网络地址转换实验

范子墨

2021K8009929006

1. **实验内容**

1、SNAT实验

* 1. 运行给定网络拓扑(nat\_topo.py)
  2. 在n1, h1, h2, h3上运行相应脚本
     1. n1: disable\_arp.sh, disable\_icmp.sh, disable\_ip\_forward.sh, disable\_ipv6.sh
     2. h1-h3: disable\_offloading.sh, disable\_ipv6.sh
  3. 在n1上运行nat程序： n1# ./nat exp1.conf
  4. 在h3上运行HTTP服务：h3# python3 ./http\_server.py
  5. 在h1, h2上分别访问h3的HTTP服务
     1. h1# wget http://159.226.39.123:8000
     2. h2# wget http://159.226.39.123:8000

2、DNAT实验

* 1. 运行给定网络拓扑(nat\_topo.py)
  2. 在n1, h1, h2, h3上运行相应脚本
     1. n1: disable\_arp.sh, disable\_icmp.sh, disable\_ip\_forward.sh, disable\_ipv6.sh
     2. h1-h3: disable\_offloading.sh, disable\_ipv6.sh
  3. 在n1上运行nat程序： n1# ./nat exp2.conf
  4. 在h1, h2上分别运行HTTP Server： h1/h2# python3 ./http\_server.py
  5. 在h3上分别请求h1, h2页面
     1. h3# wget http://159.226.39.43:8000
     2. h3# wget <http://159.226.39.43:8001>

3、手动构造一个包含两个nat的拓扑

* 1. h1 <-> n1 <-> n2 <-> h2
  2. 节点n1作为SNAT， n2作为DNAT，主机h2提供HTTP服务，主机h1穿过两个nat连接到h2并获取相应页面

1. **设计思路**
2. 总体设计思路

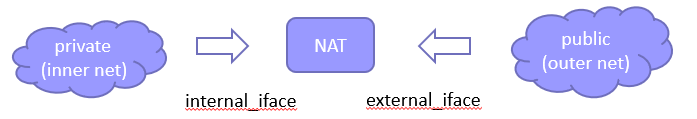
首先依据原有代码，可以看到在main函数中引出两个本次实验相关的函数，一个是nat\_init，一个是ustack\_run。

在ustack\_run中会持续接收数据包，如果是ip数据包，则会跳转到handle\_ip\_packet函数中处理。在handle\_ip\_packet函数中，如果不是目的地址为本接口的ICMP包的话，均会跳转至nat\_translate\_packet函数处理。在nat\_translate\_packet函数中首先判断数据包的方向，而后如果是无效方向则发送icmp，如果协议不是TCP则记录错误日志，最后进行网络地址转换，即跳转到do\_translation函数中。在该函数中，如果该数据包在NAT中有对应连接映射，则直接处理即可。若该数据包的方向为DIR\_OUT，且为该TCP连接的第一个数据包（请求连接数据包），NAT中没有对应连接映射 (SNAT)，则分配端口，创建新的映射项。若该数据包的方向为DIR\_IN，为该TCP连接的第一个数据包，NAT中没有对应连接映射，但有对应处理规则 (DNAT)，则新建映射项。最终将处理好的数据包发送。

在nat\_init的函数中，会在parse\_config函数中对配置文件进行解析，同时新创一个线程，在nat\_timeout中定时清理已完成的NAT映射。

1. get\_packet\_direction

由于NAT只处理两个方向的数据包，即当源地址为内部地址，且目的地址为外部地址时，方向为DIR\_OUT和当源地址为外部地址，且目的地址为external\_iface地址时，方向为DIR\_IN。由于只有两个方向，因此可以简化判断条件位源地址为外部地址或内部地址。



图表 1：数据包方向

首先获取ip头部，而后对源地址进行最长前缀查找，即在路由表中查询，如果目的地址相应转发条目对应的iface与内部接口相同的话，则方向为DIR\_OUT，与外部接口相同则为DIR\_IN。

1. do\_translation

首先提取ip头部和tcp头部，而后根据数据包方向获取端口号和ip地址。

    u32 addr = (dir == DIR\_IN)? ntohl(iphdr->saddr) : ntohl(iphdr->daddr);

    u16 port = (dir == DIR\_IN)? ntohs(tcphdr->sport) : ntohs(tcphdr->dport);

