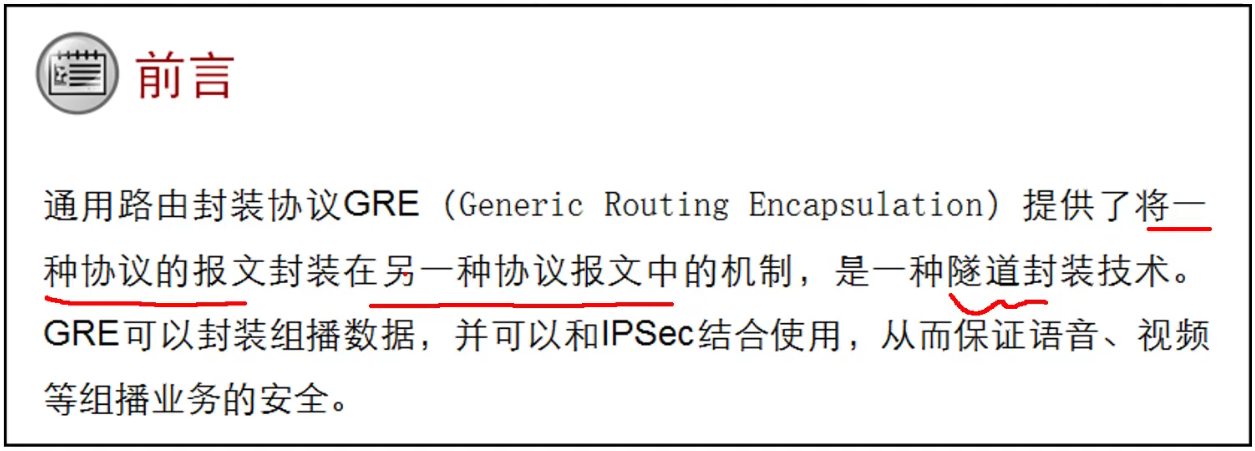
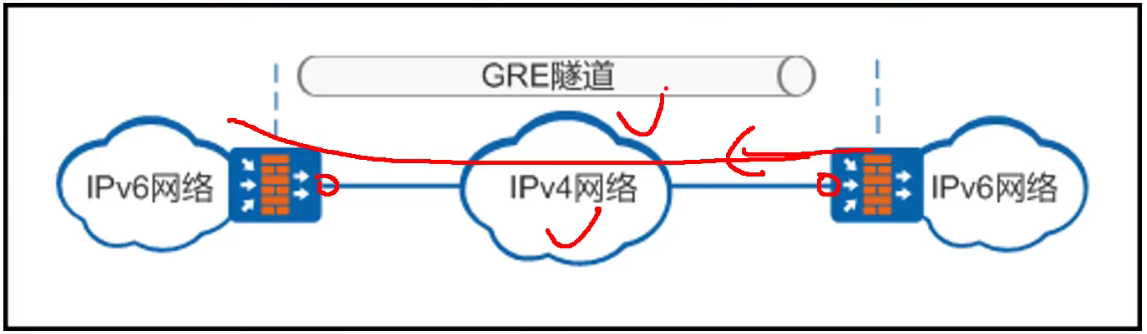
* 
* GRE简称通用路由封装协议（Generic Rrouting Encapsulation）他的作用是把一种协议的报文封装在另外一种协议的报文中的技术，是一种隧道技术（只要这样子做，都称为隧道技术）。所以我们抓包看到的至少有两个头部，这两个头部的作用就是穿越异种网络。而且他还可以对组播报文进行封装，这是IPsec做不到的，也就是说我们可以在GRE上面跑路由协议（可以让两个异地的设备跑动态路由），并且可以和IPsec结合，保证更高的安全性。
* GRE虽然我们把它称为VPN技术，但是他一般来说不会单独使用，因为他缺乏安全性，但是他的优点很明显，非常的简单，成为来隧道技术的一个代表。

