网络地址转换实验报告

李昊宸

2017K8009929044

（一）nat机制实现

一、实验内容

1. SNAT源地址转换实验：

1）运行网络拓扑(nat\_topo.py)

2）在n1, h1, h2, h3上运行相应脚本：

n1: disable\_arp.sh, disable\_icmp.sh, disable\_ip\_forward.sh, disable\_ipv6.sh

h1-h3: disable\_offloading.sh, disable\_ipv6.sh

3）在n1上运行nat程序

4）在h3上运行HTTP服务

5）在h1, h2上分别访问h3的HTTP服务

2. DNAT目的地址转换实验：

1）运行网络拓扑(topo.py)

2）在n1, h1, h2, h3上运行相应脚本：

n1: disable\_arp.sh, disable\_icmp.sh, disable\_ip\_forward.sh, disable\_ipv6.sh

h1-h3: disable\_offloading.sh, disable\_ipv6.sh

3）在n1上运行nat程序

4）在h1, h2上分别运行HTTP Server

5）在h3上分别请求h1, h2页面

3. 两个nat穿越实验：

1）手动构造一个包含两个nat的拓扑：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

二、实验流程

1. 搭建实验环境

include：相关头文件

scripts：禁止协议栈的数据包处理

main.c： NAT的代码实现，编译后在路由器结点上运行

ip.c：处理IP数据包，包括转发

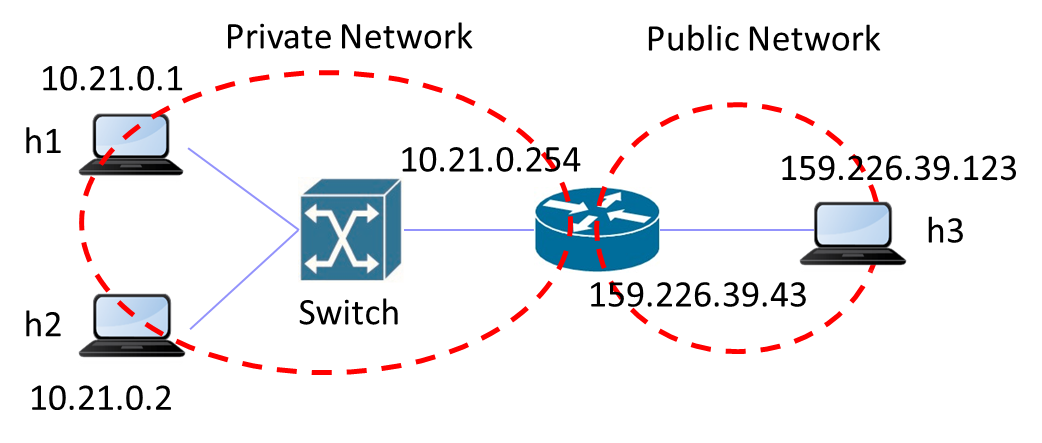
libipstack.a：路由实验实现的ARP和路由表相关函数编译出的库

http\_server.py：简单HTTP Server实现

exp[1-41].conf: NAT配置文件

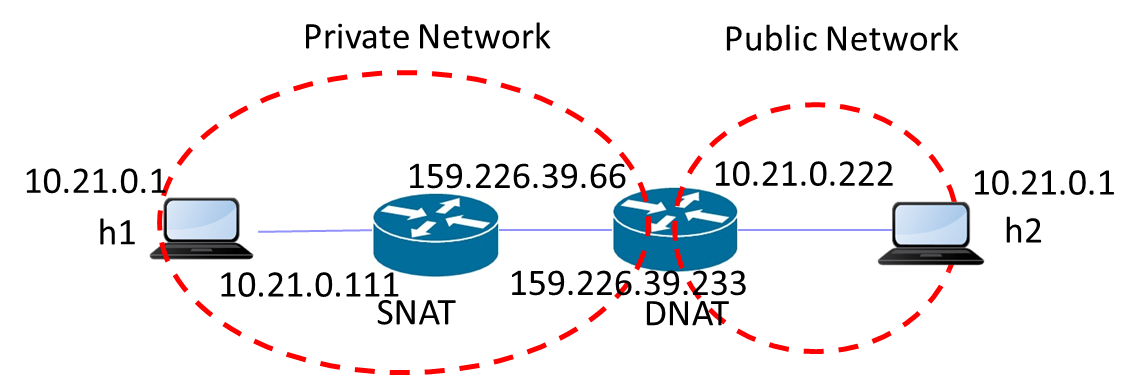
nat.c：NAT相关代码实现

nat\_topo.py：实现如下图的公-私网络拓扑



图一 公-私网络拓扑

exp3\_nat-topo.py：实现如下图的私-公-私网络拓扑



图二 私-公-私网络拓扑

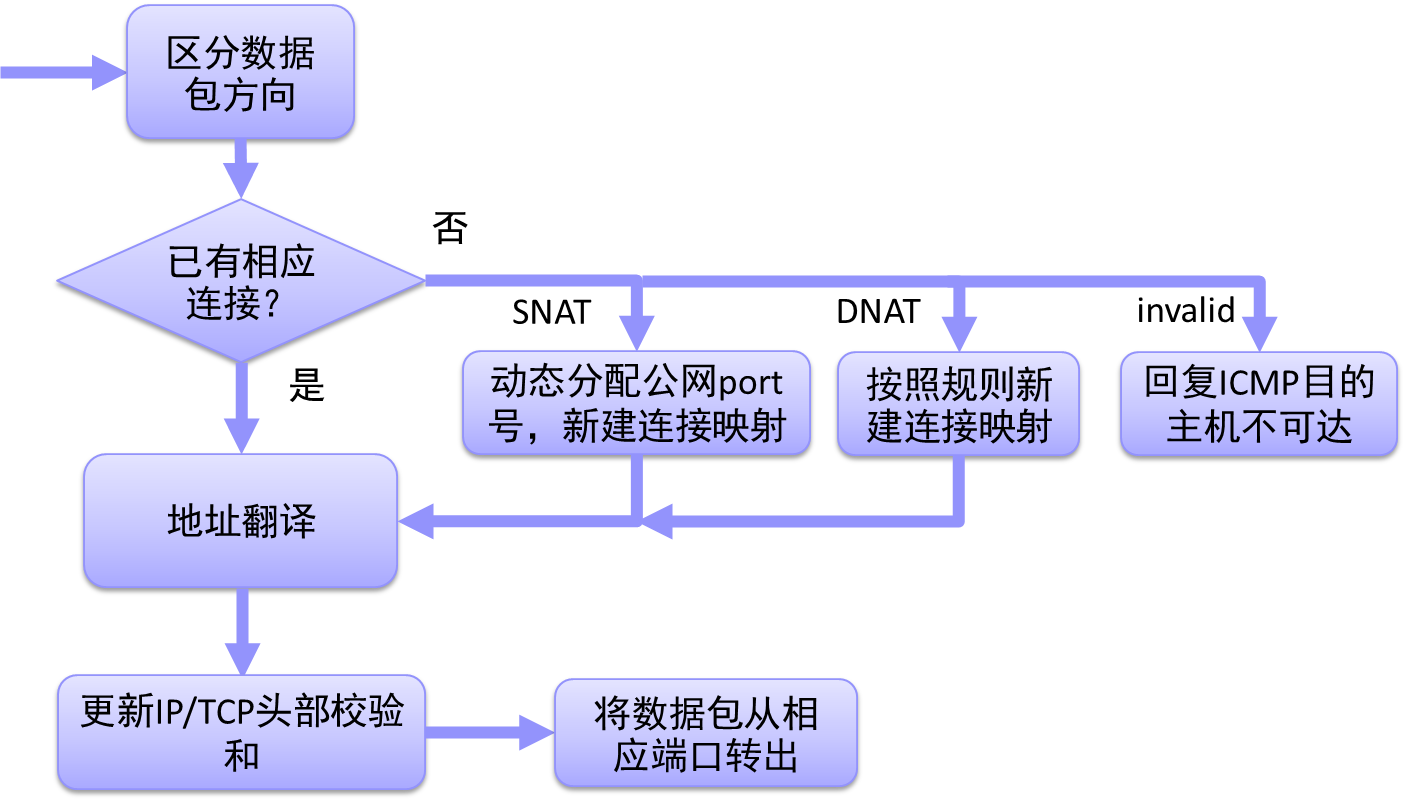
NAT：Network Address Translation，网络地址转换，也叫做网络掩蔽或者IP掩蔽（IP masquerading），是一种在IP数据包通过路由器或防火墙时重写来源IP地址或目的IP地址的技术，被普遍使用在有多台主机但只通过一个公有IP地址访问因特网的私有网络中。但是，NAT也让主机之间的通信变得复杂，导致了通信效率的降低。