由于NAT可能需要同时维护数万条映射关系，链表查找方式效率很低，因此考虑使用hash方式。在原代码中也提供了一种hash函数，使用该hash函数对ip和port进行映射。

u8 hash = hash8((char\*)&rs, sizeof(rmt\_set\_t));

而后，如果方向是DIR\_IN的话，对NAT映射表进行遍历，如果存在外部ip地址和端口符合该数据包目的地址ip地址和端口的，则设置found=1。

如果found=1，则表示可以直接利用该映射，修改TCP头部的目的端口及IP头部的目IP地址，并更新映射表中的序列号，和连接信息的ACK和FIN标志，最后更新时间。

如果found=0，则遍历DNAT规则表，检查rule对应的外部端口是否被分配，rule的外部IP地址是否与ip头部的目标地址匹配且rule的外部端口是否与tcp头部的目的端口匹配。若满足条件，则按照规则创造新的映射项，并将映射项最终添加到映射表的链表中。最后与found=1类似。

if (nat.assigned\_ports[rule->external\_port] == 0 && rule->external\_ip == ntohl(iphdr->daddr) \

        && rule->external\_port == ntohs(tcphdr->dport))

若数据包方向为DIR\_OUT，则同样先寻找到是否有匹配的映射项。若found=1，则同样更新映射项的信息，包括外部序列号、ACK、FIN标志等。

如果未找到匹配的映射项，说明是新的连接，需要分配一个外部端口，并创建新的映射项。遍历 NAT\_PORT\_MIN 到 NAT\_PORT\_MAX 的端口号，找到第一个未被分配的端口。如果找到可用端口，将其标记为已分配，并将新的映射项的信息设置为新分配的端口和输出方向数据包的源IP和源端口。而后也更新映射表的信息。

最终更新校验和后发送数据包。

1. parse\_config

首先查看要解析的配置文件都大概是什么样子的，可以看到首先规定了internal-iface和external-iface，在实验二的配置文件中还包含目标网络地址转换的规则。首先获取内部接口和外部接口，只需读取而后赋值即可。而对于DNAT的规则，其形如“dnat-rules: 159.226.39.43:8000 -> 10.21.0.1:8000”，因此按照空格分为四段，这里需要将字符串形式的ip地址转换为32位整数。在最开始尝试使用函数inet\_addr或者inet\_aton但是都出现了问题（比如说core\_dump），最后决定还是使用最原始的方法将字符串转换为整数。

sscanf(name, "%u.%u.%u.%u", &ip4, &ip3, &ip2, &ip1);

    ip = (ip4 << 24) | (ip3 << 16) | (ip2 << 8) | (ip1);

1. nat\_timeout

相比于之前实验中的老化操作还更为简单一些，只包含有两种情况，首先是双方都已发送FIN且回复相应ACK的连接，一方发送RST包的连接，可以回收。另一种是双方已经超过60秒未传输数据的连接，认为其已经传输结束，可以回收。因此遍历NAT映射表的每个槽中的每个映射项，若老化或传输已完成，则收回端口。

if (now - cur->update\_time > TCP\_ESTABLISHED\_TIMEOUT || is\_flow\_finished(&(cur->conn))) {

        nat.assigned\_ports[cur->external\_port] = 0;

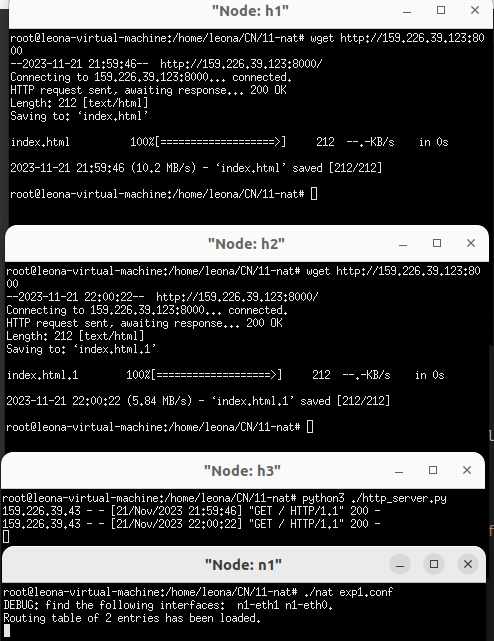
        list\_delete\_entry(&(cur->list));

