Week 1: Reconnaissance, Information Gathering, and Scanning

INT302: Kali Linux Tools and System Security

Name: Iyiola, Oluwaleke

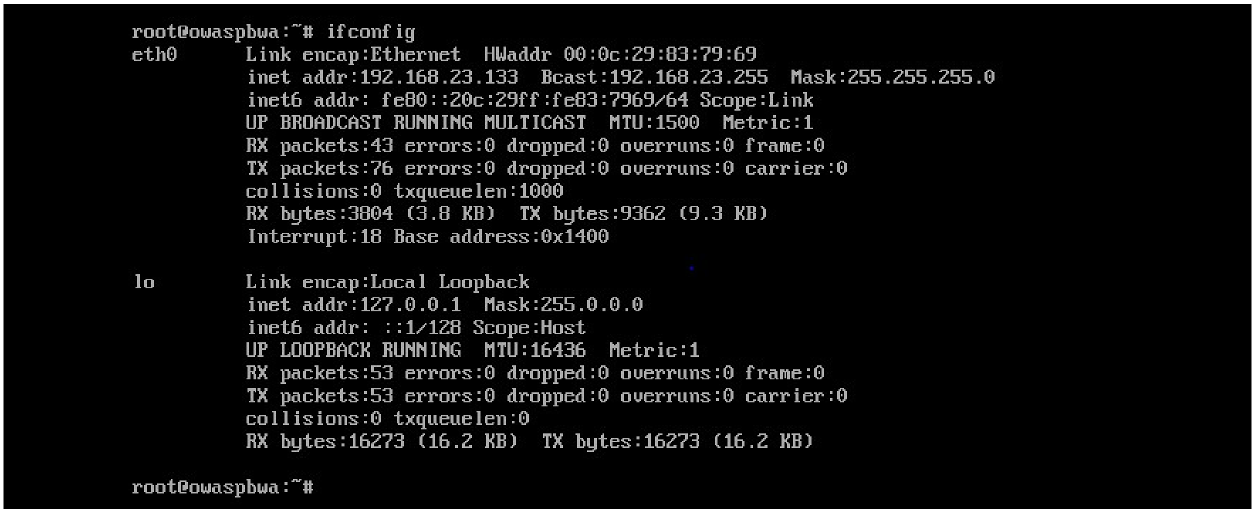
Student ID: IDEAS/24/21635

Lab 4: Basic Port Scanning

Step 1: Gather the IP Address of Your OWASP VM

Exercise 1: Record the IP Address:

- **OWASP VM IP Address**: 192.168.23.133

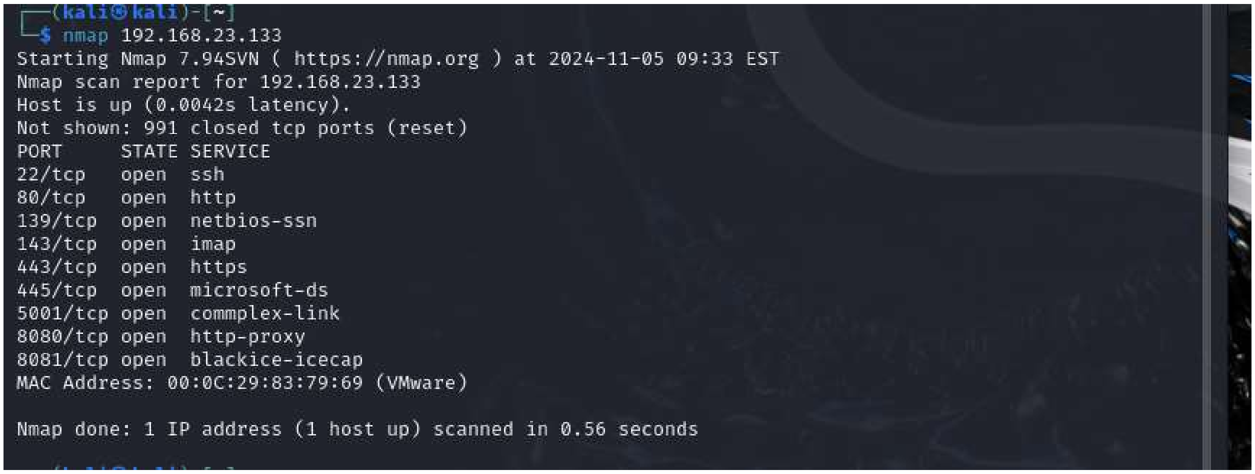


```
root@owaspbwa:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:83:79:69
          inet addr:192.168.23.133  Bcast:192.168.23.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe83:7969/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:43 errors:0 dropped:0 overruns:0 frame:0
          TX packets:76 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3804 (3.8 KB)  TX bytes:9362 (9.3 KB)
          Interrupt:18 Base address:0x1400

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:53 errors:0 dropped:0 overruns:0 frame:0
          TX packets:53 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:16273 (16.2 KB)  TX bytes:16273 (16.2 KB)

root@owaspbwa:~#
```

