# lab6_report_nat


| 姓名 | 学号 | 邮箱 | 完成时间 |
| --- | --- | --- | --- |
| 陈嘉昀 | 211220137 | jiayunchen@smail.nju.edu.cn | 2023年12月28日 |


# 1 代码设计

## 1-1 data structure

现在使用内部ip作为hash表的key。

有一个关键问题需要注意：对于一个包，如何查询到它的mapping?

- 如果是内部向外的包，根据内部ip得到对应的链，逐个便利即可
- 如果是外部向内的包，只能通过外部端口来找到这个mapping，所以，需要修改一下数据结构，让端口也能找到mapping

```c
    u8 assigned_ports[65536];           // port pool, indicates
whether a port is assigned
    nat_mapping_t* port_2_map[65536];           // get mapping by port
```

当创建mapping的时候需要建立连接

```c
// when creating a new mapping
static void add_mapping(u32 e_ip, u16 e_port, u32 i_ip, u16 i_port,
u32 r_ip, u16 r_port) {
    u8 key = hash8((char*)(&i_ip), sizeof(i_ip));
    nat_mapping_t* r = (nat_mapping_t*)malloc(sizeof(nat_mapping_t));
    ...
    nat.port_2_map[e_port] = r;

    atomic {
        list_add_tail(&r->list, &nat.nat_mapping_list[key]);
        log(DEBUG, GREEN "add a rule: %x:%hu -> %x:%hu" CLR, e_ip,
e_port, i_ip, i_port);
    }
}
```

外部来的包也可以通过port得到这个mapping

```c
// when get an in packet
void do_translation(...) {
    ...
    // check dst port first
    u32 i_ip = 0;
    u16 i_port = 0;
    nat_mapping_t* e = nat.port_2_map[d_port];
    if(e) {
        log(DEBUG, GREEN "find old mapping: %x:%hu -> %x:%hu\n" CLR,
e->internal_ip, e->internal_port, e->external_ip, e->external_port);
    }
    ...
}
```

## 1-2 method

- add_rule: 读取配置文件，增加dnat的规则
- init_nat_comm：初始化一个nat中tcp的状态
- add_mapping：增加一个nat映射
- get_new_free_port：找到一个空闲端口，用作新的external port
- get_direction：根据包的source_ip, dst_ip和到达的端口判断方向
  - DIR_OUT: s_ip and iface's ip is in the same subnet && d_ip is not in the same subnet
  - DIR_IN: d_ip and iface's ip is in the same subnet
- do_translation:
  - dir == in: 根据port，利用port_2_map得到mapping，修改包
  - dir == out: 根据内部ip查询hash表，查看是否有可用mapping：
    - 若有，直接改写
    - 若无，创建mapping，并且让新的mapping与端口建立连接，并改写
- parse_config && timeout && nat_exit：读文件，清理，dtor

## 2 截图

## 2-1 snat

```
mininet> nodes
available nodes are:
h1 h2 h3 n1 s1
mininet> links
h1-eth0<->s1-eth1 (OK OK)
h2-eth0<->s1-eth2 (OK OK)
h3-eth0<->n1-eth1 (OK OK)
n1-eth0<->s1-eth3 (OK OK)
mininet> h3 python3 ./http_server.py &
mininet> n1 ./nat exp1.conf &
DEBUG: find the following interfaces:  n1-eth1 n1-eth0.
mininet> h1 wget http://159.226.39.123:8000
--2023-12-28 11:05:20--  http://159.226.39.123:8000/
Connecting to 159.226.39.123:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 212 [text/html]
Saving to: 'index.html'

index.html          100%[===================>]     212  --.-KB/s    in 0s

2023-12-28 11:05:20 (31.1 MB/s) - 'index.html' saved [212/212]

mininet> h2 wget http://159.226.39.123:8000
--2023-12-28 11:05:26--  http://159.226.39.123:8000/
Connecting to 159.226.39.123:8000... connected.
```

```
index.html          100%[===================>]     212  --.-KB/s    in 0s

2023-12-28 11:05:20 (31.1 MB/s) - 'index.html' saved [212/212]

mininet> h2 wget http://159.226.39.123:8000
--2023-12-28 11:05:26--  http://159.226.39.123:8000/
Connecting to 159.226.39.123:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 212 [text/html]
Saving to: 'index.html.1'

index.html.1        100%[===================>]     212  --.-KB/s    in 0s

2023-12-28 11:05:26 (26.7 MB/s) - 'index.html.1' saved [212/212]

mininet>
```