        free(cur);}

1. **结果**

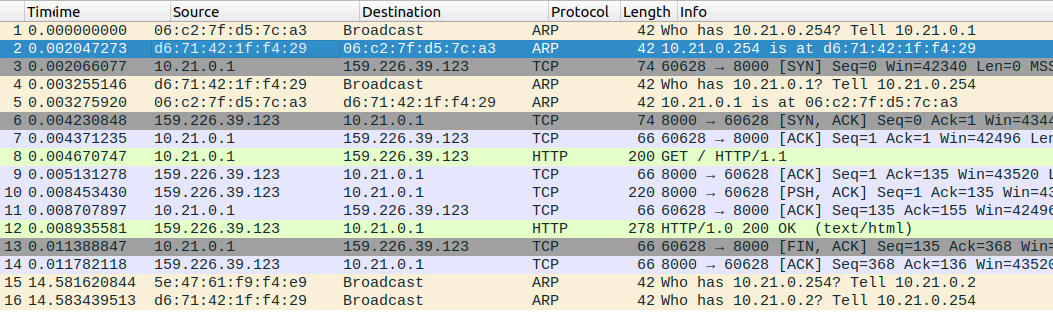
1、实验内容1

如图在n1上运行nat程序，h3上运行HTTP服务，在h1和h3上分别访问h3的HTTP服务

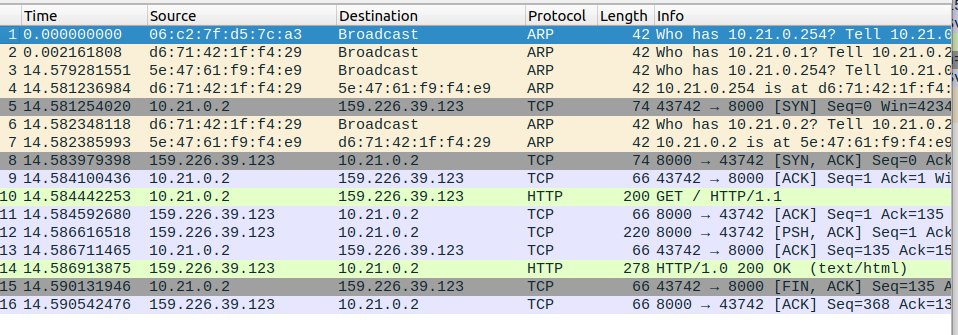


图表 2：实验一终端

抓包结果如下，可以看到TCP连接正常建立到关闭的过程。

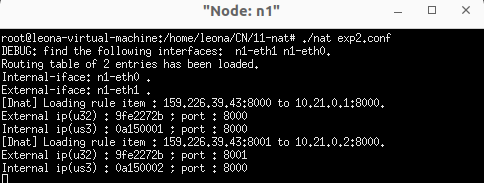


图表 3：h1访问h3抓包



图表 4：h2访问h3抓包

2、实验内容2

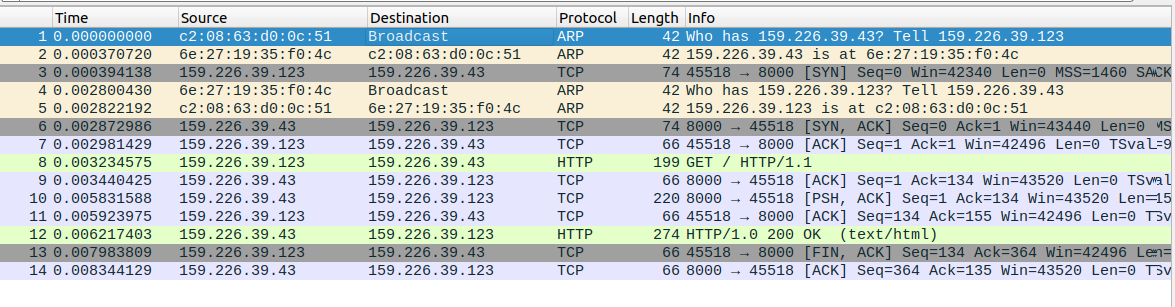


图表 5：n1读取规则成功

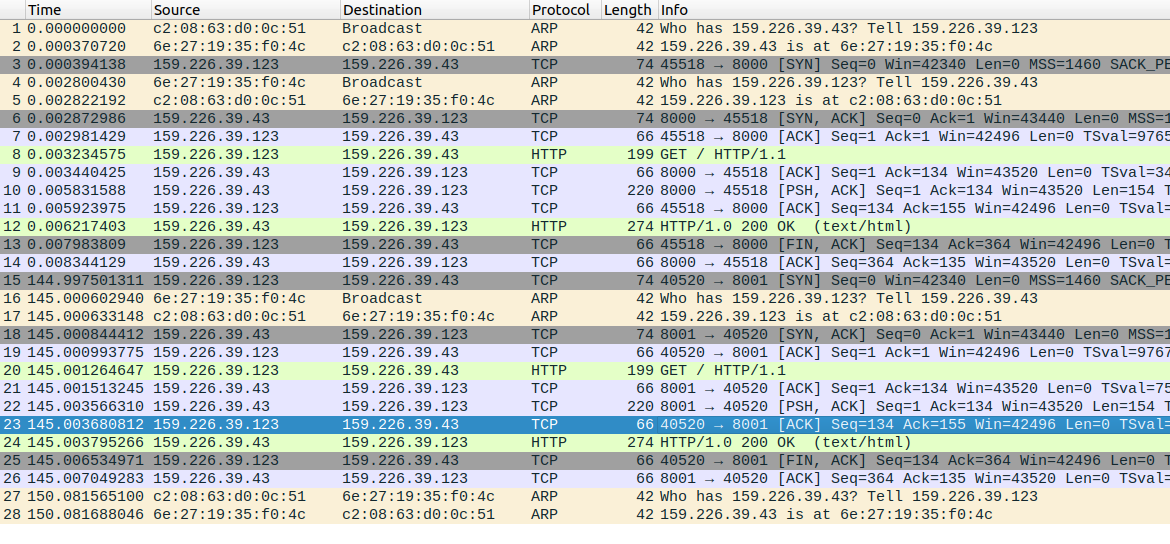