Step 2: Basic Port Scanning with nmap

Exercise 1:

Perform a basic port scan on your OWASP VM IP address and record your findings:



```
┌──(kali㉿kali)-[~]
└─$ nmap 192.168.23.133
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-05 09:33 EST
Nmap scan report for 192.168.23.133
Host is up (0.0042s latency).
Not shown: 991 closed tcp ports (reset)
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
139/tcp  open  netbios-ssn
143/tcp  open  imap
443/tcp  open  https
445/tcp  open  microsoft-ds
5001/tcp open  commplex-link
8080/tcp open  http-proxy
8081/tcp open  blackice-icecap
MAC Address: 00:0C:29:83:79:69 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.56 seconds
```

Step 3: Aggressive Scanning with nmap

Exercise 2:

Perform an aggressive scan on your OWASP VM IP address and record your findings:

• **Service Versions**: SF:(NULL,4,"\xac\xed\0\x05")

o • **Operating System**: OS details: Linux 2.6.17 - 2.6.36

Step 4: Vulnerability Scanning with nmap

Exercise 3:

Conduct a vulnerability scan on your OWASP VM IP address and record your findings:

• **Vulnerabilities**:

```
VULNERABLE:
|  Cross-domain and Client Access policies.
|    State: VULNERABLE
|      A cross-domain policy file specifies the permissions that a web client such as Java, Adobe Flash, Adobe Reader,
|      etc. use to access data across different domains. A client acces policy file is similar to cross-domain policy
|      but is used for M$ Silverlight applications. Overly permissive configurations enables Cross-site Request
|      Forgery attacks, and may allow third parties to access sensitive data meant for the user.
|    Check results:
|      /crossdomain.xml:
|        <?xml version="1.0"?>
|        <!DOCTYPE cross-domain-policy SYSTEM "http://www.macromedia.com/xml/dtds/cross-domain-policy.dtd">
|        <cross-domain-policy>
|          <allow-access-from domain="*" />
|        </cross-domain-policy>
|    Extra information:
|      Trusted domains:*
|
|    References:
```

```
|       https://www.adobe.com/devnet-
docs/acrobatetk/tools/AppSec/CrossDomain_PolicyFile_Specification.pdf
|       http://acunetix.com/vulnerabilities/web/insecure-clientaccesspolicy-xml-file
|       http://sethsec.blogspot.com/2014/03/exploiting-misconfigured-crossdomainxml.html
|       https://www.adobe.com/devnet/articles/crossdomain_policy_file_spec.html
|       http://gursevkalra.blogspot.com/2013/08/bypassing-same-origin-policy-with-flash.html
|_      https://www.owasp.org/index.php/Test_RIA_cross_domain_policy_%28OTG-CONFIG-008%29
| ssl-ccs-injection:
|   VULNERABLE:
|   SSL/TLS MITM vulnerability (CCS Injection)
|     State: VULNERABLE
|     Risk factor: High
|       OpenSSL before 0.9.8za, 1.0.0 before 1.0.0m, and 1.0.1 before 1.0.1h
|       does not properly restrict processing of ChangeCipherSpec messages,
|       which allows man-in-the-middle attackers to trigger use of a zero
|       length master key in certain OpenSSL-to-OpenSSL communications, and
|       consequently hijack sessions or obtain sensitive information, via
|       a crafted TLS handshake, aka the "CCS Injection" vulnerability.
|
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0224
|       http://www.cvedetails.com/cve/2014-0224
|_      http://www.openssl.org/news/secadv_20140605.txt
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-cookie-flags:
|   /mono/:
|     ASP.NET_SessionId:
|       secure flag not set and HTTPS in use
|_      httponly flag not set
| http-vuln-cve2011-3192:
```

```
|   VULNERABLE:
|   Apache byterange filter DoS
|     State: VULNERABLE
|     IDs:  CVE:CVE-2011-3192  BID:49303
|       The Apache web server is vulnerable to a denial of service attack when numerous
|       overlapping byte ranges are requested.
|     Disclosure date: 2011-08-19
|     References:
|       https://www.securityfocus.com/bid/49303
|       https://www.tenable.com/plugins/nessus/55976
|       https://seclists.org/fulldisclosure/2011/Aug/175
