취약점 데이터베이스 생성 솔루션 1.0

소프트웨어 매뉴얼

2017. 7. 4.

CSSA - Center for Software Security and Assurance
cssa@korea.ac.kr

목 차

1. 개요

A. 매뉴얼 정보

본 매뉴얼은 “취약점 데이터베이스 생성 솔루션 1.0” 소프트웨어의 매뉴얼임.

취약점 데이터베이스 생성 솔루션은 Center for Software Security and Assurance (이하 CSSA) 가 개발한 소프트웨어 취약점 탐지 플랫폼 IoTcube (https://iotcube.net) 의 “Vulnerable Code Clone Detection” 에 사용된 핵심 기술임.

B. 대상

본 매뉴얼은 삼성전자 소프트웨어 센터 Security Lab. 의 실무자를 대상으로 작성되었음.

C. 문서 버전 기록

릴리즈 날짜	문서 버전	사유	작성자
2017년 7월 4일	V1.0	초판	김슬배 (seulbae@korea.ac.kr)

2. 구성 요소

A. 솔루션 구성 요소

본 솔루션은 다음의 디렉터리 트리와 같이 구성됨.

```
vulnDBGen(root)
├── config.py                          : 환경 설정 값 저장
├── initialize.py                      : 솔루션 초기화
├── data                               : 데이터 파일 저장
├── docs                               : 매뉴얼 문서 저장
├── src
│   ├── get_cvepatch_from_git.py       : Git 저장소에서 CVE 패치 추출
│   ├── get_source_from_cvepatch.py    : CVE 패치에서 패치 전/후 함수 복원
│   ├── vul_dup_remover.py             : 중복된 취약 소스 제거
│   ├── vul_verifier.py                : 오탐 유발 취약 소스 제거
│   └── vul_hidx_generator.py          : 취약 소스 인덱싱
└── tools
    ├── FuncParser-opt.jar             : 함수 파서
    ├── parseutility.py                : 함수 파서 헬퍼 라이브러리
    └── cvedatagen
        ├── common.py                  : 데이터 파일 생성 헬퍼 라이브러리
        ├── cveXmlDownloader.py        : CVE 정보 XML 파일 다운로드
        ├── cveXmlParser.py            : XML 파일 파싱 및 데이터 파일 생성
        └── cveXmlUpdater.py           : 신규/수정된 CVE 정보 업데이트
```

