Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.34, NO.4, Aug. 2024 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2024.34.4.667

N-day 취약점 데이터베이스 개선 및 활용 방안 연구*

정 종 선,[1†] 박 정 흠[2‡]
[1,2]고려대학교 (대학원생, 교수)

A Study on the Improvement and Utilization of Public N-Day Vulnerability Databases*

JongSeon Jeong,[1†] Jungheum Park[2‡]
[1,2]Korea University (Graduate student, Associate professor)

요 약

취약점이 공개된 후 소프트웨어가 업데이트되지 않으면 계속해서 공격을 받을 수 있다. 이로 인해 취약점을 악용하는 공격이 증가하면서 N-day 탐지의 중요성이 커지고 있다. 그러나 공개된 취약점 데이터베이스에서 특정 버전 정보를 찾기 어렵거나 잘못된 버전 혹은 소프트웨어가 출력되는 문제점이 발생한다. 또한 공개된 취약점 데이터베이스 간의 연계가 잘되지 않는다는 한계점이 있다. 본 논문에서는 이러한 한계점을 극복하기 위해 CVE, CPE, Exploit Database와 같은 종합적인 취약점 정보를 포함하는 정보들을 통합된 데이터베이스로 구축하는 방법을 제안한다. 나아가서 본 연구의 결과물로 구축된 통합 데이터베이스 기반 취약점 검색용 웹사이트를 개발함으로써, 특정 소프트웨어의 버전과 Windows 운영체제에서의 취약점을 탐지 및 활용하는데 효용성을 보인다.

ABSTRACT

If the software is not updated after the vulnerability is disclosed, it can continue to be attacked. As a result, the importance of N-day detection is increasing as attacks that exploit vulnerabilities increase. However, there is a problem that it is difficult to find specific version information in the published vulnerability database, or that the wrong version or software is outputted. There is also a limitation in that the connection between the published vulnerability databases is not good. In order to overcome these limitations, this paper proposes a method of building information including comprehensive vulnerability information such as CVE, CPE, and Exploit Database into an integrated database. Furthermore, by developing a website for searching for vulnerabilities based on an integrated database built as a result of this study, it is effective in detecting and utilizing vulnerabilities in specific software versions and Windows operating systems.

Keywords: Cybersecurity, Vulnerability, CVE, CPE, Exploit DB

I. 서 론

소프트웨어 취약점은 크게 Zero-day 취약점과 N-day 취약점으로 구분된다. Zero-day 취약점은 아직 알려지지 않은 취약점이며, N-day 취약점은 이미 알려진 취약점이다. Zero-day 취약점은 발견 즉시 악용될 가능성이 높아, 이를 효과적으로 예방하기 위해서는 사전 대비가 필요하다. N-day 취약점은 이미 알려진 취약점이므로, 취약점 패치를 통해 이를 해결할 수 있다. 그러나 취약점 패치를 적용하지 않는 경우, 공격자가 취약점을 악용하여 사이버 공격을 수행할 수 있다. Bullough 외 3명[1]에 의

Received(05. 16. 2024), Modified(06. 12. 2024), Accepted(06. 28. 2024)
* 본 연구는 2024년도 정보통신기획평가원의 지원을 받아 수행하였습니다.(RS-2023-00227165, 무기체계 플랫폼에 적용 가능한 바이너리 기반 SW취약점 자동 탐지·분석 기술 개발
† 주저자, chqh0828@korea.ac.kr
‡ 교신저자, jungheumpark@korea.ac.kr(Corresponding author)

하면 매년 수천 개의 소프트웨어 취약점이 발견되고 보고되며, 취약점이 공개될 시 악용될 확률은 급격히 높아진다고 보고하였다. Bilge 외 1명[2]은 취약점이 공개된 후 이를 악용한 공격의 양이 5배 증가한다는 것을 발견했다. 실제 최근 사례로 2017년에 공개된 취약점 'PHPUnit'의 경우 22년 대비 23년에 약 10배 이벤트가 발생하였다[3]. 따라서, Zero-day 뿐만 아니라 N-day 취약점의 분석과 연구는 사이버 공격에 대한 예방과 대응에 있어 매우 중요하다.

이와 관련하여 실제 데이터를 활용하는 대표적인 체계 중 하나는 MITRE에서 취약점을 관리하는 Common Vulnerabilities and Exposures(이하 CVE)이다[4]. CVE는 취약점을 고유한 ID로 관리하고, 취약점의 특징과 영향을 제공한다. 그러나 CVE 시스템만으로는 취약점을 효과적으로 식별하고 대응하기에는 부족하다. 취약점을 검색할 때 소프트웨어와 버전을 함께 입력하면 모든 버전이 출력되거나 관련 없는 소프트웨어도 함께 나타나기 때문에 취약점을 정확하게 식별하기 한계점이 존재한다.

오픈소스 소프트웨어(OpenSource Software, 이하 OSS)의 활성화, 모듈화, 그리고 재사용 증가는 소프트웨어 패키지를 컴포넌트 중심으로 구성하는 방식을 가능하게 했다. 이러한 변화는 OSS 중심의 글로벌 공급망을 통한 소프트웨어 개발과 운영을 가능하게 했지만, 동시에 보안 취약점 관리와 라이선스 준수, 버전 관리 등 컴포넌트 관리의 중요성을 부각시켰다. 최근 솔라윈즈 해킹 사건과 같은 소프트웨어 공급망 공격이 증가하면서 소프트웨어 공급망 보안을 강화하기 위한 방안으로 Software Bill of Materials(이하 SBOM)이 주목받고 있다[5]. CVE만으로는 알려진 취약점을 대응하기에 부족함이 있으며, 공급망 공격에 따른 대응도 부족하다. 따라서 본 논문에서는 취약점 검색을 보완하며, 공급망 공격과도 연계할 수 있는 Common Platform Enumeration(이하 CPE) 개념을 활용한다[6].

CPE는 소프트웨어, 운영체제, 응용 프로그램 등과 관련된 취약점 정보를 식별하기 위해 정의된 목록이다[7]. CPE와 CVE는 미국 국립표준기술연구원(National Institute of Standards and Technology, 이하 NIST)[8]에서 관리하는 미국 국가 취약점 데이터베이스(National Vulnerability Database, 이하 NVD)[9]에서 활용되고 있다.

하지만 CPE와 CVE만으로 취약점을 구현하거나 연구하는 것에는 어려움이 존재한다. 본 논문에서는 이를 극복하기 위하여 익스플로잇 데이터베이스(Exploit Database, 이하 Exploit-DB)를 연결하였다. Exploit-DB는 익스플로잇(exploit) 코드와 그에 대한 세부 정보를 얻을 수 있는 데이터베이스이다. Exploit-DB에는 취약점 공격 개념 증명 코드(Proof-Of-Concept, 이하 PoC)가 있으며, PoC를 통해 보안 위협을 식별하고 적절한 보안 조치 및 대응의 구현을 수행할 수 있다.

따라서 본 논문에서는 NVD에서 구축되어 있는 CPE와 CVE 그리고 Exploit-DB를 통합한 데이터베이스를 제시한다. 왜냐하면 익스플로잇 코드나 취약점 관련 문서들을 Exploit-DB에서 보완할 수 있기 때문이다. 또한 CVE에 등록되지 않은 취약점도 CPE를 기준으로 식별할 수 있기 때문이다. 그러나 Exploit-DB에는 CVE가 누락되거나 CVE 참고자료의 정보 누락으로 인해 CPE와 CVE의 연결점을 저해하는 요소들이 존재한다. 이를 통합 및 개선한 데이터베이스는 취약점 정보를 찾는 과정을 간소화했고 정확도를 향상시켰다.