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-3192
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| ssl-poodle:
|   VULNERABLE:
|   SSL POODLE information leak
|     State: VULNERABLE
|     IDs:  CVE:CVE-2014-3566  BID:70574
|           The SSL protocol 3.0, as used in OpenSSL through 1.0.1i and other
|           products, uses nondeterministic CBC padding, which makes it easier
|           for man-in-the-middle attackers to obtain cleartext data via a
|           padding-oracle attack, aka the "POODLE" issue.
|     Disclosure date: 2014-10-14
|     Check results:
|       TLS_RSA_WITH_AES_128_CBC_SHA
|     References:
|       https://www.imperialviolet.org/2014/10/14/poodle.html
|       https://www.securityfocus.com/bid/70574
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-3566
|_      https://www.openssl.org/~bodo/ssl-poodle.pdf
```

```
| http-csrf:
| Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=192.168.23.133
|   Found the following possible CSRF vulnerabilities:
|
|     Path: https://192.168.23.133:443/ghost/
|     Form id:
|     Form action: submit.php
|
|     Path: https://192.168.23.133:443/gallery2/main.php
|     Form id: search_searchblock
|     Form action: main.php
|
|     Path: https://192.168.23.133:443/shepherd/login.jsp
|     Form id:
|     Form action: login
|
|     Path: https://192.168.23.133:443/getboo/
|     Form id: search_box
|     Form action: psearch.php
|
|     Path: https://192.168.23.133:443/AppSensorDemo/login.jsp
|     Form id:
|_    Form action: Login
| ssl-dh-params:
|   VULNERABLE:
|   Diffie-Hellman Key Exchange Insufficient Group Strength
|     State: VULNERABLE
|       Transport Layer Security (TLS) services that use Diffie-Hellman groups
|       of insufficient strength, especially those using one of a few commonly
|       shared groups, may be susceptible to passive eavesdropping attacks.
|     Check results:
|       WEAK DH GROUP 1
|             Cipher Suite: TLS_DHE_RSA_WITH_AES_128_CBC_SHA
|             Modulus Type: Safe prime
|             Modulus Source: mod_ssl 2.2.x/1024-bit MODP group with safe prime modulus
|             Modulus Length: 1024
|             Generator Length: 8
|             Public Key Length: 1024
|     References:
|_      https://weakdh.org
| http-enum:
|   /wordpress/: Blog
|   /test/: Test page
|   /mono/: Mono
|   /crossdomain.xml: Adobe Flash crossdomain policy
|   /phpmyadmin/: phpMyAdmin
|   /wordpress/wp-login.php: Wordpress login page.
|   /icons/: Potentially interesting folder w/ directory listing
|_  /images/: Potentially interesting folder w/ directory listing
445/tcp  open  microsoft-ds
5001/tcp open  commplex-link
8080/tcp open  http-proxy
| http-cookie-flags:
|   /manager/html/upload:
|     JSESSIONID:
|       httponly flag not set
|   /manager/html:
|     JSESSIONID:
|_      httponly flag not set
| http-enum:
```

```
|   /examples/: Sample scripts
|   /manager/html/upload: Apache Tomcat (401 Unauthorized)
|   /manager/html: Apache Tomcat (401 Unauthorized)
|_  /docs/: Potentially interesting folder
8081/tcp open  blackice-icecap
MAC Address: 00:0C:29:83:79:69 (VMware)

Host script results:
| smb-vuln-regsvc-dos:
|   VULNERABLE:
|   Service regsvc in Microsoft Windows systems vulnerable to denial of service
|     State: VULNERABLE
|       The service regsvc in Microsoft Windows 2000 systems is vulnerable to denial of service caused by a null deference
|       pointer. This script will crash the service if it is vulnerable. This vulnerability was discovered by Ron Bowes
|       while working on smb-enum-sessions.
|_
|_smb-vuln-ms10-061: Could not negotiate a connection:SMB: ERROR: Server returned less data than it was supposed to (one or more fields are missing); aborting [14]
|_samba-vuln-cve-2012-1182: Could not negotiate a connection:SMB: ERROR: Server returned less data than it was supposed to (one or more fields are missing); aborting [14]
|_smb-vuln-ms10-054: false
```