B. 코어 모듈 일람

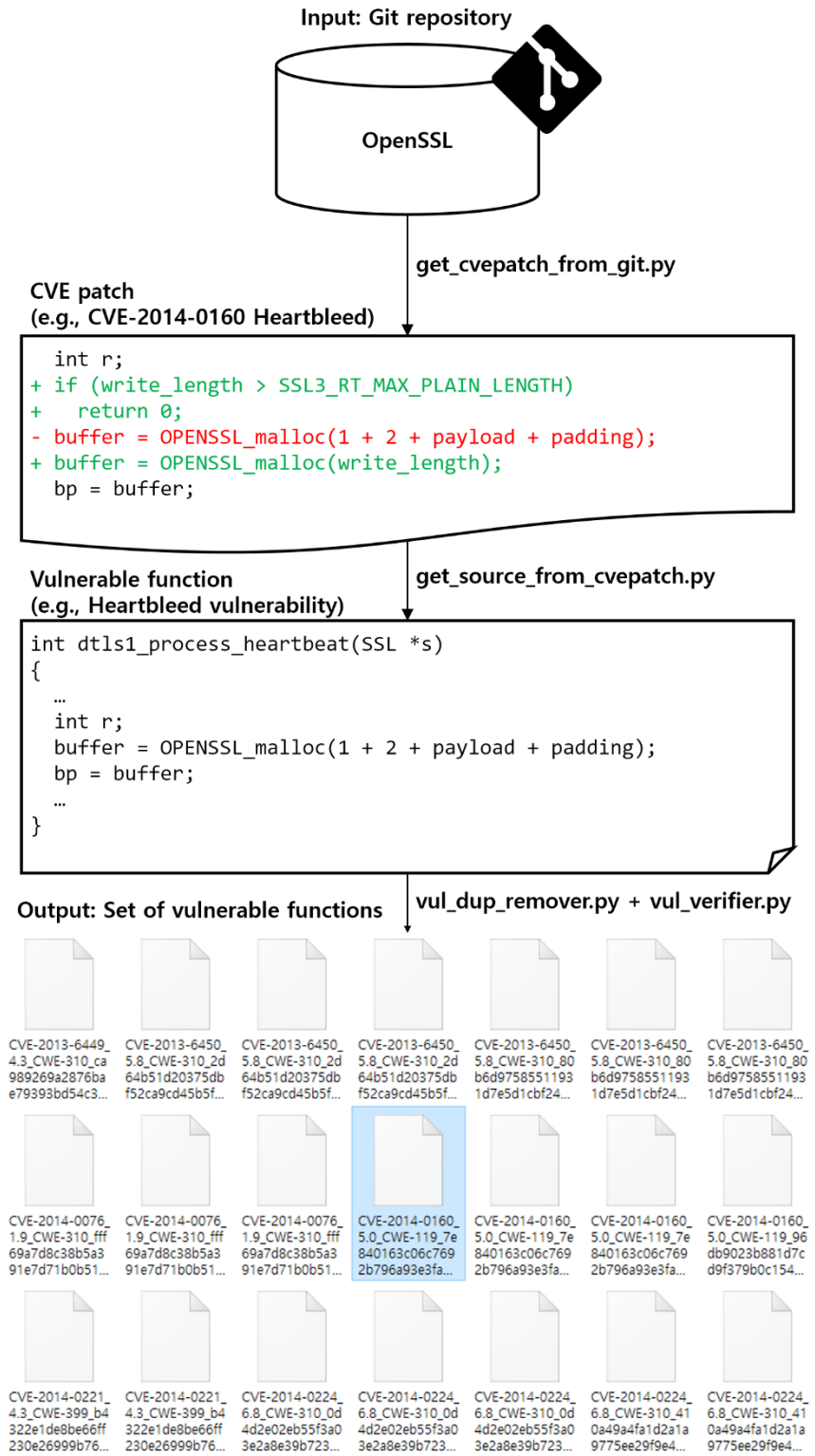
```
$ python get_cvepatch_from_git.py [-h] [-m] REPO_NAME
- Function:    Clone 해 둔 Git repository 에서 CVE 패치 추출
- Input:       Specified Git repository
- Output:      CVE patches (*.diff)
- Arguments:   -h: show help message
               -m: run in multimode (for sub-repository, see 5.D)

$ python get_source_from_cvepatch.py [-h] [-m] REPO_NAME
- Function:    추출된 CVE 패치로부터 old function과 new function을 복원
- Input:       CVE patches (*.diff)
- Output:      Old (unpatched) functions (*_OLD.vul)
               New (patched) functions (*_NEW.vul)
               Patch contents (*.patch)
- Arguments:   -h: show help message
               -m: run in multimode (for sub-repository, see 5.D)

$ python vul_dup_remover.py
- Function:    재생성된 old function들의 중복 제거
- Input:       Saved old/new functions (*.vul)

$ python vul_verifier.py
- Function:    오탐 유발 케이스 일괄 제거
- Input:       Saved old/new functions (*.vul)
```

C. 솔루션 개관


Input: Git repository
OpenSSL
get_cvepatch_from_git.py
CVE patch
(e.g., CVE-2014-0160 Heartbleed)
int r;
+ if (write_length > SSL3_RT_MAX_PLAIN_LENGTH)
+ return 0;
- buffer = OPENSSL_malloc(1 + 2 + payload + padding);
+ buffer = OPENSSL_malloc(write_length);
bp = buffer;
Vulnerable function
(e.g., Heartbleed vulnerability)
get_source_from_cvepatch.py
int dtls1_process_heartbeat(SSL *s)
{
…
int r;
buffer = OPENSSL_malloc(1 + 2 + payload + padding);
bp = buffer;
…
}
Output: Set of vulnerable functions
vul_dup_remover.py + vul_verifier.py

3. 요구 사항

A. 지원 운영 체제

본 솔루션은 POSIX 호환 운영체제(Linux, macOS)와 Windows를 지원함.