2절에서는 본 논문을 위한 배경지식과 연구 주제와 관련된 기존 연구를 살펴보고, 3절에서는 기존 취약점 데이터베이스의 특징 및 한계점을 살펴본다. 이어 4절에서는 각각의 기존 취약점 데이터 수집 과정과 데이터 연동을 하기 위해 부족한 점을 채우기 위한 알고리즘을 살펴보고, 5절에서는 통합된 데이터베이스를 활용하기 위해 구현된 웹페이지를 통해 소프트웨어 검색 예시와 Windows 환경에서 설치된 소프트웨어의 취약점을 자동 탐지한 결과를 확인해 보고, 6절에서 결론 및 향후 발전 방향을 밝히며 논문을 마친다.

II. 배경지식 및 관련 연구

Synopsys 조사에 따르면 96%의 소프트웨어가 오픈소스를 사용하고, 1,000개 이상의 파일로 구성된 소프트웨어는 99%가 오픈소스를 포함하고 있다[10]. 또한 2020년 Open Source Security and Risk Analysis 보고서에서는 조사 대상 1,250개의 상업용 소프트웨어의 99%는 한 개 이상의 오픈소스를 포함하고 있고, 사용된 오픈소스의 82%는 4년 이상 된 구버전으로 조사되었다. 오픈소스를 통한 개발은 취약점을 빠르게 발견하고 패치도 할 수 있다는 장점을 가지고 있으나 기업이나 기관에서 사용하

는 소프트웨어에 어떤 오픈소스가 사용되고 있는지 모른다는 한계가 있다.

이렇듯 오픈소스를 이용한 소프트웨어가 많아지면서 소프트웨어 공급망 관리를 위해 SBOM이 나오게 됐고 SBOM 표준 중 하나인 CPE를 이용한 취약점을 찾는 연구도 나오고 있다.

Ushakov 외 3명[11]은 설치되어 있는 소프트웨어 및 하드웨어의 취약점 탐지를 하기 위해 알려진 취약점의 가장 포괄적인 데이터베이스인 NVD를 이용한다. NVD는 CPE 형식으로 표시되는 소프트웨어 및 하드웨어의 취약점을 저장한다. 설치되어 있는 프로그램으로부터 CPE를 알아내기 위하여 문자열의 유사성을 결정하기 위한 Ratcliff/Obershelp 알고리즘을 이용하였고 평균 79%의 정확도로 CPE를 찾아냈다.

Sanguino 외 2명[12]은 알려진 취약점을 확인하기 위해 Vulnerability Management Systems라는 오픈소스를 만들었고 호스트에 설치된 소프트웨어 목록들을 가지고 와 설치된 버전과 최신 버전을 보여주고 설치된 버전의 CPE, 존재하는 CVE의 개수와 CVE 목록을 확인할 수 있다.

Takahashi 외 3명[13]은 조직 내 보안 유지를 위해선 IT 자산 및 관련 취약점을 관리해야 하지만 자원이 제한되어 있기에 한계가 있다고 한다. 그렇기에 조직의 관리 네트워크 단에서 IT 자산의 취약점을 모니터링하여 취약점을 관리하는 도구를 제안한다. 여기서 도구는 CPE 및 CVE를 이용하여 정보의 식별 및 검색을 용이하게 한다. 도구의 프로세스를 보면 IT 자산 정보를 모은 후 IT 자산에 대해 식별자를 생성 후 식별자를 통해 취약점을 파악하게 된다. 이 과정에서 식별자는 CPE를 뜻하고 CPE와 연계된 CVE를 통해 취약점을 파악하게 된다.

Cheng 외 4명[14]의 연구에 따르면, IoT 장치 제조업체들은 Open-Source Components를 통합하여 펌웨어 개발을 용이하게 하지만, 오래된 버전은 N-day 취약점이 존재한 채로 IoT 장치에서 계속 작동한다. 이러한 N-day 취약점을 찾기 위해 VERI 시스템을 만들어 펌웨어에 존재하는 취약점을 CPE를 통해서 파악하게 된다.

Ecik[15]은 네트워크 기반 취약점 스캔을 하기 위해 능동적인 방법과 수동적인 방법 두 가지로 나누어 제시하고 있다. Active Vulnerability Scanning는 네트워크의 호스트를 내부적으로 스캔하여 악의적인 행위자가 악용할 수 있는 것으로 알려진 결함이나 취약점을 탐지하는 것이다. 반면에 Passive Vulnerability Detection는 네트워크 모니터링을 통한 수동으로 캡처된 데이터 분석과 로그, 구성 파일을 분석하여 제품 및 버전을 관리한다. 마지막으로 CPE를 이용하여 사용 중인 제품과 비교하여 취약점이 있는 제품인지 확인한다. 정확도와 정밀도를 계산하여 여러 개의 OS와 CVE를 분석한 결과 수동적인 방법이 취약점을 찾는 데 더 낫다는 결과를 보여줬다.

취약점 정보를 탐지하는 스캐너는 QualysFreeScan[16], Nexpose Vulnerability Scanner[17], OpenVas[18] 등이 있다. 이 중 가장 널리 알려진 프로그램은 OpenVAS이다. OpenVAS는 IP주소를 지정하여 스캔하는 방식을 가지며, 자산과 취약점 리스트 등을 확인할 수 있다. 수집된 자산들은 CPE를 식별하고 CVE 정보를 보여주는데 활용된다.

이와 같은 연구들은 소프트웨어, 하드웨어, 네트워크 기반 등 여러 가지 취약점을 분석하는 데 있어 CPE를 활용하여 식별한 뒤, 취약점 정보를 제공하는 방식을 가진다. 하지만 취약점을 찾아낸 이후 취약점을 관리하는 측면에서의 연구는 진행되지 않았다. 또한, 기존 공개 취약점 데이터베이스들이 갖는 한계점을 고찰하고 이를 개선하는 관점의 연구는 없었다.

III. 기존 취약점 데이터베이스의 특징 및 한계점 분석

3.1 Common Vulnerabilities and Expousre (CVE)

CVE는 MITRE에서 관리하며 각 항목에는 고유한 CVE 번호, 짧은 요약 및 적어도 하나의 외부 참조가 포함된다[19]. IT 제품에서 발견되는 알려진 취약점에 식별자를 할당하고 취약점에 대한 정보(소프트웨어, 버전 등)를 제공하는 데 사용되는 방법이다. CVE는 컴퓨터 시스템 및 소프트웨어에서 발견된 보안 취약점을 식별하고 추적하기 위한 표준 식별자 체계이고, 광범위한 보안 생태계에서 널리 사용되고 있다. 취약점 보고서, 보안 패치, 보안 솔루션 등 다양한 보안 관련 자료 및 활동에서 CVE 식별 번호가 사용되고 있다. CVE의 설명에 대한 예시로 CVE-2021-26237은 Table 1과 같다[20].

NVD에서는 이 정보를 사용하여 공통 취약점 등

Table 1. CVE-2021-26237 Description

FastStone Image Viewer <= 7.5 is affected by a user mode write access violation at 0x00402d7d, triggered when a user opens or views a malformed CUR file that is mishandled by FSViewer.exe. Attackers could exploit this issue for a Denial of Service (DoS) or possibly to achieve code execution.