INT302: Kali Linux Tools and System Security – Lab 5: Wireshark

Exercise 1:

- **Explore the Wireshark GUI. Identify and list the main components you see, including where to find the Statistics menu.**

Week 1: Reconnaissance, Information Gathering, and Scanning

INT302: Kali Linux Tools and System Security

Wireshark GUI Components:

1. Main Menu Bar

1. File

2. Edit

3. View

4. Go

5. Capture

6. Analyze

7. Statistics (Statistics menu located here)

8. Telephony

9. Wireless

10. Tools

11. Help

2. Toolbar

1. New Capture (Start/Stop)

2. Open Capture File

3. Save Capture File

4. Print

5. Quit

3. Main Window

1. Packet List Pane (displays captured packets)

2. Packet Details Pane (displays packet details)

3. Packet Bytes Pane (displays packet bytes)

4. Packet List Pane Columns

1. No. (packet number)

2. Time

3. Source

4. Destination
5. Protocol
6. Length
7. Info

5. Status Bar

1. Capture file information
2. Packet count
3. Filter status

6. Statistics Menu

Located under the Statistics menu:

1. Summary
2. Protocol Hierarchy
3. Conversations
4. Endpoints
5. Packet Lengths
6. Packet Counts
7. IO Graphs
8. TCP Stream Graph
9. UDP Stream Graph

Other notable components:

1. Filter bar (applies display filters)
2. Capture options (interface selection, capture options)
3. Expert Information (displays expert analysis)

Step 2: Capturing Network Traffic Using the Wireshark GUI:

Exercise 2:

• Capture network traffic using both Wireshark and tshark. Compare the two methods and note any differences in the user experience

Answer:

The differences are as follows:

- Interface: WireShark uses GUI while TSHARK uses CLI
- Resource: Wireshark uses more resource than TShark (Lightweight)
- Scalability: Tshark is better for large capture while Wireshark is limited
- Automation: TShark is easier than WireShark in automation
- Wireshark has more visual representation than TShark (Text-Based)

Step 3: Analyzing Captured Packets

Exercise 3:

• Use filters to analyze different types of traffic. Record the following:

o Number of HTTP packets captured: __________

o Number of DNS packets captured: __________

o Specific IP addresses you identified in the traffic: __________

Step 4: Understanding Packet Details

Exercise 4:

• Select a packet and list the following information:

o Source IP: 192.168.23.1

o Destination IP: 192.168.23.2

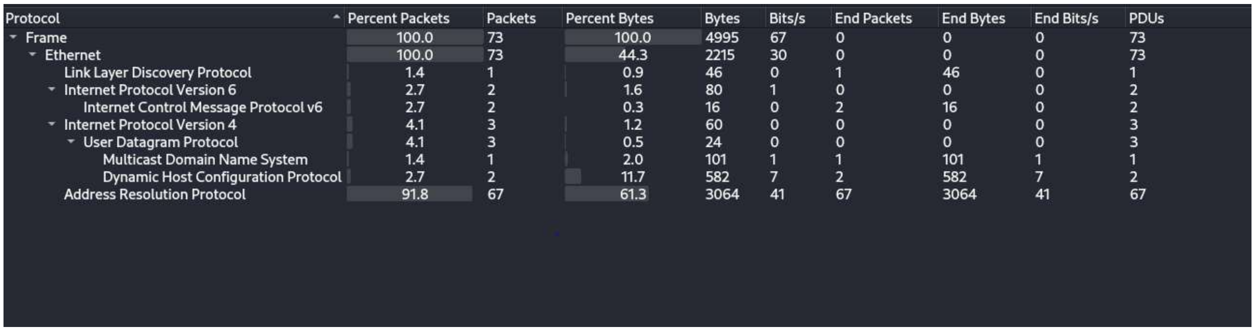
o Protocol: 4, IPV4 (0x0800

o Any TCP Flags observed:No

Step 5: Advanced Packet Analysis Techniques

Exercise 6:

• Take a screenshot of the Protocol Hierarchy and analyze the data. Which protocol is most prevalent in your capture? IpV6



Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s	PDUs
Frame	100.0	73	100.0	4995	67	0	0	0	73
Ethernet	100.0	73	44.3	2215	30	0	0	0	73
Link Layer Discovery Protocol	1.4	1	0.9	46	0	1	46	0	1
Internet Protocol Version 6	2.7	2	1.6	80	1	0	0	0	2
Internet Control Message Protocol v6	2.7	2	0.3	16	0	2	16	0	2
Internet Protocol Version 4	4.1	3	1.2	60	0	0	0	0	3
User Datagram Protocol	4.1	3	0.5	24	0	0	0	0	3
Multicast Domain Name System	1.4	1	2.0	101	1	1	101	1	1
Dynamic Host Configuration Protocol	2.7	2	11.7	582	7	2	582	7	2
Address Resolution Protocol	91.8	67	61.3	3064	41	67	3064	41	67

3. IO Graphs:

Exercise 7:

• Create an IO Graph showing TCP traffic. Describe any noticeable patterns you observe:

Step 6: Exporting Captured Data

Exercise 8:

• Save your capture file and describe a scenario where you would need to review this data later.

What specific findings do you hope to extract?

Answer:

Scenario: A situation whereby a cloud-based CRN system office is experiencing concurrent connectivity issues.

Symptoms: disconnections that has become frequents. Slow data loading, Connection Time out.

Troubleshooting:

First capture the office network router. CRM IP address traffic filter, analysis of the lose packet, latency and TCP retransmission, Look for potential bottlenecks

Step 7: Practical Applications of Wireshark

1. **Detecting Network Issues**:

Exercise 9:

• Describe a real-world scenario where you would use Wireshark to troubleshoot a network issue.

What specific symptoms would you investigate? __________

2. **Security Analysis**:

Exercise 10:

• Identify at least two potential security threats in your captured traffic. What indicators led you to suspect these activities? __________

INT302: Kali Linux Tools and System Security – Lab 6: Advanced Packet Analysis Techniques

Lab Steps

Step 1: Dissecting Protocols 1. **TCP Analysis**:

Exercise 1: Describe the purpose of the SYN and ACK flags in the TCP handshake. How do these flags indicate the status of a connection? __________

2. **HTTP Analysis**:

o Filter the captured traffic to show only HTTP packets using the filter: http.

o Examine the headers of an HTTP request and response

Key Headers to Focus On:

• Request Method (GET, POST, etc.)

• Status Code (200, 404, etc.)

• User-Agent

Exercise 2:

• Choose an HTTP packet and summarize its request method, status code, and any notable headers. What can you infer about the transaction? __________

3. **DNS Analysis**:

Exercise 3:

• Identify a DNS query and its corresponding response. What information does the response provide, and how is it structured? __________

Step 2: Creating Custom Filters

Week 1: Reconnaissance, Information Gathering, and Scanning

INT302: Kali Linux Tools and System Security

Exercise 4:

• Create a custom filter that captures only TCP traffic from your machine to a specific target IP.

Document the filter syntax and the packets captured. __________

Exercise 5:

• Write a filter that captures traffic on a specific port (e.g., HTTP port 80) and analyze the results.

What packets were captured? __________

Step 3: Identifying Vulnerabilities

1. **Recognizing Anomalies**:
2. **Exercise 6**:
   - Analyze your capture for any anomalies or indicators of potential vulnerabilities. Document your findings and suggest possible remediation steps. __________

3. **Security Protocols**:

   Exercise 7:
   • Capture HTTPS traffic and identify the initial handshake packets. What information is exchanged during this handshake, and how does it contribute to security? __________

Step 4: Practical Applications and Reporting 1. **Conduct a Security Assessment**:
Exercise 8:
• Prepare a brief report summarizing your findings during the assessment. Include potential risks and recommended actions. __________

2. **Creating a Capture Report**:
Exercise 9:
• Create a capture report that includes your objectives, methods, key findings, and any recommendations for improving network security. __________