NAT设备主要工作：1）维护私网地址/端口 与 公网地址/端口的映射关系2）对数据包内容进行重写（Translation），修改IP地址、端口等字段，使得数据包在相应网络中有意义。

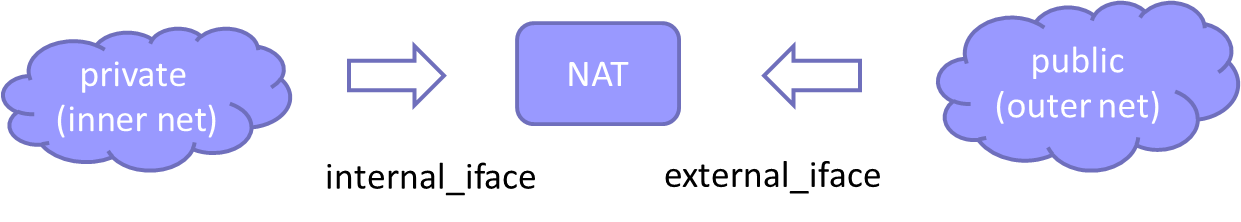
SNAT：Source Network Address Translation，源地址转换，将IP数据包的源地址转换成另外一个地址。内网主机A和外网主机B通信，A可以向B发送IP数据包，但B向A回复时，如果A使用的是内网地址，那么数据包无法到达，此时需要SNAT设备记录内网主机的A的IP，与之建立并记录连接，更换A发来的数据包的IP为路由器公网IP，为该连接保存一个端口。

DNAT：Destination Network Address Translation，目的地址转换，将一组本地内部的地址映射到一组全球地址。继续上面的例子，此后主机B与主机A交互就使用该IP和端口，DNAT设备收到来自主机B的数据包后解析，如果目的地址为路由器公网IP，端口为记录的端口，就将该数据包转发至A。

NAT工作机制：



图三 NAT工作机制

区分数据包方向：

图四 NAT工作示例

NAT设备收到一个数据包，进行数据包方向的区分：

1. 源地址为内部地址，且目的地址为外部地址时，方向为DIR\_OUT
2. 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

可以通过查询源地址和目的地址的转发表项，将转发表项的端口取出，如果源地址侧端口是内网端口，目的地址侧是外网端口，方向为DIR\_OUT；如果源地址侧是外网端口，目的地址侧是内网端口，方向为DIR\_IN。在该实验中，因为不涉及其他的路由，不满足这两项的IP数据包设置为invalid即可。

建立连接：

如果连接已经建立，跳过此步。

如果连接没有建立，进行如下操作：

SNAT的场合：如果该数据包的方向为DIR\_OUT，说明该包为内网主机到外网主机TCP连接的第一个数据包（请求连接数据包），在NAT缓存表中加入该表项，记录内网主机的内网IP、内网端口号和映射的外网IP、随机分配的外网端口号（不能为0）。

DNAT的场合：如果该数据包的方向为DIR\_IN，说明该包为外网主机到内网主机TCP连接的第一个数据包（请求连接数据包），查看DNAT是否有记录处理来自该外网主机的规则，如果有，将该规则添加到NAT缓存表中。

INVALID的场合：回复ICMP Destination Host Unreachable。

NAT地址翻译：对于已经存在的表项，进行(internal\_ip, internal\_port) <-> (external\_ip, external\_port)之间的转换

SNAT：修改IP首部的源主机地址为表项中记录的外网IP，TCP首部的源主机端口为表项中记录的外网端口号，按TCP内容修改表项是否有效（如果TCP连接结束就将表项设置为无效）和内网侧最新序列号和内网侧最新确认号，更新使用时间，更新IP和TCP的校验和。最后将数据包从表项中记录的端口转发出去。

DNAT：修改IP首部的目的主机地址为表项中记录的内网IP，TCP首部的目的主机端口为表项中记录的内网端口号，按TCP内容修改表项是否有效（如果TCP连接结束就将表项设置为无效）和外网侧最新序列号和外网侧最新确认号，更新使用时间，更新IP和TCP的校验和。最后将数据包从表项中记录的端口转发出去。

连接的维护：可以考虑使用Hash表的方式进行存储。对外网主机IP进行哈希，相同哈希值的外网主机与内网主机的映射关系保存在同一个表单下，这样一来，每当有新的数据包要翻译地址，第一层查找只需要直接拉取对应哈希值的表单，在该表单下的链表进行第二层查找即可。