급 시스템(Common Vulnerability Scoring System, 이하 CVSS) 점수와 CPE 목록을 추가하여 제공한다. CVE-2021-26237의 상세 정보를 확인하기 위해서는 CVE-2021 Feed[21]를 다운로드 받아 확인해야 한다. 상세 정보는 JSON 형식으로 저장되고 CVE 기본 정보, CWE정보, 참조 URL, CVSS 점수, CPE 관련 정보 등이 key-value 형식으로 저장된다. CVE-2021-26237의 CPE 관련 정보를 예시로 보면, CPE는 "cpe23Uri" key와 "cpe:2.3:a:faststone:image_viewer:*:*:*:*:*:*:*:*" value로 저장된다. 또한 CPE 버전 정보는 7.5 이하를 표현하기 위해서 "versionEndIncluding" key와 "7.5" value 형태로 저장된다. 버전 정보의 key는 저장되는 버전의 범위에 따라서 key 값이 달라진다.

3.2 Exploit Database

NIST에서 관리하고 있는 NVD는 알려진 취약점과 세부 정보를 자세히 기록하고 있지만, 실제로 공격을 수행할 수 있는 익스플로잇 코드는 제공하지 않는다. 그에 반해, Exploit-DB는 exploit 코드와 함께 세부 정보를 제공하는 커뮤니티 기반 데이터베이스이다. 이를 통해 취약점과 관련된 연구나 분석을 진행할 수 있다. 다양한 언어(C/C++, Python, JavaScript, ASM 등)로 작성된 코드와 정보 등을 제출받아 1994년부터 현재까지 지속적으로 업데이트 되고 있다[22]. CVE보다 구체적으로 정보를 얻을 수 있으며 Exploit-DB에서 직접 검증을 한 코드도 존재한다. 모든 코드와 세부 정보는 엑셀 파일 형태로 체계적으로 정리되어 있으며, GitLab을 통해 압축된 형태로 배포되고 있다. 2024년 1월 기준 48,928개의 익스플로잇이 존재하며 PoC를 사용할 수 있다면 유사한 공격을 쉽게 수행할 수 있다[23].

3.3 Common Platform Enumeration (CPE)

CPE는 NIST가 개발 및 유지 관리하고 있으며

Table 2. CPE Attribute

CPE 2.3 Version Format	
cpe:2.3:part:vendor:product:version:update:edition:language:sw_edition:target_sw:target_hw:other	
part	Contains **a** for application, **o** for operating systems, and **h** for hardware devices
vendor	Identifies person or organization which manufactured or created the product
product	Contains official product name
version	Describes a version
update	An update of the product
sw_edition	An edition of the product
target_sw	Defines operating environment
target_hw	specifies an architecture of product
language	Defines language of user interface of product
other	Contains any other information of product specific

일련의 응용 프로그램, 운영체제 및 하드웨어 장치를 설명하고 식별하는 표준화된 방법이다. CPE는 조직의 컴퓨터 리소스를 구성하는 소프트웨어 애플리케이션, 운영체제 및 하드웨어 구성 요소의 다양한 클래스를 식별하고 문서화하기 표준화된 기술로 NIST에서 NVD의 일부로 관리되고 있다.

CPE는 다음과 같은 표준화된 명명 규칙을 가진다. "cpe:2.3:part:vendor:product:version:update:edition:language:sw_edition:target_sw:target_hw:other"

CPE의 주요 속성은 vendor, product, version이다. 이 속성들을 통해 소프트웨어를 신속하고 정확하게 식별할 수 있다[24]. Table 2는 CPE 전체 형식과 각 속성에 대한 설명이다.

3.4 기존 취약점 데이터의 현재 한계점

CVE를 사용하여 소프트웨어 취약점을 확인할 때, 소프트웨어 이름을 검색하면 모든 버전의 취약점을 확인할 수 있다. 그러나 특정 버전에 해당하는 취약점을 알아내기 위해 버전 정보를 함께 입력하면,

Fig 1과 같이 특정 버전이 검색되지 않는 문제가 발생할 수 있다. 예를 들어 2024년 4월 기준 “FastStone Image Viewer 6.5”을 검색 시 CVE-2018-15813, CVE-2018-15814, CVE-2018-15815, CVE-2018-15816, CVE-2018-15817와 같이 5개가 검색된다. 그러나 CVE-2021-26237는 Table 1에서 언급된 것처럼, 7.5 버전 이하의 소프트웨어에도 적용된다. 즉 FastStone Image Viewer 6.5 버전도 CVE-2021-26237이 검색되어야 한다. 하지만 CVE 검색 기능은 CVE-2021-26237를 확인할 수 없으므로 해당 취약점을 놓치게 된다.

이러한 문제점은 Exploit-DB에서도 발견할 수 있다. Exploit ID 49660 “FastStone Image Viewer 7.5 - .cur BITMAPINFOHEADER ′Bit Count′ Stack Based Buffer Overflow (ASLR & DEP Bypass)”의 경우 위와 똑같은 취약점이지만, CVE는 N/A 값으로 존재하지 않는다. 하지만 CVE-2021-26236의 참고자료를 확인해 보면 “https://www.exploit-db.com/exploits/49660”가 존재하고 연관된 취약점임을 확인할 수 있다. CVE와 Exploit DB에서 찾은 결과를 나타낸 그림은 Fig 1과 같다.

취약점 스캐너인 OpenVAS는 설치된 프로그램 목록을 대상으로 하지 않으므로, 동일한 방식으로 비교할 수 없다. OpenVAS는 입력된 IP에 스캔을 수행하여 CPE 2.2 버전 정보를 수집한다. 다음 예시는 수집된 CPE 중 일부에 대한 분석한 결과이다.

수집된 결과 “cpe:/a:openbsd:openssh:7.4”는 “CVE-2021-41617, CVE-2020-14145, CVE-2018-15919“와 같은 3개의 취약점을 확인할 수 있다. 그러나 NVD에 동일한 CPE를 검색한 결과는 총 18개이다. 검색되지 않은 15개의 CVE 중 CVE-2021-36368을 예시로 보면, NVD 데이터에서 CVE-2021-36368은 openssh 8.9 버전까지의 모든 CPE를 포함한다. 그러나 OpenVAS 데이터에서의 CVE-2021-36368은 CPE 1개(cpe:/a:openbsd:openssh:8.8:p1)만 기록되어 있다. 즉 CVE-2021-36368이 8.8버전에서만 발생한 취약점으로 인식될 수 있다. 위의 예시처럼 OpenVAS의 취약점 정보가 다량의 CPE와 제대로 연동되지 않는 경우가 많다.

위와 같이 살펴본 문제들을 종합해 보면, CVE 데이터는 소프트웨어 세부 버전을 구분하여 검색하는 기능을 제공하지 않아서 활용에 한계가 있다. 취약점 스캐너인 OpenVAS에서는 CPE와 CVE가 제대로 연동되지 않은 경우가 발견됐다. 또한, Exploit-DB 같은 커뮤니티 기반의 인프라에서도 CVE가 존재하지만 누락된 경우가 발견됐다. 그로 인해 CVE와 제대로 통합되지 않은 데이터가 있다는 것을 확인할 수 있었다.

IV. N-day 대응 통합데이터베이스 설계 및 구축

3절에서 살펴본 문제와 같이 CVE 또는 Exploit-DB를 단일로 사용하는 경우, 소프트

웨어 취약점을 명확히 확인할 수 없는 한계점이 존재한다. 이런 한계점을 극복하기 위하여 CPE를 기준으로 통합된 데이터베이스 Vulnerability Knowledge Base for Security and Forensics(이하 Vulbase)를 제시한다.

