). One of NASA's recent endeavors has been WorldWind, a publicly accessible application allowing high-resolution camera monitoring from satellite feed onto nearly any location on Earth [24]. As a result, a security vulnerability in Java could result in apps being targeted by attackers and potentially crashing.

4.4. Applications of C

As established, the relatively higher number of C-specific vulnerabilities and their prevalence compared to other languages means that any application based on C has an increased security risk. The applications of C are just as foundational to our computers, underpinning most operating systems (OS). Many C-based standard OS include but are not limited to Windows, Linux (nearly entirely in

written C), Google Chrome OS, RIM Blackberry OS 4.x, Symbian OS, Apple Mac OS X, iPad OS, Apple, and Cisco IOS [25]. Security recommendations point toward minimizing the usage of C, especially in website development, where security risks are incrementally increased. Despite its shortcomings, C is still a prevalent programming language.

4.5. Applications of C++

C++ is an all-purpose programming language used for many projects, such as applications, games, browser development, and operating systems [26]. Since C++ is used over such a large spectrum of software programs, it is vital to ensure coding vulnerabilities in C++ do not cause significant security issues. Otherwise, the same basic vulnerabilities can be exploited across a plethora of programs using C++.

4.6. Applications of PHP

PHP is a scripting language mainly used for web development, but it can also be used to make projects such as GUIs or Graphical User Interfaces [27]. Many popular web interfaces use PHP, including Facebook, Wikipedia, and Etsy, among many others [28].

4.7. Origin of Security Vulnerabilities

Many security vulnerabilities originate from an early stage when beginners are taught how to program but not how to code securely. Taylor *et al.* demonstrated that significant work on SQL injection vulnerabilities shows a surprising lack of concern or discourse around preventative security measures [29]. This lack of education around preventive measures yields a measure of minimizing security concerns and a boom in analysts that create accessible products; websites, applications, and other programs inevitably contain vulnerabilities. Hackers can often easily exploit these vulnerabilities, resulting in data breaches and significant fiscal consequences in the industry.

Furthermore, in recent years, we experienced an explosion in the number of vulnerabilities from 2017 onwards, potentially due to the C++ spike in security issues and applicability [30]. This underscores the importance of secure coding practices, as more vulnerabilities mean more access points for attackers to target.

4.8. Research Implications

By exploring the risk of using popular programming languages in everyday computer programs and websites, this research paper what risks are involved in using each language. Furthermore, by identifying the various uses of each language, this paper can help software programmers weigh the advantages and disadvantages of using specific languages for certain tasks.

5. Cyberattacks by Vulnerability Types

In this section, data and trends are analyzed in order to identify how secure dif-

ferent programming languages are by investigating how well each language reacts to the leading types of cyberattacks.

5.1. Cross-Site Scripting (XSS)

To recoup, cross-site scripting, or XSS, is a type of cyberattack that occurs when attackers incorporate malicious code into a credible website to attack an unsuspecting user. Vulnerabilities susceptible to XSS can be found very often; according to a Virginia Journal of Science study, more than 60% of web applications are vulnerable to XSS attacks [31].

XSS is commonly found among the top vulnerability list of leading coding languages. According to the mend.io WhiteSource Report, PHP, Javascript, Java, and Python all feature XSS as one of their top three most common security vulnerabilities [7]. This trend exemplifies XSS's threat to several websites, as PHP and Javascript are ubiquitous in website development.

5.2. Input Validation

Input Validation is a process that involves verifying a user's input to ensure it follows a set of standards. For example, an input field asking for an email address from the user can verify the user's input by checking for an "@" symbol and other features. However, this process can sometimes be bypassed. Attackers may also be able to obtain private information. One scenario in which this could happen is when a person orders an item off of a website but inputs an SQL command that fetches the credit card numbers of different people who have previously ordered.

Input Validation is widespread among top coding languages. According to Mend's WhiteSource Report, the CWE for Input Validation is second-most common in C, Java, and C++ and is most common in Python (when compared to other CWEs) [7].

