AN OVERVIEW OF DNS DOMAINS/SUBDOMAINS VULNERABILITIES SCORING FRAMEWORK

Sanat Talwar (Corresponding Author)[1] , Aakarsh Mavi[2]
[1]sanattalwar1994@gmail.com
[2]mavi.aakarsh4@gmail.com

Abstract
Keeping up with today's quickly changing IT landscape has never been more difficult than maintaining complicated networks. This intricacy frequently results in mistakes and omissions, which create weaknesses. An outline of creating a rating system for DNS domain and subdomain vulnerabilities is given in this paper. It begins by reviewing the foundational ideas of DNS, such as domains and subdomains. After understanding the underlying technology, the paper looks at problems that haven't been sufficiently addressed by previous studies or frameworks, pointing out any holes that still need to be filled. Lastly, it suggests a new framework that presents fixes and scoring methods, such as creating a formula to standardize vulnerability ratings and generate a result on a scale of 1 to 10.

Keywords
Domain Name System (DNS), Vulnerability Scoring, Cybersecurity Framework, Risk Priority, CNAME Records, Open Ports, Internet Protocol (IP) Address, Subdomain Scoring, Automation in Security, Threat Mitigation.

1. Introduction

The Domain Name System (DNS) serves as the Internet's phonebook, allowing humans to access information online using domain names **[1]** rather than numerical Internet Protocol (IP) addresses. Although every entity on the internet has an IP address, people cannot realistically memorize the roughly 4.3 billion potential IPv4 addresses **[7]**. To overcome this difficulty, DNS serves as a directory that converts IP addresses into domain names. For example, the IP address 192.10.1.1 may be converted to example.com using DNS queries. With the help of this translation, users may quickly visit websites by entering domain names rather than complicated IP addresses.

Understanding domains and subdomains is crucial to comprehending the research described in this paper. Let's use an example to better grasp this. Domains serve as a firm's primary phone number, and subdomains serve as extensions that link users to particular company departments. Likewise, a subdomain is an extension of the primary domain in the context of DNS.

For instance:

192.10.1.1 → example.com

└── 192.10.1.6 → sub1.example.com

Because DNS is essential to the operation of the internet, its security must be ensured. DNS system vulnerabilities can have serious repercussions, which makes them appealing targets for bad actors. Because customer-facing domains are especially vulnerable, firms must routinely handle risks. Finding and ranking the domains and subdomains that require urgent cleanup is still quite difficult.

To address this gap, this paper introduces the Scoring Framework for Subdomain Security (SCSS). Businesses may effectively allocate resources and manage risks by using this framework, which offers a systematic, quantitative method for evaluating and prioritizing vulnerabilities. In order to assess the priority for remediation, the SCSS framework assigns weights to each of the elements that contribute to DNS vulnerabilities and scores them accordingly.

This rating system also makes it easier to investigate the most serious flaws that impact domains and subdomains in an environment. The framework is used to score and rank several dummy domains using the SCSS technique to show how effective it is. This paper analyzes the gaps in existing models before introducing the framework, which helped shape the methodology and score algorithm.

2. Literature Review

2.1 How Existing Research Falls Short

- **Lack of Prioritization Mechanisms**

Many studies, such as those by Zhao et al. (2021) and Wang & Wang (2020), focus on detecting vulnerabilities or categorizing misconfigurations without addressing the challenge of prioritizing which subdomains to secure first. This paper introduces a scoring framework (SCSS) that considers multiple dimensions—criticality, age, hierarchy, and exposure—to prioritize subdomain remediation.

- **Limited Integration of Factors**

Existing works often focus on individual factors like CVSS scores (Yadava & Singh, 2022) or attack surfaces (Matherly, 2016) without integrating these into a comprehensive model. The SCSS framework combines these diverse factors into a unified scoring algorithm, offering a structured approach to vulnerability management.

- **Neglect of Organizational Context**

Research by Ali & Rajpoot (2018) provides theoretical insights into DNS vulnerabilities but does not address practical implementation challenges within organizational contexts. This paper bridges this gap by offering actionable steps for organizations to assess and prioritize subdomain vulnerabilities using real-world and experimental data.

- **Overemphasis on Detection Tools**

Tools like Shodan and DNS Dumpster discussed in existing research, are valuable for data gathering but lack a framework for analyzing the gathered data in a way that prioritizes actionable results. This paper complements these tools by providing a methodology to evaluate and rank subdomains systematically.

- **No Structured Scoring Framework**

While some works discuss vulnerability scoring using CVSS, they fail to account for other critical factors such as hierarchy, customer-facing exposure, and subdomain age. This proposed framework integrates these aspects to create a robust scoring system that is more practical for real-world applications.

3. Proposed Framework

3.1 Purpose and Objectives

The SCSS framework aims to:

1. Quantify and rank the security posture of domains and subdomains.
2. Provide a data-driven methodology to prioritize remediation efforts.
3. Incorporate multiple dimensions of risk, including public exposure, domain age, hierarchy, CVSS scores, and attack exposure.

