L-019

脆弱性に基づいたサイバー演習環境の構築に関する検討

A semi-automatic infrastructure construction for cyber exercise based on public Information of Vulnerability

真島 匠海† 小早川 倫広†
Takumi Majima Michihiro Kobayakawa

1. 背景・目的

近年、オープンソースソフトウェア（OSS）の利用が企業において急速に増加しており、それに伴い OSS の脆弱性を狙ったサイバー攻撃が増えている[1]。サイバー攻撃の被害が増加することで、企業や組織のシステムが攻撃対象となり、個人情報や機密情報の漏洩などのリスクが高まっている。このような状況下で、高度化するサイバー攻撃に対応するためには、高度な情報セキュリティ技術者の育成が課題となっている。Lukas Sadlek らは、公開情報と資産に関するデータを用いて、サイバースペースにおける脆弱性と脅威を特定するための現在の課題について研究している[2]。

情報セキュリティ技術者育成のためには、実践的な演習環境が重要である[1]。しかしながら、様々な脆弱性を含んだシステムを演習として利用できる環境は限られており、またその構築には手間とコストがかかるという問題が存在する。そのため、脆弱性情報が発表されると同時に、その脆弱性を含むサイバー演習環境が自動的に構築できることで演習用・検証用として利用することが可能となる。本研究では、公開されている脆弱性情報を活用し、情報セキュリティ技術者育成のための演習環境を自動的に構築するための設定ファイルの自動生成手法を提案する。具体的には、脆弱性情報から必要な情報を抽出し、それに基づいて脆弱なシステムの設定ファイルを自動的に生成し、これにより、効率的かつリアルな演習環境を提供することで演習用・検証用として活用できることを目的とする。

2. 要件定義

日々更新される脆弱性に対応するためにはユーザのスキルアップが必須となる。そのためには、脆弱性情報を収集し演習者の要求に合わせた脆弱性を脆弱なシステムとして構築することで効率的に演習ができる。脆弱性情報に基づいてサイバー演習環境に必要な OS、ミドルウェア、アプリケーションソフトウェアを自動選択し、サイバー演習環境を半自動で構築することが必要である。

そこで、脆弱性情報に基づいたサイバー演習環境の構築に関する要件を定義する。

- 脆弱性情報を自動収集できること
- ユーザが要求する脆弱性情報を複数取得できること
- 必要な OS、ミドルウェア、アプリケーションソフトウェアを自動選択できること
- 収集した脆弱性情報を管理できること
- 脆弱性情報を基に仮想マシン設定ファイルを自動生成できること
- 様々な脆弱性を組み合わせることができること
- 仮想マシン設定ファイルを基にサイバー演習環境の自動構築ができること

3. 設計・実装

脆弱性を含むシステムを構築するためには、いくつかのプロセスが存在する。いくつかのプロセスを以下に示す。想定している脆弱性のシステムは、OS では Ubuntu, CentOS, AlmaLinux, KaliLinux、ミドルウェアでは MySQL, MariaDB, PostgreSQL、アプリケーションソフトウェア Apache, nginx, ApacheTomcat, Java で構成する。サイバー演習環境システム構成図を図 1 に示す。

クエリ生成機能

ユーザからの要求に対し脆弱性情報を自動で収集できることが必要となる。公開されている脆弱性情報をシステムのクエリ生成機能を活用してユーザが要求した脆弱性情報の取得を行う。キーワード入力後 JVNDB より関連する脆弱性一覧が取得される。要求する脆弱性の CVE-ID の入力後クエリ生成し脆弱性情報の取得を行う。

実装では脆弱性情報を取得するために、まず JVNDB（脆弱性対策情報データベース）で公開されている脆弱性の API をシステムにて Python のプログラムを実行し、読み込みを行った。クエリ生成は、ユーザからキーワード入力を行いキーワードに関連する脆弱性一覧を取得する。脆弱性一覧からユーザが実装したい脆弱性の CVE-ID 入力を行い JVNDB の API から脆弱性情報を xml ファイルで取得後、自動構築設定ファイルを生成するために必要な情報の属性抽出を行い、脆弱性情報ファイルを生成した。

属性抽出機能

脆弱性情報が含まれている xml ファイルには不必要な情報が含まれているため、必要な情報のみを抽出する必要がある。脆弱性情報から脆弱性名, JVNDB-ID, CVE-ID, 日付, 対策情報, ソフトウェア名, ソフトウェアのバージョン情報のみを属性抽出し、json 形式で脆弱性情報ファイルを作成する。

実装では、ユーザが要求した CVE-ID の脆弱性情報が JVNDB から xml ファイル形式で取得する。xml ファイルには実装に不必要な情報が多く含まれているため属性抽出で、脆弱性名,JVNDB-ID,CVE-ID,日付,対策情報,ソフトウェア名・ソフトウェアのバージョン情報の抽出を行い、操作しやすいように脆弱性情報ファイルを json 形式で保存した。