Vulbase의 주요 데이터인 CPE-Site[25], CVE-Site[26], Exploit-DB-Site[27]에서 수집되었다. 2024년 1월 기준으로 CPE는 XML 형태로 1,203,302개가 존재하였고 CVE는 CSV 형태로 296,795개가 존재하였다. 마지막으로 Exploit-DB 정

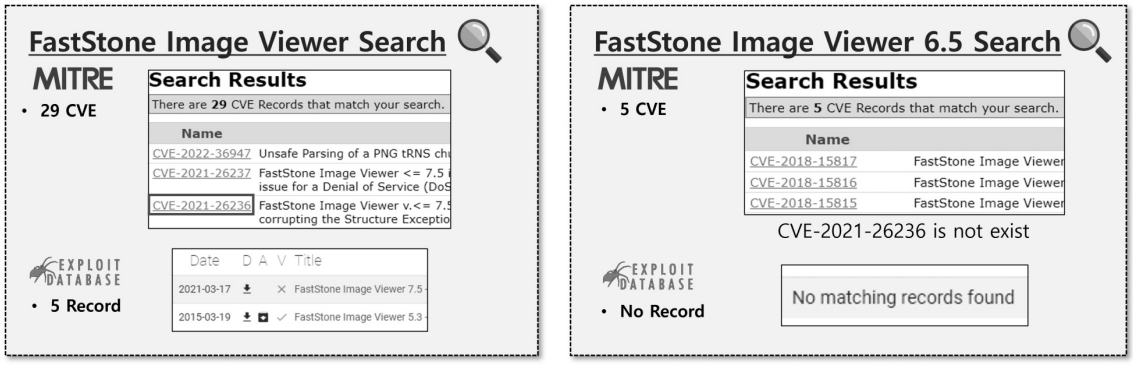


Fig. 1. FastStone Image Viewer Search Result

보는 CSV 형태로 48,928개가 수집되었다.

4.1 기존 공개 데이터 수집

기존 공개 데이터 수집은 Fig 2의 Collect 과정과 같이 이루어졌다.

CPE Site에 접속하면 매일 업데이트된 CPE 데이터를 다운로드할 수 있다. 가장 최신 파일인 2.3 버전의 Zip 파일을 다운로드 및 압축 해제하여, XML 파일을 확인한다.

CVE Site에서 CVE 데이터를 CSV, HTML, Text, XML 확장자로 다운로드할 수 있다. 데이터베이스 변환에 용이한 CSV 파일을 다운로드한다.

Exploit-DB Site는 Exploit-DB를 검색과는 별개의 사이트로 운용되고 있다. Exploit-DB site는 Exploit-DB의 Raw 데이터를 CSV 형태의 3개 파일을 다운로드할 수 있다.

4.2 데이터 전처리 및 통합

Vulbase는 Exploit_Data, CVE_Data, 그리고 Vulnerabilities_info 세 가지 테이블이 존재하며, 데이터를 획득하여 통합하는 방법은 Fig 2와 같다.

CPE-Site를 통해 다운받은 XML 파일에는 name, references, cpe 요소가 존재한다. 이 중 cpe 정보만 추출하여 Vulnerabilities_info 테이블의 CPE에 추가한다.

CVE-Site을 통해 다운받은 CSV 파일에는 Name, Status, Description, References, Phase, Votes, Comments 요소가 존재한다. 모든 정보를 사용하기 위하여 CVE_Data에 추가한다.

Exploit-DB Site를 통해 다운받은 3개의 CSV 파일에는 ID, file, description, date_published, author, type, platform, port, size, language, date_added, date_updated, verified, codes, tags, aliases, screenshot_url, application_url, source_url 요소가 존재한다. 모든 정보를 사용하기 위하여 Exploit_Data에 추가한다.

Vulnerabilities_info테이블의 CVE 데이터를 추가하는 과정은 NVD에 HTTP요청을 통해 CPE에 해당하는 CVE를 가져온다.

마지막으로 Exploit_DB_index 데이터를 수집하기 위해서는 Exploit_Data 테이블의 Codes 열과 Vulnerabilities_info 테이블의 CVE 열을 비교해야 한다. 이후 두 열이 동일한 경우에만 Exploit_Data의 ID 값을 Vulnerabilites_info 테이블의 Exploit_DB_index 열에 추가한다.

Vulbase는 CPE를 기준으로 CVE의 존재 여부, Exploit ID의 존재 여부를 나타낸다. Fig 3의 초기 상태(Initial state)는 데이터를 단순히 전처리하고 통합한 결과이다. 초기 상태는 CPE를 기준으로 총 1,203,302개의 레코드로 이루어져 있다. CVE, Exploit ID가 모두 존재하는 경우는 166,843개, CVE만 존재하는 경우는 795,787개, CVE와 Exploit ID가 존재하지 않는 경우는 240,672개이다.

4.3 누락 정보 추가 및 관련 정보 연계

연동하는 과정에서 누락된 정보가 있는 두 가지 경우를 발견하였다. Exploit-DB에서 CVE 정보가

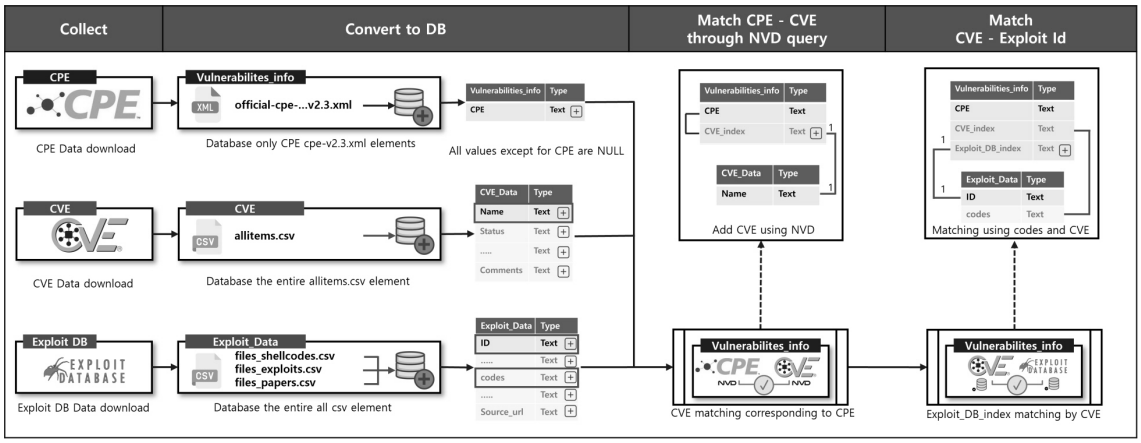
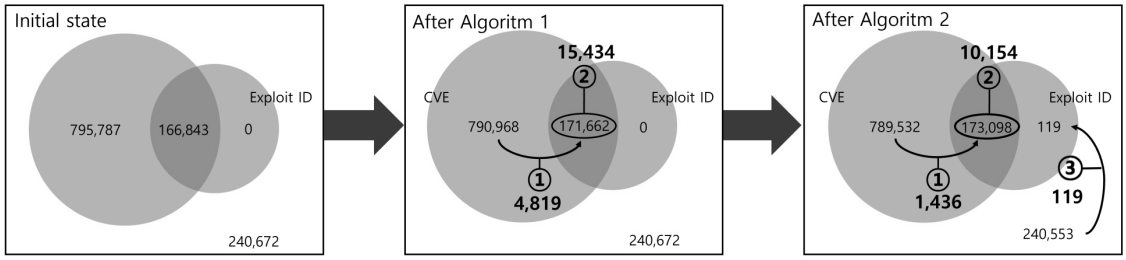