连接的老化：

1. TCP连接结束时的四次握手：双方都发送FIN并回复相应ACK，说明该连接已经结束，可以将该表项删除。
2. 一方发送了RST包，说明该连接意外终止，可以将该表项删除。
3. 双方超过60秒没有进行过数据传输，认为传输已经结束，可以将该表项删除。

2.启动脚本

1)SNAT实验

make all

sudo python nat\_topo.py

mininet> xterm h1 h2 h3 n1

n1# ./nat exp1.conf

h3# python ./http\_server.py

h1# wget http://159.226.39.123:8000

h1# cat index.html

h2# wget http://159.226.39.123:8000

h2# cat index.html.1

mininet> quit

2)DNAT实验

make all

sudo python nat\_topo.py

mininet> xterm h1 h2 h3 n1

n1# ./nat exp1.conf

h1# python ./http\_server.py

h2# python ./http\_server.py

h3# wget http://159.226.39.43:8000

h3# cat index.html.2

h3# wget http://159.226.39.43:8001

h3# cat index.html.3

mininet> quit

3)双nat穿透实验

make all

sudo python exp3\_nat-topo.py

mininet> xterm h1 h2 h3 n1

n1# ./nat exp1.conf

h1# python ./http\_server.py

h2# python ./http\_server.py

h3# wget http://159.226.39.43:8000

h3# cat index.html.2

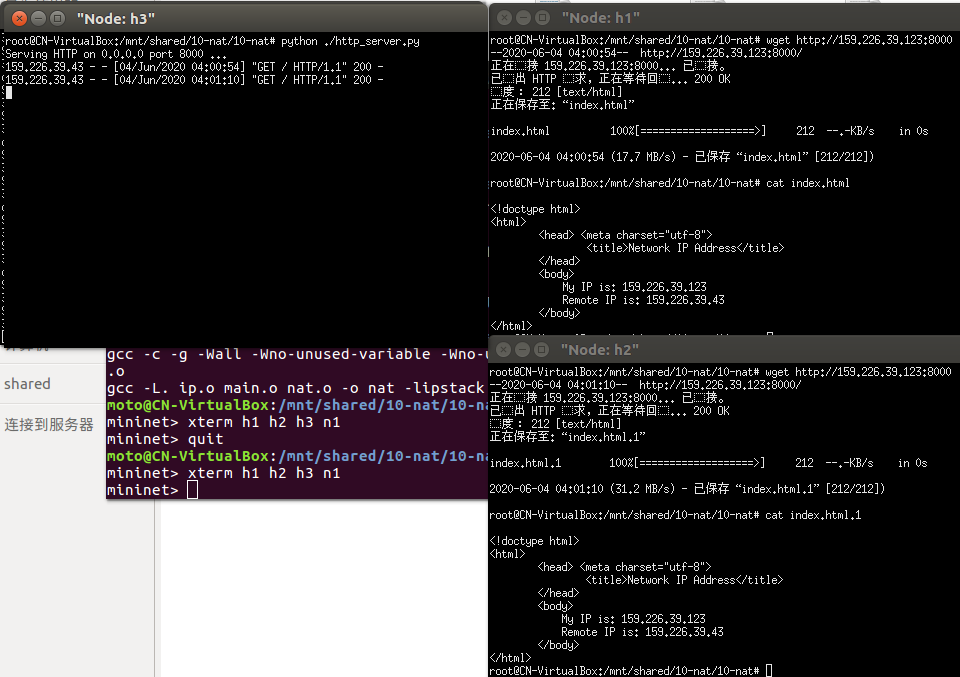
h3# wget http://159.226.39.43:8001

h3# cat index.html.3

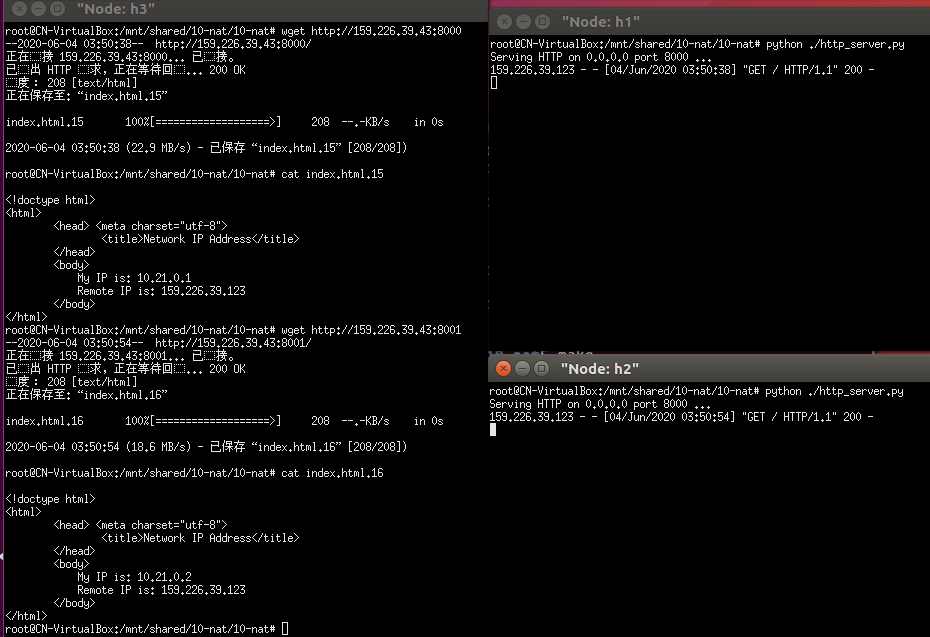
mininet> quit

三、实验结果及分析

1. 实验结果

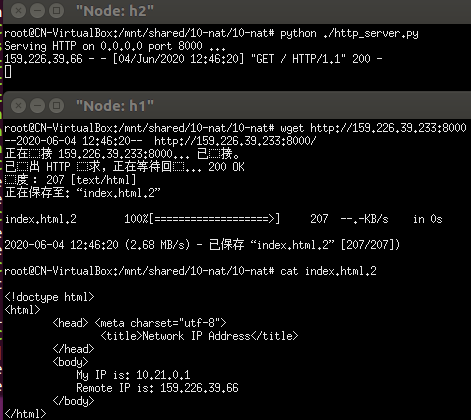


图五 SNAT实验结果

 可以看到，h1作为服务器的IP为公网159.226.39.123，h2和h3处于同一内网，共用同一个公网IP159.226.39.43

图六 DNAT实验结果