B. 운영 체제별 의존 프로그램

i. Windows

- `Python 2` (권장: 2.7 이상) (https://www.python.org/downloads)
- `Git` for Windows (https://git-for-windows.github.io)

ii. macOS

`Python 2` (권장: 2.7 이상), `Git`, `Java` 필요. 다음 명령으로 설치 가능.

```
$ brew install python git
$ brew cask install java
```

iii. Linux

`Python 2` (권장: 2.7 이상), `Git`, `Java` 필요. 다음 명령으로 설치 가능.

```
ex) Debian, Ubuntu, Linux Mint
$ sudo apt-get install python git openjdk-8-jre
```

4. 기본 사용법

A. 환경 설정: config.py

솔루션 사용에 앞서, **config.py**를 설정해야 함.

```
~/vulnDBGen$ vim config.py
```

i. 공통

- gitStoragePath: 취약점을 수집할 Git repository를 clone할 루트 디렉터리.
- version: IoTcube 사용 시 유효 데이터베이스 버전. 일반적인 경우 설정하지 않아도 무방함.

```
gitStoragePath = r“~/gitrepos”
version = “3.0.3”
```

ii. POSIX

Package manager를 통해 git과 java를 설치한 경우, 아래의 default 값을 사용.

```
gitBinary = “git”
diffBinary = “diff”
javaBinary = “java”
```

Custom build 사용 시 각각에 해당하는 path와 binary명 기입.

iii. Windows

Git for Windows (혹은 다른 설치 방법)를 통해 설치된 git.exe와 diff.exe의 정확한 path와 binary명을 입력.

```
gitBinary = r“C:\Program Files\Git\bin\git.exe”
diffBinary = r“C:\Program Files\Git\usr\bin\diff.exe”
```

B. 초기화: initialize.py

i. 공통

initialize.py는 필수 디렉터리를 생성하며, National Vulnerability Database에 등록된 CVE 정보를 수집하여 vulnDBGen/data/cvedata.pkl 파일에 저장.

ii. Windows

윈도우에서 초기화 시, 윈도우용 파서인 FuncParser-opt.exe를 다운로드 받아 vulnDBGen/tools 에 저장함. (POSIX용 파서는 프로젝트 패키지에 미리 포함.)

```
$ cd ~/vulnDBGen
~/vulnDBGen$ python initialize.py
```

위 명령은 tools/cvedatagen/ 이하의 **cveXmlDownloader.py**, **cveXmlParser.py**, 그리고 **cveXmlUpdater.py** 를 순차적으로 실행하는데, 최초 실행을 통해 cvedata.pkl 이 생성된 상태라면 이후 데이터 (신규 CVE 정보 등) 업데이트 시에는 **cveXmlUpdater.py** 만 단독으로 실행하면 됨.

또한, **initialize.py** 를 통해 자동으로 초기화하지 않고 직접 **cveXmlDownloader.py**, **cveXmlParser.py**, 그리고 **cveXmlUpdater.py** 를 순차적으로 실행하여도 그 결과물은 동일함.

C. Git repository 클론

취약점 추출을 위해서는 타겟 repository의 Git object가 필요함. Git object의 획득을 위해 **config.py**의 gitStoragePath에 지정한 경로에, 취약점을 수집할 Git repository를 클론.

ex) Apache HTTPD 와 Linux kernel Git repository 의 클론 과정

```
~$ cd ~/gitrepos                              // gitStoragePath in config.py

~/gitrepos$ git clone https://github.com/apache/httpd.git
Cloning into 'httpd'...
remote: Counting objects: 457213, done.
remote: Compressing objects: 100% (65/65), done.
remote: Total 457213 (delta 34), reused 46 (delta 19), pack-reused 457129
Receiving objects: 100% (457213/457213), 265.54 MiB | 1.84 MiB/s, done.
Resolving deltas: 100% (379737/379737), done.
Checking connectivity... done.

~/gitrepos$ git clone https://kernel.googlesource.com/pub/scm/linux/
kernel/git/torvalds/linux.git
Cloning into 'linux'...
remote: Sending approximately 1.04 GiB ...
remote: Counting objects: 83992, done
remote: Finding sources: 100% (241/241)
remote: Total 5403087 (delta 4534718), reused 5402959 (delta 4534718)
Receiving objects: 100% (5403087/5403087), 1.03 GiB | 9.19 MiB/s, done.
Resolving deltas: 100% (4534718/4534718), done.
Checking connectivity... done.
Checking out files: 100% (59845/59845), done.

~/gitrepos$ ls -l
total 8
drwxrwxr-x 12 squizz squizz 4096 Jun 21 16:03 httpd
drwxrwxr-x 23 squizz squizz 4096 Jun 21 16:06 linux
```