Fig. 2. Data Collection Method



	Name	Example		
		CPE	CVE	Exploit ID
①	New Exploit ID With CVE	cpe:2.3:o:dlink:dph-400se_firmware:2.2.15.8:*:*:*:*:*:*:*	CVE-2023-43960	NULL → 51709
②	Add Exploit ID	cpe:2.3:a:b2evolution:b2evolution:4.1.7:*:*:*:*:*:*:*	CVE-2022-30935, CVE-2020-22841, ...	49551 → 49551, 49554
③	New Exploit ID Without CVE	cpe:2.3:a:f5:big-ip:9.4.3:*:*:*:*:*:*:*	NULL	NULL → 31364

Fig. 3. Vulbase Distribution of Algorithm Stages

누락된 경우와 CVE 참고자료에 Exploit-DB와 관련 데이터가 없는 경우이다. 이를 개선하기 위해 CVE 참고자료를 이용한 알고리즘 1과 Exploit-DB 타이틀을 토큰화한 알고리즘 2를 적용한 결과, Fig 3의 After Algoritm 2와 같은 분포를 나타냈다.

결론적으로 Vulnerabilities_info 테이블의 모든 알고리즘을 적용한 레코드 개수는 CPE 수집 시점을 기준으로 1,203,302개이다. CPE 중에서 CVE만 존재하는 경우는 789,532개이며, Exploit ID만 존재하는 경우는 119개이다. 마지막으로 CVE와 Exploit ID가 동시에 존재하는 경우는 173,098개이다.

4.3.1 누락 정보 추가 알고리즘 1: CVE 참고자료를 이용한 연동

첫 번째 문제점은 Exploit-DB에서 제공되는 CVE 정보가 누락된 경우이다. 이를 해결하기 위하여 CVE 참고자료(reference)를 확인하는 방식을 제시한다. CVE 참고자료는 CVE-Site에서 취약점과 관련된 정보들의 링크들을 제공한다. 이 중 Exploit-DB와 연동된 링크들도 포함되어 있으며, 링크가 존재하는 경우는 "EXPLOIT-DB"라는 키워드를 표기하고 있다. Fig 4는 첫 번째 알고리즘을 나타낸 것이다. 단순 통합된 Vulbase의 CVE_Data 테이블의 references 열을 전수조사하여 "EXPLOIT-DB" 키워드와 링크들을 파싱한다. 그리고 Exploit_Data 테이블 Codes 열의 CVE 정보와 일치 여부를 판단한다. 만약 두 정보가 일치하지 않는 경우, CVE 정보를 Exploit_Data 테이블의 Codes 열과 Vulnerabilities_info 테이블의 Exploit_DB_index 열에 추가한다.

첫 번째 알고리즘의 예시는 다음과 같다. CVE-2023-43960의 참고자료에 "MISC:https://www.exploirt-db.com/exploits/51709"라는 정보가 기록되어 있다. 그러나 Exploit-DB의 Exploit ID 51709에는 CVE 정보가 빠져있다. 따라서 Vulnerabilities_info 테이블의 CVE 열에 CVE-2023-43960이 존재하지만, Exploit_DB_index에 51709가 기록되지 않았다.

첫 번째 알고리즘의 결과 Exploit-DB에서 누락된 CVE를 727개를 식별할 수 있었다. 그러나 식별된 727개 중 118개의 경우는 Vulbase에 추가할 수 없었다. 왜냐하면 118개의 경우는 현재 Exploit-DB에서 접근할 수 없기 때문이다. 예를 들어 Exploit ID 1658은 CVE-2006-1747에 해당하지만 현재 Exploit-DB에 데이터가 없고 Exploit-Site에서도 접근할 수 없다. 이러한 경우를 제외한 609

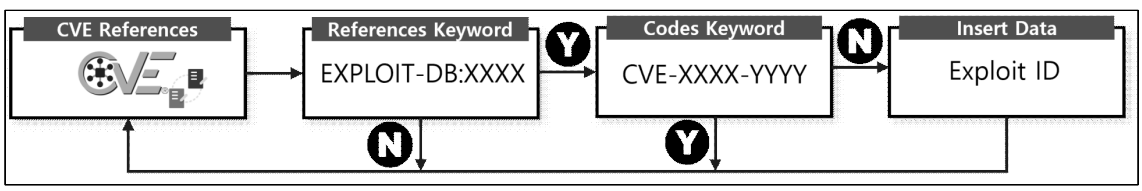


Fig. 4. Data consolidation with CVE References

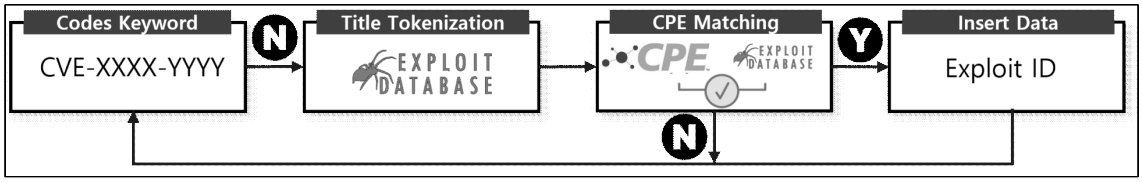


Fig. 5. Data consolidation with Exploit Database Title Tokenization

개의 누락된 CVE는 Vulbase 전체 레코드 중 20,253개의 레코드를 갱신하였다.

Exploit ID 갱신 작업은 Fig 3의 표에서 언급된 "New Exploit ID With CVE"와 "Add Exploit ID"로 총 두 가지 방식이다. 첫 번째 방식은 Exploit ID가 존재하지 않았지만, CVE 참고자료에서 기존에는 확인하지 못했던 Exploit ID를 찾아서 저장하는 방식이다. 두 번째는 Exploit ID가 CVE 참고자료보다 부족하여, 누락된 Exploit ID를 찾아서 추가하는 방식이다. 갱신된 20,253개의 레코드를 조사한 결과, "Add Exploit ID" 방식으로 저장된 경우는 총 15,434개이고 "New Exploit ID With CVE" 방식으로 추가된 경우로 총 4,819개이다.

4.3.2 누락 정보 추가 알고리즘 2: Exploit-DB 타이틀 토큰화을 이용한 연동

두 번째 문제점은 Exploit-DB에서 제공되는 CVE 정보가 누락되는 동시에 CVE 참고자료가 없어 연동할 수 없는 경우이다. 이를 해결하기 위하여 Exploit-DB 타이틀과 CPE를 토큰화하여 연동하는 방식을 제시한다. Exploit-DB 타이틀은 일반적으로 〔벤더사〕〔소프트웨어〕〔버전〕 - 〔취약점〕 형식으로 이루어져 있다. 마찬가지로 CPE도 벤더사, 소프트웨어, 버전의 정보가 포함되어있다. Fig 5는 두 번째 알고리즘을 나타낸 것이다. 첫 번째 문제점이 해결된 Vulbase의 Exploit_Data 테이블 Codes 열을 전수조사하여 CVE가 존재하는지 판단한다. 만약 CVE 존재하지 않는다면 Exploit-DB 타이틀을 토큰화한다.

첫 번째 단어를 사용하여 관련 CPE를 식별 및 토큰화한 후, 토큰화된 타이틀과 일치하는 단어의 개수가 세 개 이상인지 여부를 판단한다. 일치된 단어의 개수 기준을 두 개로 설정할 경우 오탐 가능성이 높고, 네 개 이상의 단어로 진행할 경우 유효한 결과를 얻기 어렵다는 것을 발견했다.