3.2 Factors for Scoring:

1. Customer-Facing vs. Internal:

a. **Importance:** Customer-facing subdomains are publicly exposed, making them more vulnerable to attacks. These subdomains often host critical services that are directly accessible by end users, leading to a higher likelihood of exploitation.

b. **Impact on Score:** A binary factor, with customer-facing subdomains receiving a higher score (e.g., 1) due to increased exposure, compared to internal subdomains (e.g., 0), which are less likely to be targeted.

2. Age of Domain/Subdomain:

a. **Importance:** Older subdomains may have legacy DNS configurations or other outdated settings that increase the risk of subdomain takeover. These subdomains might have been orphaned or misconfigured over time.

b. **Impact on Score:** The age of the domain or subdomain is scored in years, with older domains assigned higher risk scores as they are more likely to have unmonitored or forgotten configurations that could be exploited.

3. Hierarchy (Domain vs. Multi-level Subdomain):

a. **Importance:** Multi-level subdomains (e.g., sub1.sub2.example.com) may have dependencies on other subdomains, increasing the complexity of the DNS setup and creating more points of failure. This can amplify the risks if any level in the hierarchy is misconfigured.

b. **Impact on Score:** A higher score is assigned to multi-level subdomains due to their increased complexity and interdependencies, which can create more avenues for exploitation.

4. Type of Vulnerability:

a. **Importance:** The severity of different vulnerabilities varies, from misconfigured DNS records to service disruptions. The framework classifies the vulnerabilities based on their potential to be exploited, such as CNAME issues, dangling DNS records, or expired services.

b. **Impact on Score:** Each type of vulnerability is assigned a severity weight, with critical vulnerabilities (such as CNAME issues pointing to discontinued services) being scored higher. Weights are based on the exploitability of the vulnerability.

5. Attack Exposure (Open Ports):

a. **Importance:** Exposed ports increase the attack surface, providing more entry points for attackers. Subdomains with open ports, especially those running services with known vulnerabilities, are at higher risk.

b. **Impact on Score:** This factor is normalized based on the number of open ports. More open ports lead to a higher score, as they directly increase the potential for attack.

3.3 Scoring Algorithm

Formulation:

The scoring algorithm combines all the above factors into a single vulnerability score. This allows organizations to rank and prioritize subdomains based on their overall risk level. The formula can be expressed as follows:

$$SCSS=w1\times CF+w2\times AD+w3\times H+w4\times TV+w5\times AE$$

- **Weights:**
  - CF/Internal: w1=2
  - Age: w2=1
  - Hierarchy: w3=2
  - CVSS Score: w4=3
  - Attack Exposure: w5=4

Normalization:

To ensure uniformity, individual factors like CVSS and AE are normalized to a scale that aligns their impact on the overall score. For example, CVSS scores are weighted more heavily due to their direct indication of criticality, while Age has a lesser weight.

4. Implementation:

To evaluate the efficacy of our algorithm, we would be making use of authentic and experimental data to apply our algorithm to it and to find out what is the order of domains/subdomains that the organization should be working on to fix it.

For the type of vulnerability, we will be making use of platforms such as the National Vulnerability Database (NVD) and CVE Details to find the CVSS score and type of vulnerability that domains/subdomains are vulnerable to. To not make vulnerable domains/subdomains public, we would be relying on experimental data to come up with an evaluation. For coming up with experimental data, we would be making use of open-source tools and sources to showcase how to get the relevant information on domains/subdomains, which companies can use as a framework for applying it to their organization data.

4.1 Data Points:

1. Domain/Subdomain list:
This is the list of all domains/subdomains that exist within organizational environments, whether production or development. It is best advisable for organizations to use an aggregated data source, which is used by DNS servers to resolve all the DNS entries, even though name servers correspond to different DNS zones.

2. Customer Facing / Internal:
Whether the domain/Subdomain is customer-facing or internal to the organization's network can be found using various open-source methodologies such as DNS Dumpster, Shodan, etc. These are set as Binary values as the input for the Scoring algorithm. A value of 1 indicates that the DNS entry is Customer-facing, and a value of 0 indicates that the DNS entry is Internal to the Organization network.

3. Age:
Age of DNS entry can be found as part of Passive enumeration, which can be done by using open-source tools such as WHOIS lookups, indicating the historical data on when the DNS entry was created, and also relevant details such as who is the registrar, etc.… We will be setting the entries in months, indicating how long before the DNS entry was created in months.

4. Hierarchy:
The hierarchy in our Scoring algorithm is to determine if the DNS entry is a domain or a subdomain, if it is a domain, then the binary value is set to 1, else it is set to 0. This factor is inculcated because of the reason that one level-up domain is more critical to an organization; if they are compromised, then the likelihood of Subdomains to that root domain increases.