5.3. SQL Injection

SQL Injection is a strategy that attackers use against weak input validation. Attackers can input a malicious SQL string of code into a website, allowing them to access restricted information from databases. In a basic example, attackers could type "SELECT * FROM Users" in a text box to get the elements in the "Users" table.

Mend's WhiteSource Report ranks SQL Injection as SQL's second-most common CWE vulnerability [7]. Furthermore, according to research by the internet company Akamai, SQL Injection accounts for over 65.1% of all web application attacks [32].

5.4. Out-of-Bounds Write

Out-of-bounds Write occurs when a program writes data outside a set buffer, causing the program to crash or fail to run. Another name for this is Buffer Over-

flow. Attackers can take advantage of buffer overflow to crash the program on purpose or cause the program to do what the attacker wants.

According to the CWE website, Out-of-bounds Write ranks first in their 2022 Top 25 Most Dangerous Software Weaknesses list [33].

6. Conclusions

This paper aimed to answer the research questions: (1) What do coding vulnerabilities/security risks come with using different languages? and (2) Which coding languages are most vulnerable when faced with the most common cyberattacks? To answer these questions, the various vulnerabilities that affect trending programming languages were analyzed by the top CWEs and other trends for each language. C and C++ experience the most vulnerability and are most vulnerable to cyberattacks. While Python, Java, JavaScript, and PHP are not entirely secure, they still are relatively safer than C and C++.

In order to decrease the frequency of cyberattacks, future users of programming languages must take into account common vulnerabilities for each language and be mindful not to replicate such security threats. This is especially important for users of C and C++, who can drastically minimize the number of vulnerabilities in their code by adding security measures to prevent hackers from exploiting such vulnerabilities.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Petrosyan, A. (2023) Number of Data Breaches and Victims U.S. 2022. Statista. https://www.statista.com/statistics/273550/data-breaches-recorded-in-the-united-states-by-number-of-breaches-and-records-exposed/

[2] Olsen, K.A (2010) Two Cases of Bad Web Usability: Banking and Employee Self Service, UniTech 2010, Oslo, Tapir Forlag (19.05). http://medialt.no/pub/uikt/u2010/035-Olsen/index.html

[3] Caballero, C. (2019) Software Architecture: Therac-25 the Killer Radiation Machine. https://www.carloscaballero.io/software-architecture-therac-25/

[4] Ladkin, P.B. (1998) The Ariane 5 Accident: A Programming Problem? http://www.rvs.uni-bielefeld.de/publications/Incidents/DOCS/Research/Rvs/Misc/Additional/Reports/ariane.html

[5] Khandelwal, S. (2015) These Top 10 Programming Languages Have Most Vulnerable Apps on the Internet. https://thehackernews.com/2015/12/programming-language-security.html

[6] (2022) TIOBE Index. https://www.tiobe.com/tiobe-index/

[7] Mend (2022) Most Secure Programming Languages. https://www.mend.io/most-secure-programming-languages/

[8] Bless, G. (2021) Most Common Python Vulnerabilities and How To Avoid Them.

https://infosecwriteups.com/most-common-python-vulnerabilities-and-how-to-avoid-them-5bbd22e2c360

[9] (2022) Stack Overflow Developer Survey 2021. https://insights.stackoverflow.com/survey/2021#most-popular-technologies-language

[10] Hollander, M. (2020) Most Common Security Vulnerabilities Using JavaScript. Secure Coding. https://www.securecoding.com/blog/most-common-security-vulnerabilities-using-javascript/

[11] Oracle (2022) Moved by Java Timeline. https://www.oracle.com/java/moved-by-java/timeline/

[12] DZoneRefcardz (2022) Java Application Vulnerabilities. https://dzone.com/refcardz/java-application-vulnerabilities#section-3