두 개의 단어가 동일한 것을 파악했을 때 예시는 다음과 같다. 알고리즘 2를 실행시켜 나온 결과물 중 하나는 Exploit ID 48237로 타이틀은 "Google Chrome 80.0.3987.87 - Heap-Corruption Remote Denial of Service (PoC)"이다. 이 타이틀과 두 개 단어가 동일한 CPE는 "cpe:2.3:a:google:chrome:0.1.38.1:*:*:*:*:*:*:*"가 나올 수 있지만 버전 정보가 전혀 다른것을 확인할 수 있다. 이외에도 Google Chrome의 모든 버전의 CPE가 출력되어 다수의 오탐이 발생한다. 두 개 단어를 기준으로 알고리즘 2를 실행하면 총 4,081개의 Exploit-DB에서 42,737개의 CPE를 획득할 수 있었다.

네 개의 단어가 동일한 것을 파악했을 때 예시는 다음과 같다. 알고리즘을 실행시켜 나온 결과물 중 하나는 Exploit ID 41485로 타이틀은 "WordPress Plugin Popup by Supsystic 1.7.6 - Cross-Site Request Forgery"이다. 이 타이틀과 네 개의 단어가 동일한 CPE는 "cpe:2.3:a:supsystic:popup:1.7.6:*:*:*:*:wordpress:*:*"가 확인되었다. 동일한 단어는 Wordpress, supsystic, popup, 1.7.6로 버전, 소프트웨어명, 벤더사 이외에 타겟 소프트웨어, 플러그인 같은 부가적인 정보가 동일해야 한다. 네 개 단어를 기준으로 알고리즘 2를 실행하면 총 9개의 Exploit-DB에서 9개의 CPE를 획득할 수 있었다.

따라서 알고리즘 2는 3개 단어를 기준으로 진행되어, Vulnerabilities_info 테이블의 Exploit_DB_index 열에 추가한다. 알고리즘 2의 예시는 다음과 같다. 알고리즘2를 실행시켜 나온 결과물 중 하나는 Exploit ID 31364로 타이틀은 "F5 BIG-IP 9.4.3 - Web Management Interface Console HTML Injection"이다. 이 타이틀과 세 개의 단어가 동일한 CPE는 "cpe:2.3:a:f5:big-ip:9.4.3:*:*:*:*:*:*:*"이다. 두 개를 비교해보면 동일한 소프트웨어임을 알 수 있다. 그러나 Exploit-DB 상세 정보에는 Exploit-DB 타이틀에 비해 버전를 보다 구체적으로 명시하고 있는 경우가 존재한다. 따라서 단순히 타이틀과 CPE 비교뿐만 아니라, Exploit-DB

상세 정보까지 확인하여 버전 정보를 명확하게 식별하여 추출한다. Exploit-DB 상세 정보와 비교하는 경우는 크게 세 가지로 분류된다.

첫 번째는 Exploit-DB 상세 설명에 CVE ID가 존재하는 경우이다. "CVE-X-Y" 키워드를 찾아내는 경우 Exploit-DB Codes에 등록되지 않은 정보를 확인할 수 있다. 이 경우는 Vulnerabilities_info의 CVE_index의 값과 동일하다면, 해당 Exploit ID 값을 모두 Exploit_DB_index에 대입한다.

두 번째는 Exploit-DB 타이틀에 부등호가 존재하는 경우이다. 해당하는 버전 정보들에 Exploit ID 값을 모두 Exploit_DB_index에 대입한다.

세 번째는 Exploit-DB 상세 설명에 "Version: 〈X" 키워드가 있는 경우이다. 해당하는 버전 정보들에 Exploit ID 값을 모두 Exploit_DB_index에 대입한다. 이 세 가지 경우를 제외하는 경우는 3개의 단어가 동일한 경우로 출력된 CPE에 Exploit ID 값을 대입한다.

두 번째 알고리즘의 결과 3개의 단어가 일치하는 CPE를 총 639개 식별할 수 있었다. 639개의 식별된 CPE는 Vulbase의 전체 레코드 중 11,590개를 갱신하였다.

Exploit ID 갱신 작업은 4.3.1절에서 설명한 "New Exploit ID With CVE", "Add Exploit ID" 방법과 "New Exploit ID Without CVE"로 총 세 가지 방식이다. 세 번째 방식은 Exploit ID와 CVE가 모두 존재하지 않아 연결점을 찾을 수 없었던 Exploit ID를 찾아서 저장하는 방식이다. 갱신된 11,590개의 레코드를 조사한 결과, "Add Exploit ID" 방식으로 저장된 경우는 총 10,154개이고 "New Exploit ID With CVE" 방식으로 추가된 경우는 총 1,436개이다. 마지막 세 번째 방식인 "New Exploit ID Without CVE" 방식으로 추가된 경우는 총 119개이다.

V. 구현 및 활용

본 논문에서는 4절에서 소개한 알고리즘을 통해 Vulbase를 생성하였다. 데이터베이스를 활용하기 위해 Fig 6과 같이 홈페이지를 제작하였다.

개발된 홈페이지는 "CPE 검색 기능"과 "자동 CPE 추출" 기능을 지원한다. "CPE 검색 기능"은 사용자가 프로그램 이름과 버전을 입력하면 해당 프로그램에 관련된 CPE 정보와 취약점 정보를 표시한다. 취약점이 발견되면 최신 버전이 아닌 경우, 업데이트를 권장하는 문구가 함께 표시된다. 프로그램명으로 image viewer와 버전 6.5를 입력할 시 faststone의 image viewer 버전 6.5 CPE와 Exploit ID 정보 그리고 CVE 정보, 최신 버전 정보를 Fig 7과 같이 확인할 수 있다. 이때 CVE와 Exploit ID가 있는 경우, 최신 버전이 아닐 시 최신 버전으로 업데이트 권고 구문이 나오게 된다. "자동 CPE 추출"은 유틸리티인 Winapps를 활용하여 Windows 10에 설치된 프로그램을 스캔한 결과물을 파싱하여 CPE와 취약점 정보를 제공하고 있다. 웹사이트를 통해 설치된 프로그램 목록을 수집하는 유틸리티를 배포하여, 이를 통해 소프트웨어 정보를 획득할 수 있다. Fig 8는 Windows 10 64bit에서 유틸리티를 이용하여 설치된 소프트웨어의 예시이다.

Fig. 6. Vulbase Utilization Website

Fig. 7. image viewer 6.5 search results

```
InstalledApplication(name='7-Zip 19.00 (x64)', version='19.00', inst
InstalledApplication(name='Mozilla Firefox 84.0.2 (x64 en-US)', vers
InstalledApplication(name='Mozilla Maintenance Service', version='84
InstalledApplication(name='QEMU', version='8.0.0', install_date=None
InstalledApplication(name='Remote Desktop Manager', version='2023.2.
InstalledApplication(name='VMware Tools', version='11.3.5.18557794',
InstalledApplication(name='ownCloud', version='2.9.0.5150', install_
InstalledApplication(name='FreeSWITCH (64 bit)', version='1.8.4', in
InstalledApplication(name='freeSSHd 1.3.1', version=None, install_da
InstalledApplication(name='ALSee', version='5.3', install_date=None,
InstalledApplication(name='Easy Chat Server 2.5', version=None, inst
InstalledApplication(name='FastStone Image Viewer 6.5', version='6.5
InstalledApplication(name='FileZilla Client 3.17.0', version='3.17.0
InstalledApplication(name='Chrome', version='86.0.4240.75', install_
```

Fig. 8. Example of Installed Software

프로그램명, 버전, 설치된 날짜 등 정보를 확인할 수 있다. 이렇게 확인된 소프트웨어 정보를 텍스트 파일로 업로드하면 name과 version 정보를 파싱하여 CPE, Exploit ID 값 그리고 CVE 값들을 제공한다. Fig 9는 유틸리티를 통해 설치된 소프트웨어를 자동 탐지한 후 CPE, Exploit ID, CVE를 확인한 결과이다.