5. Vulnerability CVSS Score:
Every Vulnerability has a relative score, which determines the category of vulnerability, further indicating how critical the vulnerability is. We will be making use of platforms such as the National Vulnerability Database (NVD) and CVE Details to find the CVSS score and type of vulnerability that domains/subdomains are vulnerable to.

6. Attack Exposure:
This is a factor that our Algorithm uses to find out how much is the attack surface, and it is inspired by the study that the bigger the attack surface, the more likelihood of DNS entry being compromised.

4.2 Data Table:

The table below provides sample data points necessary for formulating our framework.

Domain/Subdomain list	CF/Internal	Age	Hierarchy	Vulnerability CVSS score	Attack exposure
example.com	1	52	1	7.8	4
sub1.example.com	0	44	0	5.4	2
sub2.example.com	1	36	0	7.9	6
sub3.example.com	1	4	0	6.4	5
example2.com	1	80	1	7.2	3
sub1.example2.com	0	21	0	6.1	5
sub2.example2.com	0	15	0	3.5	7
sub3.example2.com	1	3	0	2.9	1
example3.com	1	139	1	3.4	3
sub1.example3.com	1	72	0	7.9	4
sub2.example3.com	1	54	0	5.8	5
sub3.example3.com	0	19	0	3.4	6

Table 1

4.3 Final Outcome:

The table below illustrates the final score calculation using the SCSS framework formula:

Example of Calculating a sample score:

Formula → SCSS=w1×CF+w2×AD+w3×H+w4×TV+w5×AE

Weights:

- CF/Internal: w1=2
- Age: w2=1
- Hierarchy: w3=2
- CVSS Score: w4=3
- Attack Exposure: w5=4

Example Calculation for example3.com: Using the formula and corresponding weights

SCSS = 2x1 + 1x139 + 2x1 + 3x3.4 + 4x3 = 165.2

Rank	Domain/Subdomain	CF	Age	Hierarchy	CVSS	Attack Exposure	Score
1	example3.com	1	139	1	3.4	3	165.2
2	example2.com	1	80	1	7.2	3	117.6
3	sub1.example3.com	1	72	0	7.9	4	113.7
4	example.com	1	52	1	7.8	4	95.4
5	sub2.example3.com	1	54	0	5.8	5	93.4
6	sub2.example.com	1	36	0	7.9	6	85.7
7	sub1.example.com	0	44	0	5.4	2	68.2
8	sub1.example2.com	0	21	0	6.1	5	59.3
9	sub2.example2.com	0	15	0	3.5	7	53.5
10	sub3.example3.com	0	19	0	3.4	6	53.2
11	sub3.example.com	1	4	0	6.4	5	45.2
12	sub3.example2.com	1	3	0	2.9	1	17.7

Table 2

5. Conclusion

The computed values reflect the final scores as established by the SCSS framework, as seen in Table 2. By beginning with the vulnerabilities that score the highest and working methodically from there, these scores can be utilized to prioritize fixing vulnerabilities in subdomains.

A systematic method for efficiently prioritizing subdomain vulnerabilities is provided by the SCSS architecture. Organizations can quickly address key risks by combining several factors and allocating the proper weights. By ensuring that remedial efforts are in line with organizational risk priorities, this paradigm closes a significant gap in the current vulnerability assessment methodologies.

To improve usability and adoption, this framework—which is merely a first step—can be further improved by incorporating it with widely used industry standards like NIST or MITRE.

6. References:

[1] What is DNS? | How DNS works (https://www.cloudflare.com/learning/dns/what-is-dns/)

[2] Zhao, Z., Xu, Z., & He, Z. (2021). Detecting and Mitigating Subdomain Takeover Vulnerabilities in Cloud Platforms.
This paper highlights methods for detecting subdomain takeovers in cloud platforms using heuristics and automated tools. However, it lacks a prioritization framework for addressing multiple vulnerabilities, which your SCSS framework provides.

[3] Wang, H., & Wang, Y. (2020). DNS Misconfigurations and Their Security Implications.
This work investigates the prevalence and types of DNS misconfigurations but fails to propose a structured scoring mechanism for organizations to prioritize fixes for domains and subdomains.

[4] Matherly, J. (2016). The Role of Shodan in Assessing Attack Surfaces.
While this paper introduces Shodan as a tool for identifying open ports and potential attack surfaces, it does not integrate this data into a unified scoring algorithm for subdomain vulnerabilities.

[5] Ali, M., & Rajpoot, Q. (2018). A Survey on Vulnerabilities and Mitigation Techniques in DNS Security.
This survey provides an overview of DNS vulnerabilities but does not offer actionable scoring frameworks for organizations to prioritize mitigation efforts.

[6] Yadava, S., & Singh, A. (2022). CVSS in DNS Vulnerability Management: A Critical Analysis.
This research evaluates the applicability of CVSS scores in DNS vulnerability management. However, it does not extend this to a holistic framework that considers factors like hierarchy and attack exposure.

[7] Public and Private IP Addresses Explained (https://www.matillion.com/blog/public-and-private-ip-addresses-explained)