

ARP Cache Poisoning with Scapy

Where are the VMs?

- They're on dh2020pc[0-5] under:
“/virtual/csc427scapy/”
- The VMs are based on Tiny Core Linux, a very strange tiny Linux distro that stores everything in RAM
- TinyCore_Preinstalled.7z contains a suspended VMWare image that has Scapy, tcpdump, and Python already installed
- Beware: Rebooting this VM will remove all these preinstalled programs!
- TinyCore-8.2.1.iso contains the bootable iso, which can be used to create more bare VMs easily

What are we doing?

- **Open the preinstalled TinyCore VM, we will use this for the attacker**
- **Create a new VM using the TinyCore iso file, this will be the victim**
- **<http://www.secdev.org/projects/scapy/demo.html>**
- **We will be using ARP cache poisoning to route traffic through our attacker**

What is ARP?

- **Address resolution protocol**

Hardware Type		Protocol Type
Hardware length	Protocol length	Operation Request 1, Reply 2
Sender hardware address (For example, 6 bytes for Ethernet)		
Sender protocol address (For example, 4 bytes for IP)		
Target hardware address (For example, 6 bytes for Ethernet) (It is not filled in a request)		
Target protocol address (For example, 4 bytes for IP)		

What does it do?

- **ARP Resolves IP addresses to link-layer MAC addresses**
- **Simple protocol, where one node broadcasts, asking what MAC address corresponds to the target IP address**
- **The response is targeted at the sender of the request, and they cache the response**

What is ARP Poisoning?

- **We can spoof responses, and tell our victim that we own the gateway IP address**
- **We can tell the gateway that we own the victim IP address**
- **Then all traffic to and from our victim goes to us**
- **By “we own”, I mean “our mac address has this IP associated with it”**

What does ARP look like?

- “Who has 192.168.0.1?, tell 192.168.0.23”

```
▼ Ethernet II, Src: Giga-Byt_1c:89:9c (1c:1b:0d:1c:89:9c), Dst:
  > Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  > Source: Giga-Byt_1c:89:9c (1c:1b:0d:1c:89:9c)
  Type: ARP (0x0806)
▼ Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IPv4 (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (1)
  Sender MAC address: Giga-Byt_1c:89:9c (1c:1b:0d:1c:89:9c)
  Sender IP address: 192.168.0.23
  Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
  Target IP address: 192.168.0.1
```

- “192.168.0.1 is at 8C:09:F4:B2:A8:87”

```
▼ Ethernet II, Src: ArrisGro_b2:a8:87 (8c:09:f4:b2:a8:87), Dst: G
  > Destination: Giga-Byt_1c:89:9c (1c:1b:0d:1c:89:9c)
  > Source: ArrisGro_b2:a8:87 (8c:09:f4:b2:a8:87)
  Type: ARP (0x0806)
  Padding: 0000000000000000000000000000000020202020
▼ Address Resolution Protocol (reply)
  Hardware type: Ethernet (1)
  Protocol type: IPv4 (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: reply (2)
  Sender MAC address: ArrisGro_b2:a8:87 (8c:09:f4:b2:a8:87)
  Sender IP address: 192.168.0.1
  Target MAC address: Giga-Byt_1c:89:9c (1c:1b:0d:1c:89:9c)
  Target IP address: 192.168.0.23
```

What do we do next?

- `poison_target = ARP(op='is-at', psrc=gateway_ip, pdst=target_ip, hwdst=target_mac)`
- `poison_gateway = ARP(op='is-at', psrc=target_ip, pdst=gateway_ip, hwdst=gateway_mac)`
- **Find out how to spoof a response to route traffic through your attacker VM**
- **Spam your two ARP poison packets to the gateway and the victim**

How do I know if it worked?

- Check the arp table with the arp command
- Forward traffic through yourself:
`# echo 1 > /proc/sys/net/ipv4/ip_forward`
- (Or dont! The VM won't be able to connect outside the gateway)
- Do some cool things with iptables DNAT prerouting filters (I might be able to help)

What to install?

- **Go to apps, cloud, browse**
- **Search for python.tcz, hit “Go” in the bottom left**
- **Then install python-pip.tcz**
- **Then install tcpdump.tcz**
- **Then open a terminal and type:**
- **“sudo python -m pip install scapy”**