

IP in IP Tunnel

1 介绍

实现了一个简单的 ipip tunnel 模拟，基于 UDP，IPv4（包括分片），RAW_SOCKET 二次通信。

为了方便实现，实现的是 IPv4 in IPv4，并且为了区分，将 inner IP-version 设置为 5，下称为 IPv5，以进行区分。

作为很粗糙的实现，这里假设了很多条件，给出场景：

PC1 位于 net-A 中，R1 是 net-A 的路由器，PC2 位于 net-B 中，R2 是 net-B 的路由器。

假设 PC1 和 PC2 进行通信，实际场景中，如：PC1 ping PC2，涉及 ARP 协议的实现。这里简单处理，指定 MAC 地址进行通信。

程序在 linux 上，基于 raw_socket 进行数据传输。指定 PC1 向 MAC_R1 发送帧 frame，如果是需要转发到 net-B 中的数据，R1 将数据转发到 MAC_R2，R2 再进一步将数据转发给 PC2。

这里只模拟 PC1 发送到 PC2 的数据包，即只模拟跨子网的数据传输，且规定子网内部全部使用 IPv5 进行通信，但是两个子网之间只能通过 IPv4 进行通信。

需要进一步封装IPIP，并通过 tunnel 传输到另一个子网的场景如下：

1. mac 地址和本机 mac 地址匹配
2. 源 ip 位于自己所在的子网
3. 目的 ip 位于对端所在的子网
4. ip 的 version 为 5（需要封装为IPv4）

需要将接受的数据解封装，在内部子网中转发的场景如下：

1. mac 地址和本机 mac 地址匹配
2. 源 ip 位于对端所在的子网
3. 目的 ip 位于自己所在的子网
4. ip 的 version 为 4（需要封装为IPv5）

2 启动

这里在同一个 linux 环境下进行模拟。开启四个终端，分别执行：

```
1 root@pc1:$make client1run
2 root@pc2:$make client2run
3 root@pc3:$make router1run
4 root@pc4:$make router2run
```

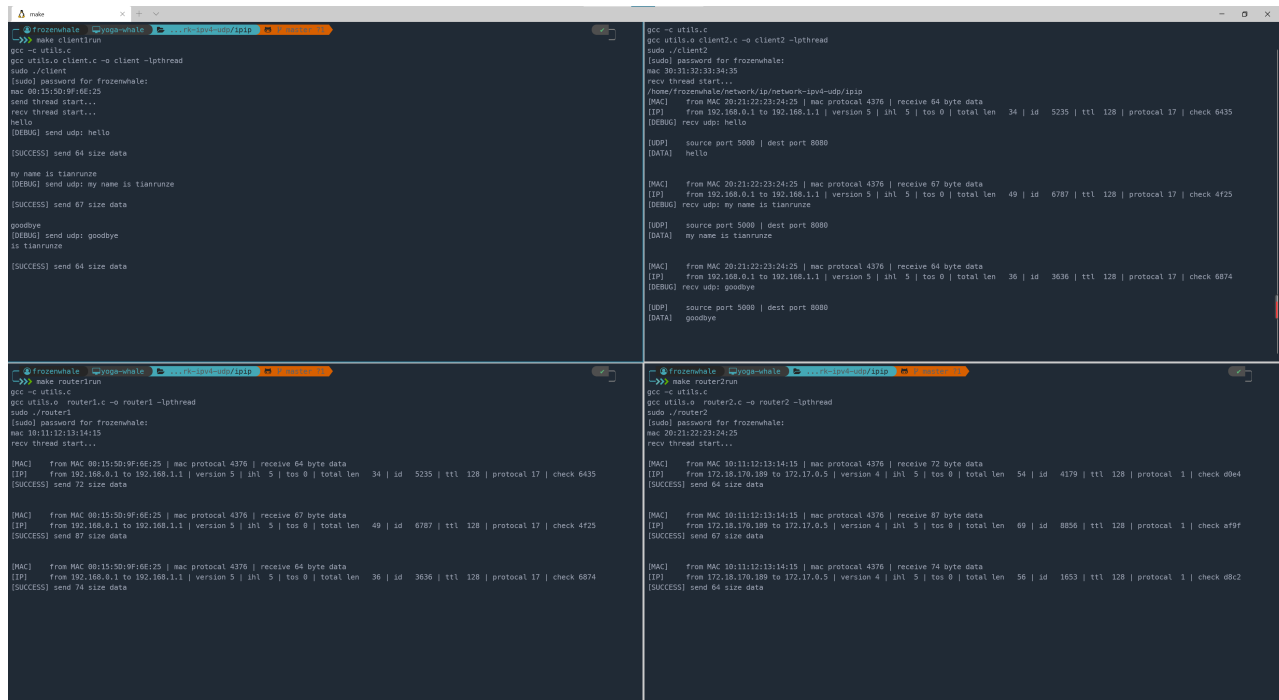
3 流程

- PC1 中输入要发送的数据
 - 数据将会被封装为 UDP
 - 封装为 IPv5
 - 封装为 Ethernet
 - 通过 RAW_SOCKET 发送给 R1（省略 ARP）
- R1 接受数据
 - 查看 Ethernet 的目的 MAC 地址，与本地 MAC 地址匹配
 - 解析为 IPv5
 - 发现版本号为 5 并且源 IP 位于子网 net-A，目的 IP 位于子网 net-B 中，封装为 IPv4
 - 封装为 Ethernet
 - 通过 RAW_SOCKET 发送给 R2（省略 ARP）
- R2 接受数据
 - 查看 Ethernet 的目的 MAC 地址，与本地 MAC 地址匹配
 - 解析为 IPv4
 - 发现版本号为 4 并且源 IP 位于子网 net-A，目的 IP 位于子网 net-B 中，解析为 IPv5
 - 封装为 Ethernet
 - 通过 RAW_SOCKET 发送给 PC2（省略 ARP）
- PC2 接受数据
 - 查看 Ethernet 的目的 MAC 地址，与本地 MAC 地址匹配
 - 解析为 IPv5
 - 解析为 UDP
 - 打印数据部分 payload

由于没有在实际环境中进行仿真，因此 IP 的 TTL 字段，在这里没有实际作用。实际上，应当在每一跳中，更新外层 IP 的 TTL 字段。

4 测试

如下图所示，在 wsl 环境中启动。



The image displays four terminal windows arranged in a 2x2 grid, showing the execution of network tests in a WSL environment. Each window has a title bar with 'frozenshale' and 'yogeshwale' icons, and a tab labeled 'work-ipv4-udp/icmp'.

Top-Left Window: Shows the execution of a client program. The user runs `gcc -c utils.c` and `gcc utils.o client.c -o client -lthread`. The program is run with `sudo ./client`. It sends a UDP packet with data 'hello' and receives a response 'my name is tianrunze'.

Top-Right Window: Shows the execution of a server program. The user runs `gcc -c utils.c` and `gcc utils.o client2.c -o client2 -lthread`. The program is run with `sudo ./client2`. It receives a UDP packet with data 'hello' and sends a response 'my name is tianrunze'.

Bottom-Left Window: Shows the execution of a router program. The user runs `gcc -c utils.c` and `gcc utils.o router1.c -o router1 -lthread`. The program is run with `sudo ./router1`. It receives a UDP packet with data 'hello' and sends a response 'my name is tianrunze'.

Bottom-Right Window: Shows the execution of a router program. The user runs `gcc -c utils.c` and `gcc utils.o router2.c -o router2 -lthread`. The program is run with `sudo ./router2`. It receives a UDP packet with data 'hello' and sends a response 'my name is tianrunze'.