可以看到，作为服务器的h1的内网IP为10.21.0.1，作为服务器的h2的内网IP为10.21.0.2，作为客户的h3的公网IP为159.226.39.123



图七 双nat穿透实验结果

可以看到，作为服务器的h2的内网IP为10.21.0.1，虽然作为客户的h1的内网IP为10.21.0.1，但是其映射的公网IP为159.226.39.66，所以h2回应的报文中RemoteIP设置为159.226.39.66。

四、思考题

1.实验中的NAT系统可以很容易实现支持UDP协议，现实网络中NAT还需要对ICMP进行地址翻译，请调研说明NAT系统如何支持ICMP协议

答：

首先来看ICMP的报文格式：

|  |  |  |
| --- | --- | --- |
| 类型Type | 代码Code | 校验和Checksum |
| 标识符Identifer | | 序列号Seq\_num |
| Option | | |

假设A和B在局域网内，C在公网上

A：IP：192.168.0.2/24

B：IP：192.168.0.3/24

Router：IP1：192.168.0.1/24

IP2：188.10.1.2（公网地址）

C：IP：200.10.2.1

如果A想要Ping C，首先A在发送ICMP报文的时候，会根据（类型+代码）的值生成源端口号，根据标识符的值生成目的端口号，也就是说，路由器会收到A发来的如下ICMP报文：

|  |  |  |  |
| --- | --- | --- | --- |
| 源IP | 源端口 | 目的IP | 目的端口 |
| 192.168.0.2 | Type + Code | 200.10.2.1 | Identifer |

路由器进行SNAT，生成如下NAT表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 内网IP | 内网端口 | 协议 | 外网IP | 外网端口 |
| 192.168.0.2 | Type + Code | ICMP | 188.10.1.2 | IDENTIFIER |

其中IDENTIFIER是随机分配的端口号。

路由器修改ICMP报文为以下：

|  |  |  |  |
| --- | --- | --- | --- |
| 源IP | 源端口 | 目的IP | 目的端口 |
| 188.10.1.2 | IDENTIFIER | 200.10.2.1 | Identifer |

C收到ICMP报文后，生成ICMP响应报文，路由器收到的C的响应报文如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 源IP | 源端口 | 目的IP | 目的端口 |
| 200.10.2.1 | Type + Code | 188.10.1.2 | IDENTIFIER |

