第十一周实验报告

网络地址转换实验

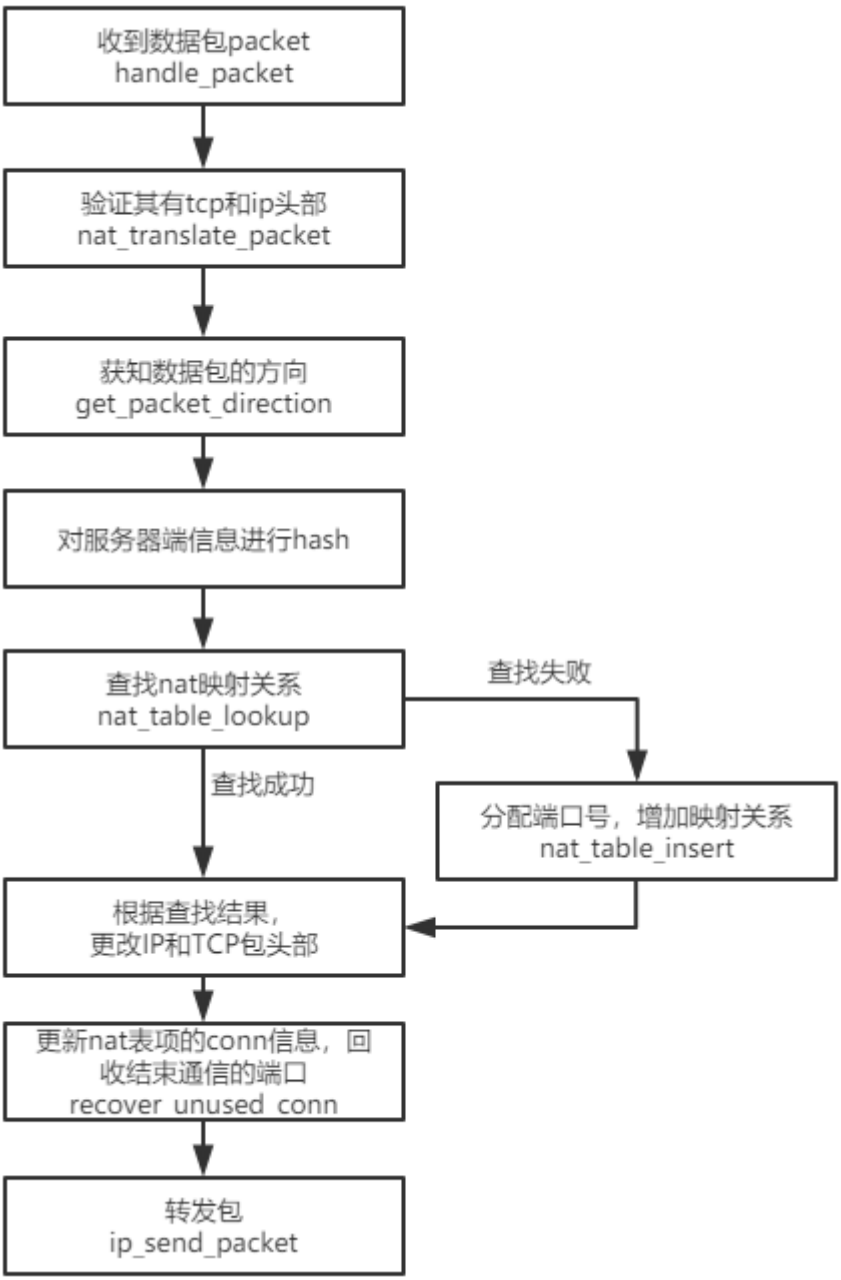
2015K8009922021
李一苇

一、实验内容

- NAT映射表管理
  - 维护NAT连接映射表，支持映射的添加、查找、更新和老化操作
- 数据包的翻译操作
  - 对到达的合法数据包，进行IP和Port转换操作，更新头部字段，并转发数据包
  - 对于到达的非法数据包，回复ICMP Destination Host Unreachable