단일 데이터베이스 CVE, Exploit-DB, CPE를 검색한 결과와 Vulbase를 활용하여 설치된 소프트웨어의 취약점을 탐색한 후 Vulbase를 통한 취약점 검색을 통해 발견되는 취약점의 차이가 있는지 확인하였다.

Fig 8과 같이 유틸리티를 통해서 36개의 프로그램의 정보를 파싱한 후, 알고리즘 1과2를 적용하기 전 Vulbase와 알고리즘 1과2를 적용한 Vulbase를 비교하여 차이점이 있는지 확인하였다. 이 과정에서

Fig. 9. Identifying Vulnerabilities Using Utility

추가로 발견된 취약점이 있는 소프트웨어는 Image Viewer, Firefox, Freeswitch, Acrobat Reader, Easy Chat Server로 총 5개로 나타났다.

Table 3는 각 데이터베이스의 검색 결과와 Vulbase를 통해서 찾게 된 취약점, 새로운 취약점을 요약한 표이다. Single DB search results 열의 CVE 검색은 MITRE의 Search CVE List[28] 웹페이지에서 진행하였다. CVE열의 숫자는 Program과 Version을 함께 검색하여 얻은 취약점 개수를 나타낸다. 검색 결과를 비교해 보면, Freeswitch와 Firefox의 취약점 개수가 크게 차이나는 것을 확인할 수 있다. 이러한 차이의 원인을 조사한 결과, Freeswitch 1.8.4를 검색한 경우는 실제 Freeswitch와 관련이 없는 CVE가 검색됐다. 이는 검색어인 "1.8.4"가 다른 프로그램의 취약점 검색 결과에 영향을 준 것으로 확인하였다. 또한, Firefox의 경우에도 Firefox 84.0.2를 검색하면 3,181건의 CVE가 검색되었는데, 이는 모든 Firefox 관련 키워드를 포함하는 것으로 확인하였다.

결론적으로 CVE 개수가 많은 것은 "프로그램 이름", "버전"이 검색 결과에 영향을 미치기 때문이다. 하지만 Vulbase는 프로그램 이름과 버전을 명확하게 확인하여 정확한 취약점 정보를 출력해준다.

Single DB search results 열의 CPE는 Search CPE[29] 웹페이지에서 진행되었고, 숫자는 Program과 Version을 함께 검색하여 얻은 취약점의 개수를 나타낸다. Single DB search results 열의 ID는 Exploit-DB[30] 사이트에서 동일하게 확인한 결과의 Exploit ID 개수를 나타낸다. Vulbase는 통합된 데이터베이스에서 확인할 수 있는 취약점의 개수를 나타낸다.

Vulbase를 통해 Exploit-DB에서 찾지 못한 취약점 정보들을 Table 3과 같이 발견할 수 있었다. 발견한 취약점 정보는 New Exploit ID 열에서 Exploit ID를 보여준다. New Exploit ID가 연동되는 CVE가 있는 경우는 CVSS도 확인하여 위험도를 같이 표기하였다. CVSS는 소프트웨어의 취약점의 특성과 심각도를 파악하는 데 도움이 되는 프레임워크이다. CVSS 3.x는 15년 6월에 발표되어 CVSS 2.0보다 정확한 취약점을 반영하기 위해 여러 가지 변경 사항이 도입됐다. CVSS 3.x로 바뀌기 전의 취약점과 관련된 요소들은 CVSS 2.0으로 그 위험성을 표기하였다. 기존 방법으로는 CVSS 2.0 또는 CVSS 3.x와 관련된 위험도가 높은 알려진 취

Table 3. Additional discovered Exploit ID

Vendor	Program	Version	Single DB search			Vulbase	New Exploit ID	CVSS
			CVE	ID*	CPE			
Faststone	Image Viewer	6.5	5	0	11	12	49660 (CVE-2021-26236)	7.8*** High
Freeswitch	Freeswitch	1.8.4	127	0	9	10	47698 (CVE-2019-19492)	9.8*** Critical
Adobe	Acrobat reader	7.1.0	1	0	142	144	34528	N/A
							34603	N/A
Mozilla	Firefox	84.0.2	3,181	0	484	489	6690 (CVE-2008-5697)	4.3** Medium
							8922 (CVE-2009-2011)	9.3** High
							29573 (CVE-2007-0896)	4.3** Medium
							30285 (CVE-2007-3670)	4.3** Medium
							49892	N/A
Echat server	Easy Chat Server	2.5	23	0	3	6	42153 (CVE-2017-9557)	7.5*** High
							42154 (CVE-2017-9543)	7.5*** High
							42155 (CVE-2017-9544)	9.8*** Critical

*: Exploit-DB ID **: CVSS 2.0 ***: CVSS 3.x

약점들을 발견하지 못한다. New Exploit ID는 CVE와 Exploit ID를 나타낸다. 그러나 CVE가 연동되지 않는 경우는 Exploit ID만 보여준다. CVE가 연동되지 않기 때문에 위험도를 확인할 수 없지만, CVE가 존재하지 않는다는 것은 CPE 혹은 CVE 기반 탐지 방법으로는 해당 취약점으로 탐지할 수 없다는 것과 같다.

그러나 Vulbase를 통해 이전에 찾아내지 못한 새로운 취약점 정보를 찾아내고, 더 나아가 CVE가 할당되지 않은 취약점 정보도 확인할 수 있다. 이를 통해 새로운 취약점을 명확히 발견할 수 있음을 확인하였다. 발견된 CVE의 CVSS 3.x 모두 High와 Critical이므로 높은 위험도를 가지고 있는 것을 확인할 수 있다. CVE가 존재하지 않는 Adobe의 Acrobat reader나 Mozilla의 Firefox 같이 사용자가 많은 프로그램이 CVE가 존재하지 않은 채로 취약점이 존재하는 것 또한 확인할 수 있다.

VI. 결 론

단일 데이터베이스를 통한 취약점 식별에 한계점이 존재하는 것을 확인하였다. Vulbase 통합 과정에서 CVSS 점수가 높은 취약점 정보와 CVE에 존재하지 않는 취약점 정보 획득이 가능하였다. 따라서 다양한 취약점 데이터베이스를 통합하면 누락된 정보를 식별하여 취약점을 더 쉽게 식별 할 수 있다는 사실을 확인하였다. Vulbase를 활용하기 위한 웹사이트는 기능을 더 강화하여 누구나 활용할 수 있도록 공개할 예정이다.

하지만 기존 취약점 검색 방법으로도 취약점 데이터베이스를 반복적으로 수동 분석을하면 유사한 결과를 얻을 수 있다. 또한 AI를 활용해 취약점 정보들을 통합 및 검색하는 방법으로도 해결이 가능하다. 그러나 취약점 검색 과정을 간소화하고 공개된 취약점 정보들을 수집 및 통합, 검증하여 정확성과 활용성을 높이는 것은 의미가 있다고 판단된다.

