Lab Report

SECURITY INSIDER LAB II
PART 4: IMPLEMENTING SECURE WEB APPLICATIONS

Group 5

Abhijeet Patil
Mohammad Saiful Islam
Thejeswi Preetham Nagendra Kamatchi

Exercise 1: White-Box Web Application Vulnerability Testing

Exercise 2: Black-Box Web Application Vulnerability Testing

1. Download two (or more) web vulnerability scanners and describe how you setup all the appropriate environment settings needed.

For Black-box web application vulnerability testing, we have downloaded OWASP Zed Attack Proxy(also known as OWASP ZAP), nikto and uniscan.

Nikto: Nikto is a small and simple tool, examines a website and reports back the potential vulnerabilities that it found that could use to exploit. Using Nikto is very straight-forward, as the following command:

```
nikto -h [hostname or ip]
```
or
```
perl nikto -host [hostname or ip]
```

Our vBank web application is located at http://192.168.0.29/vBank. Here is the screenshot of the nikto scan:

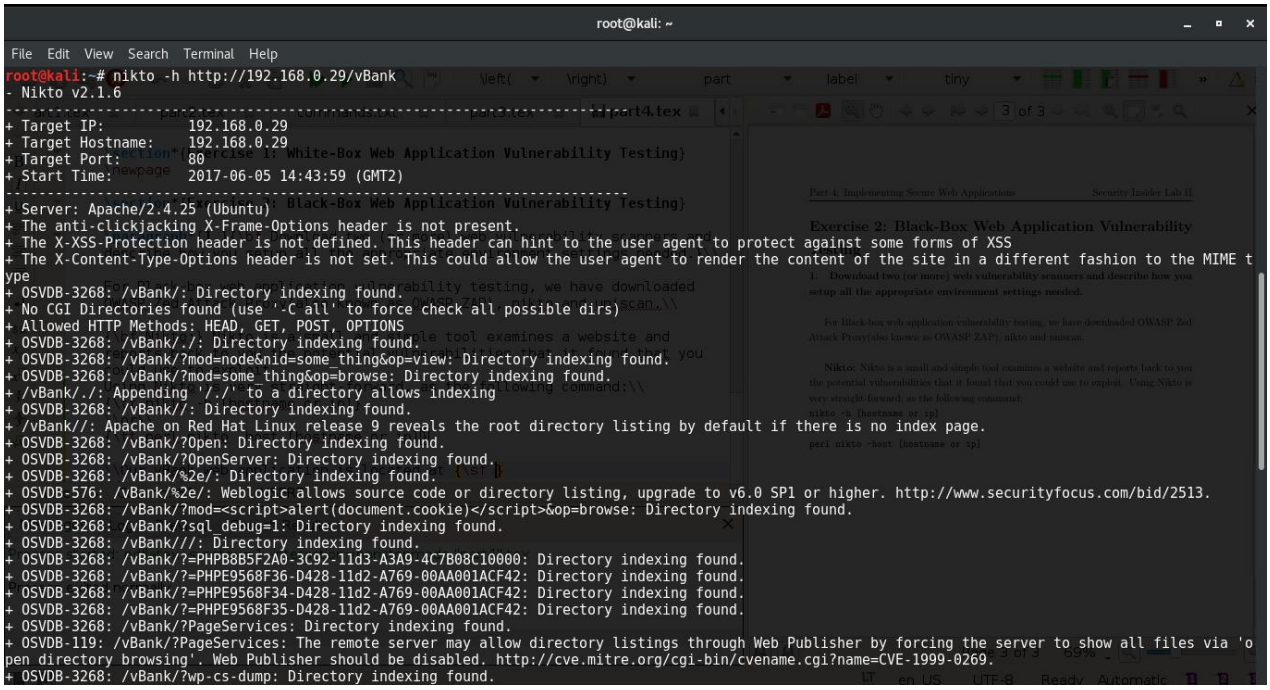

Figure 1: Nikto scan result on vBank

From the scan result we can see that, Nikto has identified Some vulnerabilities along with target server and os version. It identified vulnerabilities with the OSVDB prefix. This is the 'Open Source Vulnerability Database'. This is a database maintained of known vulnerabilities at www.osvdb.org. The vulnerabilities were briefly explained on there. Unfortunately the OSVDB project was shutdown on April,2016.
Here is an example result of identified code, searched before in www.osvdb.org :