图表 6：h3请求h1，h2页面

以下可以看到TCP连接正常建立到关闭的过程。

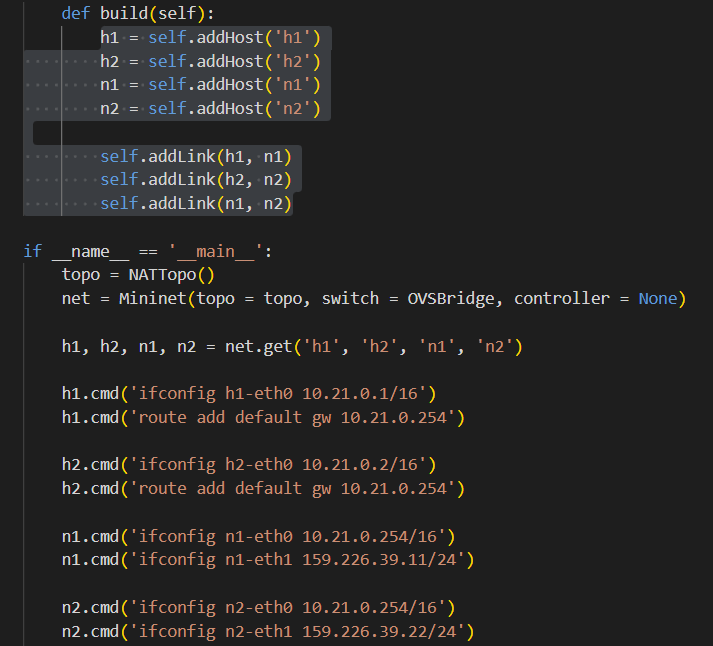


图表 7：h1抓包



图表 8：h2抓包

3、更改拓扑文件如下



设置两个NAT节点的配置文件分别为

internal-iface: n1-eth0

external-iface: n1-eth1

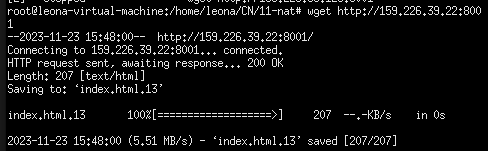
dnat-rules: 159.226.39.11:8001 -> 10.21.0.1:8000

internal-iface: n2-eth0

external-iface: n2-eth1

dnat-rules: 159.226.39.22:8001 -> 10.21.0.2:8000

而后进行访问，可以看到能够正常连接。



图表 9：h1访问h2