二、实验流程

网络地址转换的流程图如下，本实验按照流程图依次实现各函数：



1. 收到数据包，验证其头部结构已经由老师实现；

2. 获知数据包的方向：

注意判断DIR_IN的条件之一是，目的IP和external_iface的IP直接相等，不用查找最长前缀了。

```
static int get_packet_direction(char *packet)
{
    struct iphdr *ip = packet_to_ip_hdr(packet);
    u32 dst = ntohl(ip->daddr);
    u32 src = ntohl(ip->saddr);
    rt_entry_t *dst_entry = longest_prefix_match(dst);
    rt_entry_t *src_entry = longest_prefix_match(src);
    if ((src_entry->iface->ip == nat.internal_iface->ip) && (dst_entry->iface->ip ==
nat.external_iface->ip)) {
        return DIR_OUT;
    }
```

```
    if ((src_entry->iface->ip == nat.external_iface->ip) && (dst == nat.external_iface-
>ip)) {
        return DIR_IN;
    }
    return DIR_INVALID;
}
```

3. hash操作

```
u32 srv_ip = dir == DIR_OUT? ntohl(ip->daddr) : ntohl(ip->saddr);
u16 srv_port = dir == DIR_OUT? ntohs(tcp->dport) : ntohs(tcp->sport);
memcpy(srv_info, &srv_ip, 4);
memcpy(srv_info + 4, &srv_port, 2);
u8 hash_srvinfo = hash8(srv_info, 6);
```

当方向为DIR_IN时，服务器的信息为源地址；当方向为DIR_OUT时，服务器信息为目的地址，拼接后得到hash值

4. 查找nat表项：

   表头为`&nat.nat_mapping_list[srv_hash]`

   当方向为DIR_IN时，应匹配外部的IP和端口

   当方向为DIR_OUT时，应匹配内部的IP和端口

   ```
   list_for_each_entry(mapping_entry, head, list) {
           if (dir == DIR_OUT) {
               if (mapping_entry->internal_ip == ip && mapping_entry->internal_port ==
   port) {
                   return mapping_entry;
               }
           }
           else {
               if (mapping_entry->external_ip == ip && mapping_entry->external_port ==
   port) {
                   return mapping_entry;
               }
           }
       }
   ```

5. 插入nat表项

   新建mapping_entry后插入，注意新表项的外部IP为`nat.external_ip`，外部端口值为`assign_external_ports()`得到的值

6. 修改IP和TCP包头部

   具体地说，当方向为DIR_IN时，修改ip的daddr和tcp的dport；否则修改ip的saddr和tcp的sport

   与此同时还应修改tcp和ip的checksum

7. 更新nat_mapping表项

   需更新`entry->update_time = time(NULL);`

   更新连接状态：

```
void recover_unused_conn(struct nat_mapping *pos, struct tcphdr *tcp, int dir){
    if(tcp->flags == TCP_FIN + TCP_ACK) {
        if (dir == DIR_IN)
            pos->conn.internal_fin = 1;
        else
            pos->conn.external_fin = 1;
    }

    if(tcp->flags == TCP_RST){
        printf("RST!\n");
        if((pos->conn).internal_fin + (pos->conn).external_fin == 2){
            nat.assigned_ports[pos->external_port] = 0;
            list_delete_entry(&(pos->list));
            free(pos);
        }
    }
}
```

8. 发送包：由`ip_send_packet()`直接实现。

三、实验结果和分析

验证方法（运行脚本`nat_topo.py`）：

- 在n1上nat，进行数据包的处理
- 在h2上运行HTTP服务
  - 执行`python -m SimpleHTTPServer`， 启动HTTP服务
- 在h1上访问h2的HTTP服务
  - h1 # wget http://159.226.39.123:8000

结果如下：

h1和h2成功建立TCP连接，收到h2的数据包并保存

抓包截图如下：

h1的分组1表明发出TCP的SYN请求，与此同时h2的分组3收到该请求，发现分组3的源IP地址变为NAT设备的外部interface的IP地址，且源端口变为12345（即最小的可分配端口值）

h2的分组4表明回应该请求（SYN，ACK），h1的分组4表明收到该回应，且该分组的目的地址经由NAT设备转换，变为h1的IP地址

之后的过程与上文类似，可见NAT设备成功进行了地址的转换。