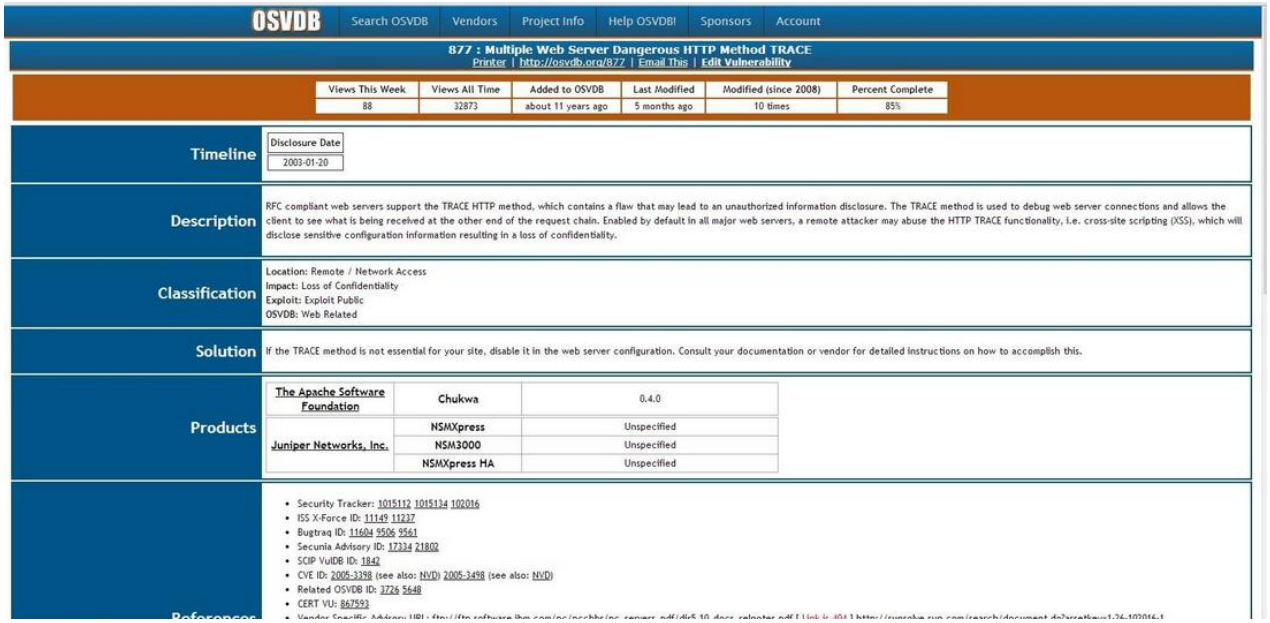

Figure 2: OSVDB search result.

Uniscan: Uniscan is a vulnerability scanner that can scan websites and web-applications for various security issues like LFI, RFI, sql injection, xss etc. It is written in perl. It is open source and can be downloaded from sourceforge project page at http://sourceforge.net/projects/uniscan/ and can be installed by executing install-module.sh.
There are several mode for using uniscan.
With the option 'j' uniscan would fingerprint the server of the url. Server fingerprinting simply runs commands like ping, traceroute, nslookup, nmap on the server ip address and packs the results together.

```
uniscan -u http://192.168.0.29/vBank -j
```

Another option is 'g' which does web based fingerprinting. It looks up specific urls.

```
uniscan -u http://192.168.0.29/vBank -g
```

Using -q option to enable directory test in targeted server.

```
uniscan -u http://192.168.0.29/vBank -q
```

For dynamic scan against the targeted server, uniscan uses -d option.

```
uniscan -u http://192.168.0.29/vBank -d
```

Uniscan also have graphical user interface. The command `uniscan-gui` is used to start gui mode.