路由器查询NAT表，找到NAT表项，将目的IP和目的端口替换后转发ICMP响应报文即可。

2.假设SNAT设备只有一个公网IP（外部端口），请问如何修改设计，使得其可以支持多于65335个并发连接

答：可以设计每一个端口号绑定多个IP地址。一方面，当SNAT时，从内访问外网使用哪个端口由外网IP决定，所以即使多个外网IP绑定同一端口也不会引起冲突；另一方面，当DNAT时，从外访问内网使用哪个端口由目的IP和端口决定，如果绑定到目的端口上的内网IP多于一个，就同时向多个IP进行多播。不过这样一来可能会出现多个主机响应的问题。

（二）实验代码详解

本次实验代码过长，不再赘述代码的内容。

1. nat.c:NAT功能具体实现代码

（1）static int get\_packet\_direction(char \*packet)：返回收到数据包的发送方向

首先解析数据包的IP首部，获取网络到本地字节序转换的源IP地址与目的IP地址。调用最长前缀匹配查找两个IP的转发表项，获取两个转发表项的端口。如果目的IP端口为外网端口，源IP端口为内网端口，那么返回DIR\_OUT；如果源IP端口为外网端口，目的IP端口为内网端口，那么返回DIR\_IN；否则返回DIR\_INVALID。

（2）void do\_translation(iface\_info\_t \*iface, char \*packet, int len, int dir)：更换数据包的IP地址和端口号，重新计算校验和，更新TCP连接状态

首先获取NAT的互斥锁。

然后解析IP报头，如果方向为DIR\_IN就计算经网络到本地字节序转换的源IP地址的哈希值，并找到该哈希值对应的表单；否则就计算经网络到本地字节序转换的目的IP地址的哈希值，并找到该哈希值对应的表单。

如果方向为DIR\_IN，遍历找到的表单，首先看有没有外网IP、外网端口号与目的IP、目的端口号相同的表项：

如果有，就按照表项中记录的信息，修改TCP中目的端口号为经本地到网络字节序转换的内网端口号，修改IP中目的IP地址为经本地到网络字节序转换的内网IP。

如果没有，就查看DNAT规则表单。遍历表单，首先看有没有外网IP、外网端口号与目的IP、目的端口号相同的表项。如果有，就新建一个表项，将对应信息填入，调用宏list\_add\_tail添加到哈希表单中。

随后设置表项的connection状态中的外网连接结束符，外网最后序列号，外网确认序列号。

如果方向为DIR\_OUT，遍历找到的表单，首先看有没有内网IP、内网端口号与源IP、源端口号相同的表项：

如果没有，就新建一个表项，设置表项的内网IP为经网络到本地字节序转换的源IP地址，内网端口号为经网络到本地字节序转换的TCP中的源端口号，外网IP为NAT记录的外网端口的IP，然后选取一个NAT设备未使用的端口号作为表项的外网端口号。最后将该表项添加到哈希表单中。

如果有（就算没有，经过上面的步骤之后也应该有了），就将TCP中的源端口号修改为经本地到网络字节序转换的表项记录的外网端口号，IP首部中的源IP地址修改为经本地到网络字节序转换的外网IP。

随后设置表项的connection状态中的内网连接结束符，内网最后序列号，内网确认序列号。

最后，重新计算TCP和IP校验和，更新表项访问时间，释放NAT锁，调用IP\_send\_packet将数据包转发。

（3）void \*nat\_timeout()：NAT老化操作

和之前的老化操作类似，程序位于while循环中，每1秒执行一次。

首先获取NAT互斥锁，获取当前时间。

遍历所有的哈希表单：

对于一个哈希表单，如果非空，就遍历其下所有表项，如果当前时间大于表项访问时间60秒，就删除该表项；如果表项记录的TCP连接已经结束，也删除该表项。

最后释放互斥锁，sleep（1）。

（4）int parse\_config(const char \*filename)：配置NAT的初始信息

首先要调用if\_name\_to\_iface从配置文件中获取端口的配置，存放到NAT结构体中，然后配置DNAT的rule，如外网IP、外网端口号到内网IP、内网端口号的映射。

（5）void nat\_exit()：释放NAT表

首先获取NAT互斥锁，然后遍历所有的哈希表单，删除所有哈希表项，调用kill杀死NAT老化进程，释放互斥锁。