## 2-2 dnat

```
mininet> h1 ./nat exp2.conf &
DEBUG: find the following interfaces:  n1-eth1 n1-eth0.
mininet> h3 wget http://159.226.39.43:8000
--2023-12-28 11:08:51--  http://159.226.39.43:8000/
Connecting to 159.226.39.43:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 208 [text/html]
Saving to: 'index.html.4'

index.html.4        100%[===================>]     208  --.-KB/s    in 0s

2023-12-28 11:08:51 (33.5 MB/s) - 'index.html.4' saved [208/208]

mininet> h3 wget http://159.226.39.43:8001
--2023-12-28 11:08:52--  http://159.226.39.43:8001/
Connecting to 159.226.39.43:8001... connected.
HTTP request sent, awaiting response... 200 OK
Length: 208 [text/html]
Saving to: 'index.html.5'

index.html.5        100%[===================>]     208  --.-KB/s    in 0s

2023-12-28 11:08:52 (29.8 MB/s) - 'index.html.5' saved [208/208]
```

## 2-3 snat-dnat

自行配置网络如下：

```
# nat2_topo.py
class NATTopo(Topo):
    def build(self):
        h1 = self.addHost('h1')
        h2 = self.addHost('h2')
        n1 = self.addHost('n1')
        n2 = self.addHost('n2')

        self.addLink(h1, n1)
        self.addLink(n1, n2)
        self.addLink(n2, h2)

if __name__ == '__main__':
    check_scripts()

    topo = NATTopo()
    net = Mininet(topo = topo, switch = OVSBridge, controller = None)

    h1, h2, n1, n2 = net.get('h1', 'h2', 'n1', 'n2')

    h1.cmd('ifconfig h1-eth0 192.168.1.2/24')
    h1.cmd('ip route add default via 192.168.1.1 dev h1-eth0')

    n1.cmd('ifconfig n1-eth0 192.168.1.1/24')
    n1.cmd('ifconfig n1-eth1 192.168.2.1/24')

    n2.cmd('ifconfig n2-eth0 192.168.2.2/24')
    n2.cmd('ifconfig n2-eth1 192.168.3.1/24')

    h2.cmd('ifconfig h2-eth0 192.168.3.2/24')
    h2.cmd('ip route add default via 192.168.3.1 dev h2-eth0')
```

运行结果：

```
mininet> n1 ./nat nat2_snat.conf &
DEBUG: find the following interfaces:  n1-eth0 n1-eth1.
mininet> n2 ./nat nat2_dnat.conf &
DEBUG: find the following interfaces:  n2-eth0 n2-eth1.
mininet> h2 python3 ./http_server.py &
mininet> h1 wget http://192.168.2.2:8000
--2023-12-28 11:17:01--  http://192.168.2.2:8000/
Connecting to 192.168.2.2:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 207 [text/html]
Saving to: 'index.html.6'

index.html.6        100%[===================>]     207  --.-KB/s    in 0s

2023-12-28 11:17:01 (426 KB/s) - 'index.html.6' saved [207/207]

mininet>
```

# 3 感言

最后lab6真的不算简单🥹，如果不用Wireshark真的有点不好理解网络的行为，一头雾水，但抓包了就好很多。

本学期实验完结撒花🎉，感谢！