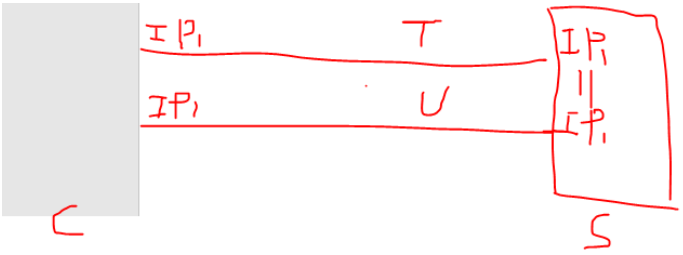
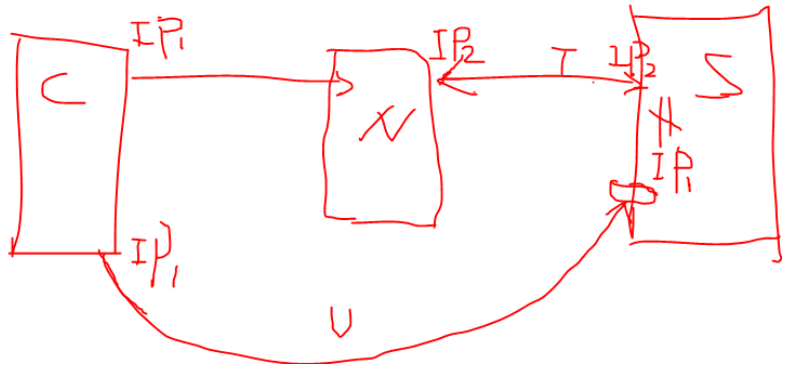
3Proxy 新增前置代理工具导致 UDP 不可用问题分析

问题背景：
在知名 Socks5 代理软件 3Proxy 中，由于该软件对于 socket 端口的配置选项有限，在某些时候并不满足生产的需求，因此为了对端口实现更精细的配置，可能需要在 3Proxy 之前增加如 Nginx 之类的**前置代理工具**（下统称前置工具），或是做端口映射，或是做负载均衡，然而经测试发现，新增前置工具后，TCP 代理正常，UDP 代理不可用。

背景知识：
Socks5 的 TCP 代理和 UDP 代理不同之处在于，在 TCP 代理中，所有的通信都是通过 TCP 口，在完成握手协议后，服务端就只做透传，而 UDP 代理则不同，在完成握手协议之后，服务端会新开一个 UDP 口用于传输 UDP 数据，该 UDP 口随机生成，并在 TCP 握手协议中告知客户端，从中可以看出，在代理 TCP 的时候，服务端和客户端只建立一个连接（TCP），在代理 UDP 的时候，服务端和客户端建立了两个连接（TCP+UDP），UDP 代理中的 TCP 连接在完成握手之后就没用了（不用于传输数据，但用于检测客户端的活跃状态，因为 UDP 是无状态的，从 UDP 无法知道客户端是否 active，所以客户的活跃状态依赖于 TCP）。

问题分析：
在为什么用 Nginx 做转发之后，UDP 不可用了呢，原因有：
原因一：Nginx 做端口映射，只转发 TCP 数据，新开的 UDP 口是客户端和 Socks5 服务器协商，Nginx 是不知道的，在 UDP 口的协商阶段，Socks5 服务器会将本机的 IP+UDP 口告诉客户端，客户端向该地址发起请求，而不是向 Nginx 发起请求。
原因二：Nginx 做转发的时候，是直接使用内网地址，比如 127.0.0.1 和 Socks5 服务器通信，则 Socks5 服务器所开的 UDP 端口将绑定在 127.0.0.1 上，则该 UDP 口将无法被外部访问到，也就代理不了 UDP 了。
原因三：为了避开**原因二**中的问题，Nginx 可以使用外网地址和 Socks5 服务器通信，那么 Socks5 服务器新开的 UDP 口将绑定在外网地址上，此时客户端确实可以将数据直接发送到该新开的 UDP 口，但是服务端在收到数据之后会去比对建立 TCP 连接的 IP 和收到 UDP 数据的 IP 是否一致，如果不一致将直接断开 TCP 连接并关闭新开的 UDP 口。





解决办法：

通过以上分析，解决办法如下：

1. Nginx 使用外网地址和 Socks5 服务器通信
2. 修改 Socks5 服务器，当收到数据 UDP 数据时，如果比对 IP 不一致，不报错，对于 3Proxy 来说，就是屏蔽掉图中所示的比对 IP 代码。

```
}
if (fds[1].revents) {
    sasize = sizeof(sin);
    printf("%s-%d\n", __FILE__, __LINE__);
    if((len = so._recvfrom(param->clisock, (char *)buf, 65535, 0, (struct sockaddr *)&sin, &sasize)) <= 10) {
        param->res = 464;
        break;
    }
    //if(SAADDRLEN(&sin) != SAADDRLEN(&param->sincr) || memcmp(SAADDR(&sin), SAADDR(&param->sincr), SAADDRLEN(&sin))){
    //  param->res = 465;
    //  break;
    //}
    if(buf[0] || buf[1] || buf[2]) {
        param->res = 466;
        break;
    }
    size = 4;
    switch(buf[3]) {
        case 4:
```

测试结果：

经测试，以上方案可行。

剩余问题：

如果有人恶意扫描 UDP 口，往正在通信的 UDP 口发送数据，那么将导致异常数据被发送到目标服务器，如果目标服务器没有容错功能，将导致通信异常（服务端报错或断链），这个漏洞将成为恶意攻击的一个手段。