[13] Rahaman, S., *et al.* (2019) CryptoGuard: High Precision Detection of Cryptographic Vulnerabilities in Massive-Sized Java Projects. Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, 11-15 November 2019, 2455-2472. https://doi.org/10.1145/3319535.3345659

[14] HPE: Hewlett Packard Enterprise (2022) Making C Less Dangerous. https://www.hpe.com/us/en/insights/articles/making-c-less-dangerous-1808.html

[15] Parasoft (2020) Battling Buffer Overflows & Other Memory Management Bugs. https://www.parasoft.com/blog/battling-buffer-overflows-other-memory-management-bugs/

[16] Dias, B. (2022) Most Dangerous CWEs of 2021. https://checkmarx.com/blog/most-dangerous-cwes-of-2021/

[17] Raygun Blog (2021) 10 Popular PHP Frameworks for Web Developers to Consider in 2021. https://raygun.com/blog/top-php-frameworks/

[18] Singla, L. (2021) What Is PHP for Web Development and Why Should You Use It? Insights—Web and Mobile Development Services and Solutions. https://www.netsolutions.com/insights/what-is-php/

[19] upGrad blog (2022) Top 12 Fascinating Python Applications in Real-World [2022]. https://www.upgrad.com/blog/python-applications-in-real-world/

[20] Coursera (2023) What Is Python Used For? A Beginner's Guide. https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python

[21] Meltzer, R. (2020) What Is JavaScript Used For? Lighthouse Labs. https://www.lighthouselabs.ca/en/blog/what-is-javascript-used-for

[22] (2022) Usage Statistics of JavaScript as Client-Side Programming Language on Websites. https://w3techs.com/technologies/details/cp-javascript

[23] Future Learn (2022) What Is Java Used for? 6 Practical Java Uses. https://www.futurelearn.com/info/blog/what-is-java-used-for

[24] New Relic (2022) 6 Amazing Ways You Can Use Java. https://newrelic.com/blog/nerd-life/what-you-can-do-with-java

[25] OpenEDG (2022) Learn C and C++, the Languages of the Past and Today. https://edube.org/learn/cpp-essentials-v20-docs/c-essentials-v1-1-faq-3

[26] Castro, S. (2021) 6 Reasons C++ Is Still In Use Today. https://jobsity.com/blog/6-reasons-c-is-still-in-use-today

[27] Chris, K. (2021) What Is PHP? The PHP Programming Language Meaning Ex-

plained.
https://www.freecodecamp.org/news/what-is-php-the-php-programming-language-meaning-explained/

[28] Fleury, D. (2020) 7 Global Websites That Use PHP in 2022. Trio Developers. https://www.trio.dev/blog/companies-using-php

[29] Almansoori, M., Lam, J., Fang, E., Soosai Raj, A.G. and Chatterjee, R. (2021) Textbook Underflow: Insufficient Security Discussions in Textbooks Used for Computer Systems Courses. *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, Virtual Event, 13-20 March 2021, 1212-1218. https://doi.org/10.1145/3408877.3432416

[30] NVD: National Vulnerability Database (2022) CVSS Severity Distribution over Time. https://nvd.nist.gov/general/visualizations/vulnerability-visualizations/cvss-severity-distribution-over-time#CVSSSeverityOverTime

[31] Mack, J., Hu, Y.H. and Hoppa, M.A. (2019) A Study of Existing Cross-Site Scripting Detection and Prevention Techniques Using XAMPP and Virtual Box. *Virginia Journal of Science*, **70**, 23 pp.

[32] (2019) Akamai Threat Research Points to Gaming Industry as a Rising Target with 12 Billion Attacks and Counting. https://www.akamai.com/newsroom/press-release/state-of-the-internet-security-web-attacks-and-gaming-abuse

[33] CWE: Common Weakness Enumeration (2022) 2022 CWE Top 25 Most Dangerous Software Weaknesses. https://cwe.mitre.org/top25/archive/2022/2022_cwe_top25.html