본 논문에서는 취약점을 실험하고 분석하기 위해,

CVE의 가장 많이 레퍼런싱되며 현재까지 많은 활동이 있는 Exploit-DB를 활용했다. 하지만 N-day 취약점은 업데이트가 되지 않은 상황에서도 발생할 수 있다. 따라서 발견된지 오래된 N-day라 하더라도 취약점 실험 및 분석에는 의미가 있을 수 있다. 또한, 취약점 관련 정보를 제공하는 많은 사이트 혹은 데이터베이스인 Open Source Vulnerability Database[31], Securitytracker[32], Zerodium[33], Vuldb[34] 등이 있다. 같은 취약점을 분석하더라도 각각의 사이트 혹은 데이터베이스에 저장되는 취약점 관련 정보의 형태와 내용은 다양하다. 이러한 정보들을 통합하여 활용한다면 취약점을 파악하고 방어하는데 더욱 효과적일 것이다.

Exploit-DB와 CPE 매칭 과정에서 토큰화 방식을 사용하기 때문에 정보 손실 및 오탐 가능성이 존재한다. 이러한 문제를 해결하기 위해 타이틀 외의 세부 정보를 추가 분석하여 포함하거나 AI 기법을 활용한다면 오탐율을 크게 줄이는게 가능한 것으로 기대된다. 또한, CPE는 현재까지 취약점 탐지에 사용되고 있으며 SBOM과 연계하여 보안 패치 관리와 공급망 보안 강화 등에 적용할 수 있다. 이를 통해 취약점 탐지, 보안 관리 등의 효율성을 크게 향상시킬 것으로 기대된다.

References

[1] Bullough, B. L., Yanchenko, A. K., Smith, C. L., & Zipkin, J. R. (2017, March). "Predicting exploitation of disclosed software vulnerabilities using open-source data," In Proceedings of the 3rd ACM on International Workshop on Security and Privacy Analytics pp. 45-53, Mar. 2017.

[2] L. Bilge and T. Dumitra¸s. "Before we knew it: an empirical study of zero-day attacks in the real world," In Proceedings of the 2012 ACM Conference on Computer and Communications Security, pp. 833 - 844, Oct. 2012.

[3] 2024 SK shieldus EQST Annual Report, pp. 11, Dec. 2023

[4] CVE, "CVE" https://cve.mitre.org/, accessed Apr. 2024

[5] MoneyToday, "SBOM" https://news.mt.co.kr/mtview.php?no=2023060813394653791/, accessed Apr. 2024

[6] Scribe, "CPE" https://scribesecurity.com/ko/sbom/standard-formats/#spdx-sbom-standard-format/, accessed Apr. 2024

[7] Google Patents, "CPE" https://patents.google.com/patent/KR20180097885A/ko/, accessed Apr. 2024

[8] NIST, "NIST" https://www.nist.gov/, accessed Apr. 2024

[9] NVD, "NVD" https://nvd.nist.gov/, accessed Apr. 2024

[10] Datanet, "OpenSource" http://www.datanet.co.kr/news/articleView.html?idxno=151523/, accessed Apr. 2024

[11] Ushakov, R., Doynikova, E., Novikova, E., & Kotenko, I. (2021, September). "CPE and CVE based technique for software security risk assessment," In 2021 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS) (Vol. 1, pp. 353-356, Sep. 2021)

[12] Sanguino, Luis Alberto Benthin, and Rafael Uetz. "Software vulnerability analysis using CPE and CVE." arXiv preprint arXiv:1705.05347, May. 2017.

[13] Takahashi, Takeshi, Daisuke Miyamoto, and Koji Nakao. "Toward automated vulnerability monitoring using open information and standardized tools," 2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops). IEEE, Mar. 2016.

[14] Cheng, Y., Yang, S., Lang, Z., Shi, Z., & Sun, L. (2023). "VERI: a large-scale open-source components

vulnerability detection in iot firmware," Computers & Security, Vol 126, 103068, Mar. 2023
〔15〕 Ecik, Harun. "Comparison of active vulnerability scanning vs. passive vulnerability detection," 2021 International Conference on Information Security and Cryptology (ISCTURKEY). IEEE, Dec. 2021.
〔16〕 Qualys, "QualysFreeScan" https://www.qualys.com/community-edition/, accessed May. 2024
〔17〕 RAPID7, Nexpose Vulnerability Scanner" "https://www.rapid7.com/products/nexpose/, accessed May. 2024
〔18〕 GreenBone "OpenVas" https://www.greenbone.net/en/, accessed Apr. 2024
〔19〕 CPE, "CPE" https://cpe.mitre.org/specification/, accessed Apr. 2024
〔20〕 NVD, "CVE-2021-26237" https://nvd.nist.gov/vuln/detail/CVE-2021-26237/, accessed Apr. 2024
〔21〕 NVD, "Data Feeds" https://nvd.nist.gov/vuln/data-feeds/
〔22〕 Perrone, G., Romano, S. P., d'Ambrosio, N., & Pacchiano, V. "Unleashing Exploit-Db Data for the Automated Exploitation of Intentionally Vulnerable Docker Containers," Available at SSRN 4779063, Mar. 2024
〔23〕 Yang, H., Park, S., Yim, K., & Lee, M. (2020). "Better not to use vulnerability's reference for exploitability prediction," Applied Sciences, 10(7), 2555, Mar. 2020
〔24〕 ReadtheDocs, "CPE" https://cpe.readthedocs.io/en/master/model/cpehierarchy/cpe2_3_fs.html, accessed Apr. 2024
〔25〕 NVD, "CPE" https://nvd.nist.gov/products/cpe/, accessed Apr. 2024
〔26〕 MITRE, "Download" https://cve.mitre.org/data/downloads/index, accessed Apr. 2024
〔27〕 GitLab, "Exploit-Database" https://gitlab.com/exploit-database/, accessed Apr. 2024
〔28〕 MITRE, "CVE Search" https://cve.mitre.org/cve/search_cve_list, accessed Apr. 2024
〔29〕 NVD, "CPE Search" https://nvd.nist.gov/products/cpe/search/, accessed Apr. 2024
〔30〕 Exploit Database, "Exploit Database" https://www.exploit-db.com/, accessed Apr. 2024
〔31〕 Security Affairs, "OSVDB" https://securityaffairs.com/46129/security/osvdb-shuts-down.html/, accessed May. 2024
〔32〕 X, "SecurityTracker" https://twitter.com/securitytracker/, accessed May. 2024
〔33〕 ZeroDium, "ZeroDium" https://zerodium.com/, accessed May. 2024
〔34〕 Vuldb, "Vuldb" https://vuldb.com/, accessed May. 2024

〈저 자 소 개〉

정 종 선 (JongSeon Jeong) 정회원
2021년 2월: 세종대학교 전자정보공학대학 컴퓨터공학과 공학사
2022년 9월~현재: 고려대학교 일반대학원 석사과정
〈관심분야〉 디지털포렌식, 침해사고대응, 역공학

박 정 흠 (Jungheum Park) 정회원
2014년 2월: 고려대학교 정보보호대학원 공학박사
2015년 1월~2019년 2월: 미국 국립표준기술연구원(NIST) 방문연구원
2019년 3월~2021년 8월: 고려대학교 정보보호연구원 연구교수
2021년 9월~2022년 2월: 고려대학교 정보보호대학원 조교수
2022년 3월~현재: 고려대학교 정보보호대학원 부교수
〈관심분야〉 디지털포렌식, 사이버범죄대응, 침해사고대응