OWASP ZAP: OWASP ZAP is a Java-based tool for testing web app security. It has an intuitive GUI and powerful features to do such things as fuzzing, scripting, spidering, proxying and attacking web apps. It is also extensible through a number of plugins. These following commands has been used to install OWASP ZAP.

```
sudo echo "deb OWASP Mantra OS / #OWASP WTE Stable Repository" » /etc/apt/sources.lis
sudo apt-get update
sudo apt-get install owasp-wte-zap
```

Here is the interface of the OWASP ZAP.

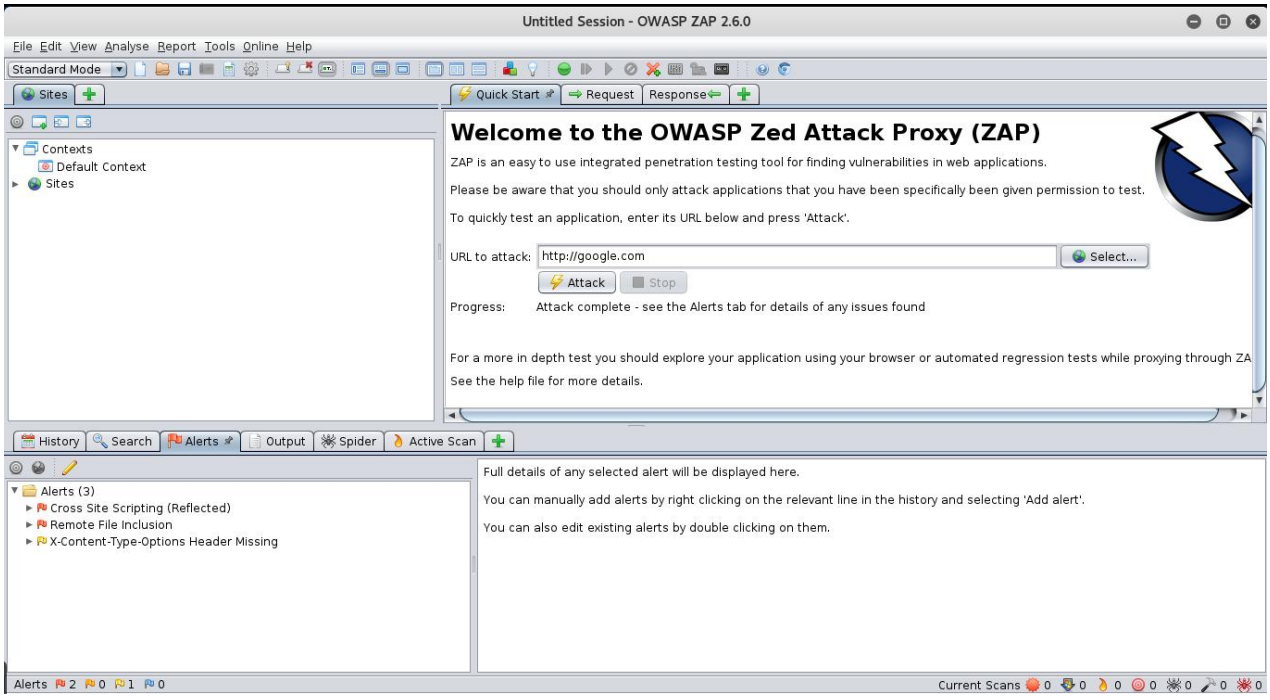

Figure 3: OWASP ZAP interface.

