|  |  |  |
| --- | --- | --- |
| 实验目的 | 本实验主要目的是设计和实现一个简单的虚拟专用网络的机制， 并与已有的标准实现（如 PPTP）进行比较， 进一步理解 VPN 的工作原理和内部实现细节。 | |
| 数据结构说明 | 参考实验手册，设置如下数据结构  //the information of the static routing table  struct route\_item{  char destination[16];  char gateway[16];  char netmask[16];  int interface;  }route\_info[MAX\_ROUTE\_INFO];  //the informaiton of the " my arp cache"  struct arp\_table\_item{  char ip\_addr[16];  char mac\_addr[18];  }arp\_table[MAX\_ARP\_SIZE];  // the storage of the device  struct device\_item{  int interface;  char ip\_addr[16];  char mac\_addr[18];  int is\_entrance;  }device[MAX\_DEVICE];  其他头文件数据结构参考  https://blog.csdn.net/cqkxboy168/article/details/9121959  https://blog.csdn.net/u012566181/article/details/38562475 | |
| 配置文件说明 | VPNServer1  10.0.0.2  10.0.0.2  255.255.255.0  0  172.0.0.2  192.168.0.1  255.255.255.0  1  192.168.0.0  0.0.0.0  255.255.255.0  1  10.0.0.2  00:0c:29:0b:46:1a  192.168.0.1  00:0c:29:f6:ab:94  0  10.0.0.1  00:0c:29:ff:97:a8  1  1  192.168.0.2  00:0c:29:ff:97:b2  1 | VPNServer1  10.0.1.2  10.0.1.2  255.255.255.0  1  192.168.0.2  172.0.0.1  255.255.255.0  0  172.0.0.0  0.0.0.0  255.255.255.0  0  10.0.1.2  00:0c:29:1a:c8:2c  172.0.0.1  00:0c:29:f6:ab:9e  0  172.0.0.2  00:0c:29:0e:ad:83  1  1  10.0.1.1  00:0c:29:0e:ad:8d  1 |
|  | |
| 程序设计的思路以及运行流程 | 在 VPN 接入点上，就是将 VPN 包进行重新封装，并传递到网络上。直到某个 VPN 接入点接收到该包，解析包头确定其确是属于某个 VPN 的包，然后将其重新解包并传递到内部的 VPN 节点上。 | |
| 运行结果截图 | 用PC1 ping PC2，下图是启动VPNServer1和VPNServer2的结果：    下图是关闭VPNServer2的结果，可以看到只有刚开始的5个数据包被接收到 | |
| 相关参考资料 | 1.  头文件  <https://blog.csdn.net/cqkxboy168/article/details/9121959>  <https://blog.csdn.net/u012566181/article/details/38562475>  2.  <https://www.cnblogs.com/zhangshenghui/p/6097492.html>  <https://www.cnblogs.com/embedded-linux/p/7073932.html>  <https://www.cnblogs.com/peida/archive/2013/03/05/2943698.html> | |
| 代码个人创新及思考 | 充分考虑了在重新封装和解封装过程的复杂性，并进行优化 | |