D. CVE 패치 추출: get_cvepatch_from_git.py

get_cvepatch_from_git.py 는 앞선 단계(4.C)에서 clone한 Git repository 에서 CVE 취약점 패치를 추출하여 vulnDBGen/**diff**/REPO_NAME/ 이하에 *.diff 파일로 저장.

ex) HTTPD 와 Linux kernel repository 각각에서 CVE 패치 추출 과정

```
~$ cd ~/vulnDBGen
~/vulnDBGen$ python src/get_cvepatch_from_source.py httpd
Retrieving CVE patch from httpd
Multi-repo mode: OFF.
Initializing... Done.
205 commits in httpd

[+] Writing CVE-2014-3583_5.0_CWE-119_bfee66~.diff Done.
[+] Writing CVE-2015-3183_5.0_~
…
154 patches saved in ~/vulnDBGen/diff/httpd
Done. (1.13679718971 sec)

~/vulnDBGen$ python src/get_cvepatch_from_source.py linux
Retrieving CVE patch from linux
Multi-repo mode: OFF.
Initializing... Done.
311 commits in linux

[+] Writing CVE-2016-3713_5.6_CWE-284_9842df~.diff Done.
[+] …
257 patches saved in ~/vulnDBGen/diff/linux
Done. (10.4052040577 sec)
```

E. 패치 전/후 함수 복원: get_source_from_cvepatch.py

get_source_from_cvepatch.py 는 모인 CVE 취약점 패치로부터 패치 전 함수와 패치 후 함수를 각각 복원함. 복원된 함수들은 vulnDBGen/**vul**/REPO_NAME/ 이하에 저장되는데, 취약점 패치 전 함수는 취약 함수로서 *_OLD.vul, 패치 후의 함수는 안전한 함수로서 *_NEW.vul 로 저장됨.

ex) HTTPD 와 Linux kernel 각각의 CVE 패치로부터 패치 전/후 함수 복원

```
~$ cd ~/vulnDBGen

~/vulnDBGen$ python src/get_source_from_cvepatch.py httpd
Retrieve vulnerable functions from httpd
Multi-repo mode: Off
1/154 [+] CVE-2014-0231_5.0_CWE-399_a30c26~.diff   (proceed)
2/154 [+] CVE-2011-4317_4.3_CWE-020_8ab3c7~.diff   (proceed)
…
Done getting vulnerable functions from httpd
Reconstructed 178 vulnerable functions from 154 patches.
Elapsed: 52.60 sec

~/vulnDBGen$ python src/get_source_from_cvepatch.py linux
Retrieve vulnerable functions from linux
Multi-repo mode: Off
1/257 [+] CVE-2005-0504_4.6_CWE-119_a2f729~.diff   (proceed)
…
Done getting vulnerable functions from linux
Reconstructed 372 vulnerable functions from 257 patches.
Elapsed: 165.58 sec
```

F. 중복된 취약 소스 제거: vul_dup_remover.py

vulnDBGen/vul/ 이하에 저장된 모든 취약 함수들 중 중복된 파일을 하나만 남기고 제거함.

ex) HTTPD 와 OpenSSL repository 의 취약 함수에서 중복 제거

```
~$ cd ~/vulnDBGen

~/vulnDBGen$ python src/vul_dup_remover.py
[RESULT]
       httpd: deleted 31 duplicate files from 534 files.
       linux: deleted 2 duplicate files from 1101 files.
Total: 33 duplicate files.
```

G. 오탐 케이스 일괄 제거: vul_verifier.py

코드 클론 탐지 시 오탐을 유발하는 케이스는 본 모듈에서 일괄적으로 제거됨. 현재는 두 가지 제거 규칙을 적용하고 있으며, 규칙은 정책에 따라 임의로 추가/제거 할 수 있음.

규칙 1) Old function과 new function의 abstraction 결과가 같은 경우 삭제

규칙 2) 함수 헤더가 바뀌어 패치의 전/후 관계 식별이 불가한 경우 삭제

ex) HTTPD, OpenSSL, Linux kernel 의 취약 함수 수집 결과에서 오탐 케이스 제거

```
~/vulnDBGen$ python src/vul_verifier.py
removed 5 FP records from linux
```

H. 취약 소스의 해시 인덱스 파일 생성: vul_hidx_generator.py

앞선 과정 (A~G) 완료 시, 중복과 오탐 케이스가 제거된 취약 함수들만 vulnDBGen/vul/ 이하에 잔류. **vul_hidx_generator.py** 는 잔류 함수들의 해시 인덱스를 생성함. 생성된 해시 인덱스는 *.hidx 확장자를 가지며, vulnDBGen/**hidx**/ 이하에 저장됨.