Unlike other tools, OWASP ZAP is gui-based, which makes scanning even easier. In the interface, we just put our target url on the 'url to attack' field and press the 'attack' button. On the lower left box, the scanned vulnerabilities will be showed.

2. Report how you found the different vulnerabilities: SQLi, XSS, etc.

We have used OWASP ZAP to find vulnerabilities of our given webapp vBank. As described before, we just put the target url http://192.168.0.29/vBank , pressed attack.

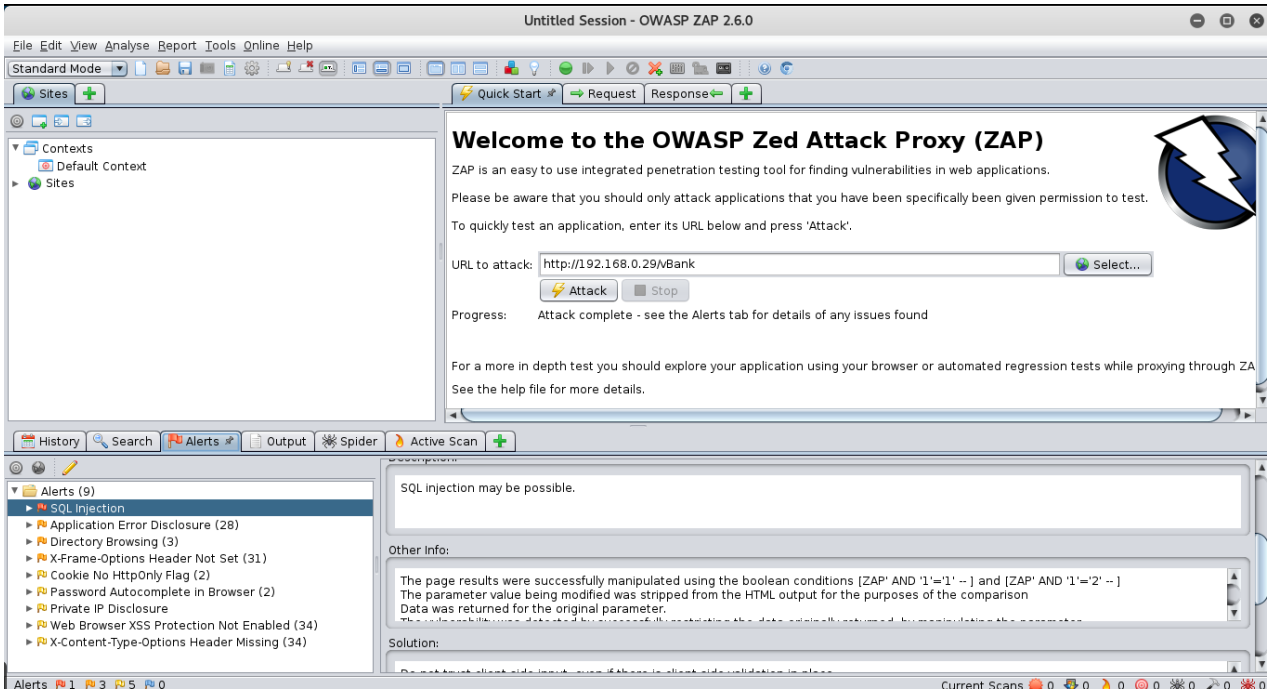

Figure 4: Scanning vBank with OWASP ZAP.

The scan results a several alerts, which can be found on the bottom left corner. The high risk alert is the sql injection. The page results were successfully manipulated using the boolean conditions [ZAP' AND '1'='1' - -] and [ZAP' AND '1'='2' - -].
Besides sql injection, there are other vulnerabilities at medium risk. The OWASP ZAP has alerted for 9 different vunerabilites.

3. Now you have collected enough information about the victim web application and found multiple serious SQL injection vulnerabilities. Use an automatic exploitation tools (e.g. sqlmap) to dump all the database, upload a web shell and prove that you have control of the bank server!