イメージ取得機能

脆弱性情報ファイルから必要なソフトウェア,ソフトウェアのバージョン情報を取得後、リポジトリサーバへイメージファイルの要求を行い、イメージファイルを取得する。

†東京都立産業技術高等専門学校 Tokyo Metropolitan College of Industrial Technology

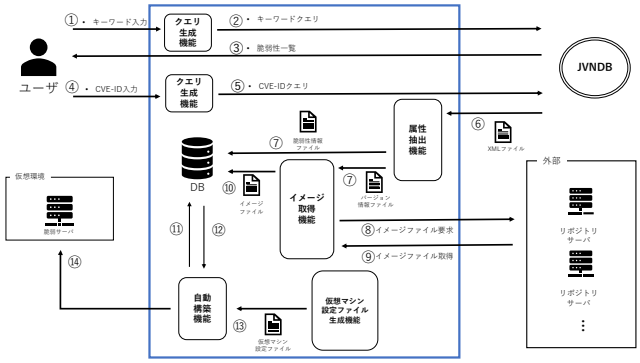

図 1　サイバー演習環境システム構成図

脆弱性情報に記載されている対象のソフトウェアのバージョン情報で「から」「より前の」という条件があった場合、範囲検索を行い必要なソフトウェアをダウンロードする。範囲検索では、リポジトリサーバへの登録日を対象として検索を行う。対象のソフトウェアが一つで限定されている場合、条件一致で検索を行い対象のソフトウェアをダウンロードする。古いバージョンのソフトウェアの場合、あらかじめリポジトリサーバへの要求を定義し、イメージファイルのダウンロードを行う。イメージファイルの取得後データベースに格納し管理する。

実装では、必要なソフトウェアのバージョンを基にリポジトリサーバへイメージファイル要求後、リポジトリサーバからイメージファイルの取得を行った。データベースでは、ソフトウェアごとにテーブルを作成しバージョンごとに管理できるように実装した。

仮想マシン設定ファイル生成機能

仮想マシン設定ファイルを生成するために必要な OS、ミドルウェア、アプリケーションソフトウェアを要求して仮想マシン設定ファイルを生成する。

システムからデータベースに対して脆弱性情報を要求し、データベースからシステムに対して脆弱性情報の受け渡しを行い、脆弱なシステムを構築するためにインストールが必要な、ソフトウェア名・ソフトウェアのバージョン情報を仮想マシン設定ファイルに書き込むための設定を行った。

自動構築機能

ダウンロードしたイメージファイルを基に、ユーザからの脆弱性の要求に応じてシステムが脆弱性情報を格納したデータベースに対して情報を要求後、仮想マシン設定ファイルを生成する。

生成した仮想マシン設定ファイルを実行することで必要なソフトウェアをインストールし、ユーザの要求した脆弱性を含むサーバの自動構築ができる。

仮想環境を構築するために Vagrant を用いて仮想マシン設定ファイルを実行し、仮想環境が立ち上がり、仮想環境内に脆弱性を実装するために必要なソフトウェアがインストールされる。

4. 検証実験

4.1 検証方法

キーワード入力で「Apache」を入力し、キーワードに関連する脆弱性一覧を取得し、その中から要求する脆弱性の CVE-ID を入力しクエリを生成した。使用した脆弱性の CVE-ID と必要なソフトウェアを表 1 に示す。

表 1　CVE-ID とソフトウェア

CVE-ID	ソフトウェア
CVE-2021-44224	Apache HTTP Server 2.4 から 2.4.52
CVE-2022-22720	Apache HTTP Server 2.4 より前の 2.4.7
CVE-2023-25690	Apache HTTP Server 2.4.0 から 2.4.55
CVE-2022-23305	Apache Log4j 1.2

ユーザが要求する脆弱性に対して、脆弱性に関するソフトウェアが正常にインストールされ、設定ファイルが自動生成できるか検証した。検証環境では、対象のソフトウェアがインストールできるか確認するために OS を CentOS7 で固定し検証した。また、その時の脆弱性は Apache と Java の脆弱性に対して関連するソフトウェアが正常にインストールできているか確認した。

4.2 検証結果

Apache の脆弱性を要求した際に、脆弱性情報がデータベースで管理され、脆弱性を実装するために必要なソフトウェアのバージョンをリポジトリサーバから取得することができた。自動構築を行うための設定ファイルが自動生成できることがわかった。

Apache の中でも何種類かのソフトウェアに分けられており、ソフトウェアをインストールするリポジトリサーバが見つからない場合、自動構築を行うための設定ファイルが自動生成できずに終了することがわかった。

5. まとめ

本研究では、脆弱性情報から属性抽出後、脆弱なシステムを自動構築するための設計・実装を行った。現段階では、脆弱性検証をするためのサーバの脆弱性が限定されているため、様々な OSS の脆弱性を検証するために脆弱性のデータベースを常に更新していく必要がある。

また、脆弱性の組み合わせができることで様々な演習が可能となるため、組み合わせ方法について検討する必要があると考える。提案手法により、情報セキュリティ技術者育成のための効率的でリアルな演習環境を提供することが可能となります。

参考文献

[1] Xin Tan, Yuan Zhang, Cheyuan Mi, Jiajun Cao, Kun Sun, Yifan Lin, Min Yang, “Locating the Security Patches for Disclosed OSS Vulnerabilities with Vulnerability-Commit Correlation Ranking”, Proc.of the 2021 ACM SIGSAC Conference on Computer and Communications Security, pp.3282–3299, November 2021.

[2] Lukas Sadlek, Pavel Celeda, Daniel Tovarnak, “Current Challenges of Cyber Threat and Vulnerability Identification Using Public Enumerations”, Proc. of the 17th International Conference on ARES'22, pp.1-8, August 2022.