이 때, 용도에 따라 두 종류의 인덱스 생성 가능:
인자로 -a 0 을 사용할 경우 *exact matching*을 위한 인덱스 생성
인자로 -a 4 를 사용할 경우 *abstract matching*을 위한 인덱스 생성

```
~/vulnDBGen$ python src/vul_hidx_generator.py -a 0 linux
loading source (done)
1 / 360 linux/CVE-2013-1797_6.8_CWE-399_0b7945~_x86.c_106_OLD.vul
…
Hash index saved to: ~/vulnDBGen/hidx/hashmark_0_linux.hidx
Elapsed time: 35.7726960182
```

5. Multimode 사용법

A. Multimode란?

Google Android Project (https://android.googlesource.com/) 처럼 다수의 Git repository가 하나의 프로젝트로 관리되는 경우, **Multimode**를 이용하여 일괄 처리 가능.

ex) Git repositories on Android

```
accessories/manifest
api_council_filter
brillo/manifest
cts_drno_filter
device/aaeon/upboard
device/asus/deb
…
```

B. Multiple Git repository 클론

config.py 에 지정한 gitStoragePath 이하에 프로젝트명(e.g., android)으로 디렉터리 생성 후, 모든 repository를 클론.

ex) 안드로이드 프로젝트의 모든 Git sub-repository 클론 과정

```
~$ cd ~/gitrepos                                    // gitStoragePath in config.py
~/gitrepos$ mkdir android && cd android
~/android$ git clone https://android.googlesource.com/accessories/manifest
accesories/manifest
~/android$ git clone https://android.googlesource.com/brillo/manifest
brillo/manifest
(아웃풋 생략)
```

C. List 파일 생성

프로젝트의 sub-repository 목록을 보관할 리스트 파일은 vulnDBGen/data/**repolists**/ 이하의 **list_프로젝트명** 파일에 저장.

ex) 안드로이드 프로젝트의 Git repository 리스트 파일 생성

```
~$ cd ~/vulnDBGen/data
~/data$ mkdir repolists && cd repolists
~/repolists$ touch list_android
```

ex) 생성한 **list_android** 파일에 sub-repository의 리스트를 저장.

```
~/repolists$ cat list_android
accessories/manifest
api_council_filter
brillo/manifest
…
```

D. CVE 패치 추출 및 패치 전/후 함수 복원 과정

get_cvepatch_from_git.py 와 **get_source_from_cvepatch.py** 의 실행 방법은 동일하나, **-m** 인자를 추가로 사용하여 Multimode를 활성화 함.

ex) 안드로이드 프로젝트의 모든 Git repository로부터 CVE 패치 추출 및 함수 복원

```
~$ cd ~/vulnDBGen
~/vulnDBGen$ python src/get_cvepatch_from_source.py -m android
Retrieving CVE patch from httpd
Multi-repo mode: ON.
Initializing... Done.
…

~/vulnDBGen$ python src/get_source_from_cvepatch.py -m android
Retrieve vulnerable functions from linux
Multi-repo mode: On
…
```

E. 이후 과정

이후의 과정은 **4.E ~ 4.H** 에 기술된 내용과 동일함.

6. 지원

A. 문의

솔루션 사용 중 문제가 발생 할 경우, 6.B의 연락처로 문의 바람.

B. 연락처 정보

CSSA

Phone: (+82)2-3290-4816

E-mail: cssa@korea.ac.kr

Web: https://cssa.korea.ac.kr

7. 참고 문헌

[1] Seulbae Kim, Seunghoon Woo, Heejo Lee, and Hakjoo Oh, "VUDDY: A Scalable Approach for Vulnerable Code Clone Discovery," in *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE. 2017.

[2] Seulbae Kim, Seunghoon Woo, Heejo Lee, and Hakjoo Oh, "Poster: IoTcube: An Automated Analysis Platform for Finding Security Vulnerabilities," in *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE. 2017.

[3] CSSA, “White-box testing in IoTcube,” https://iotcube.net/process/type/wf1, April 2016, Accessed: 2017-07-04