GRE：Generic Rrouting Encapsulation，通用路由封装协议。

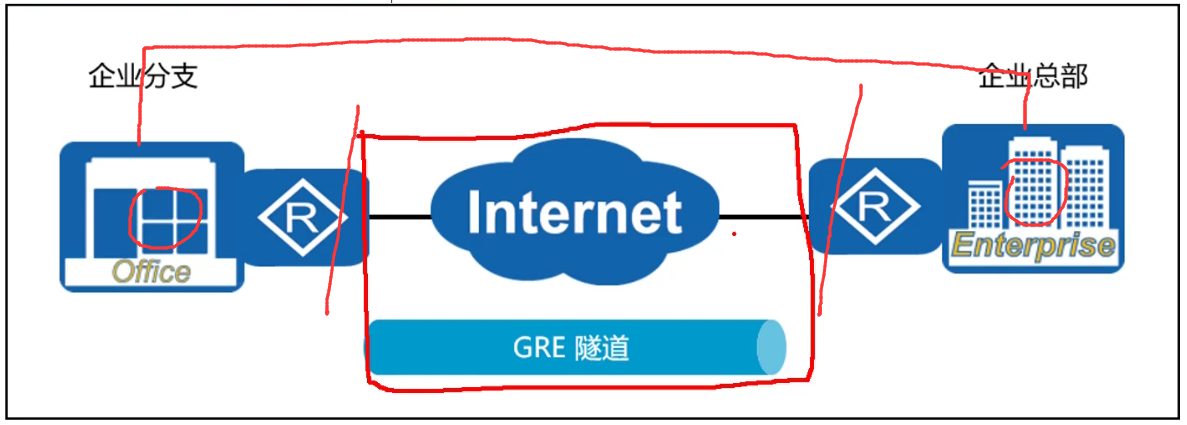
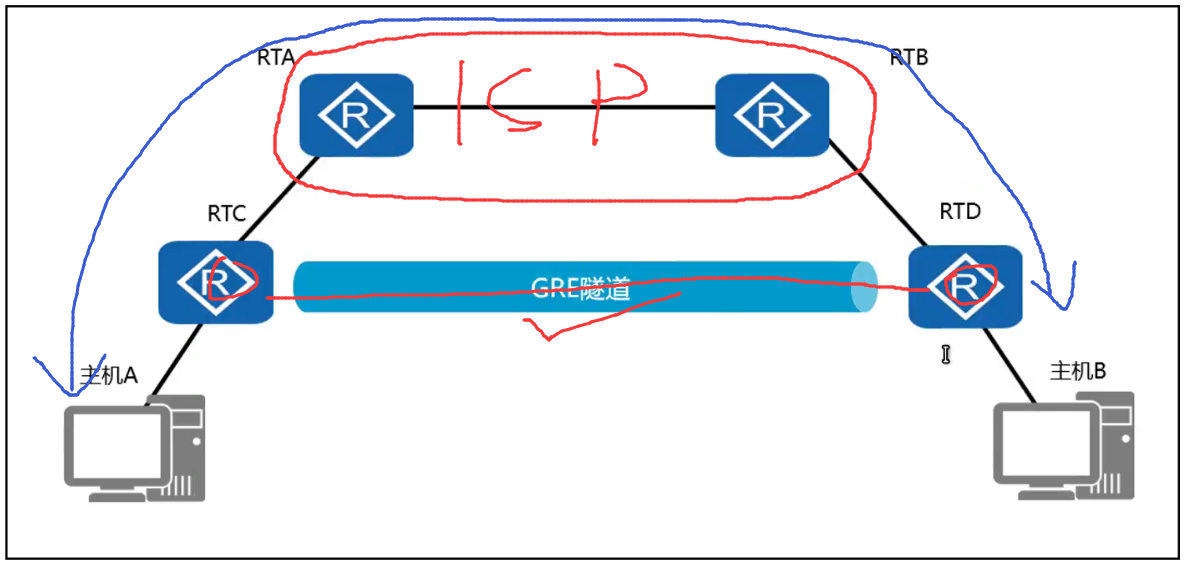
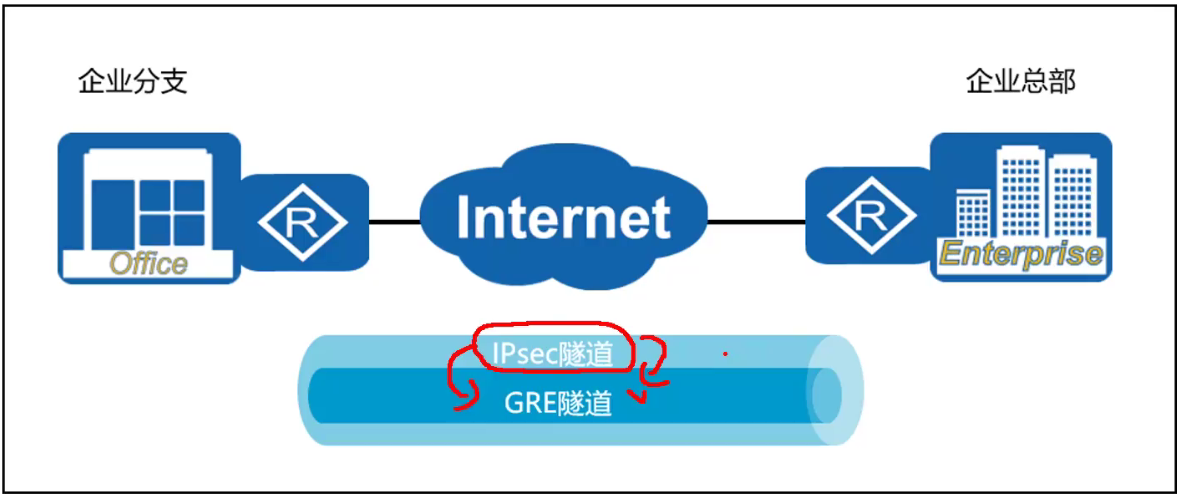
* 一种三层的VPN封装技术。
* 在任意一种网络协议上传送任意一种其他网络协议的封装方法。
* 解决了跨越异种网络（不是一种网络）的报文传输问题，异种报文传输的通道被称为隧道（Tunnel）。
* 异种网络：不是同一种网络，这些网络默认是不兼容的，通讯是有问题的。

如：私网跨越公网到达私网，IPv4跨越IPv6到达IPv4，反之亦然。本身这个技术用到的方面很广，是一种隧道技术。

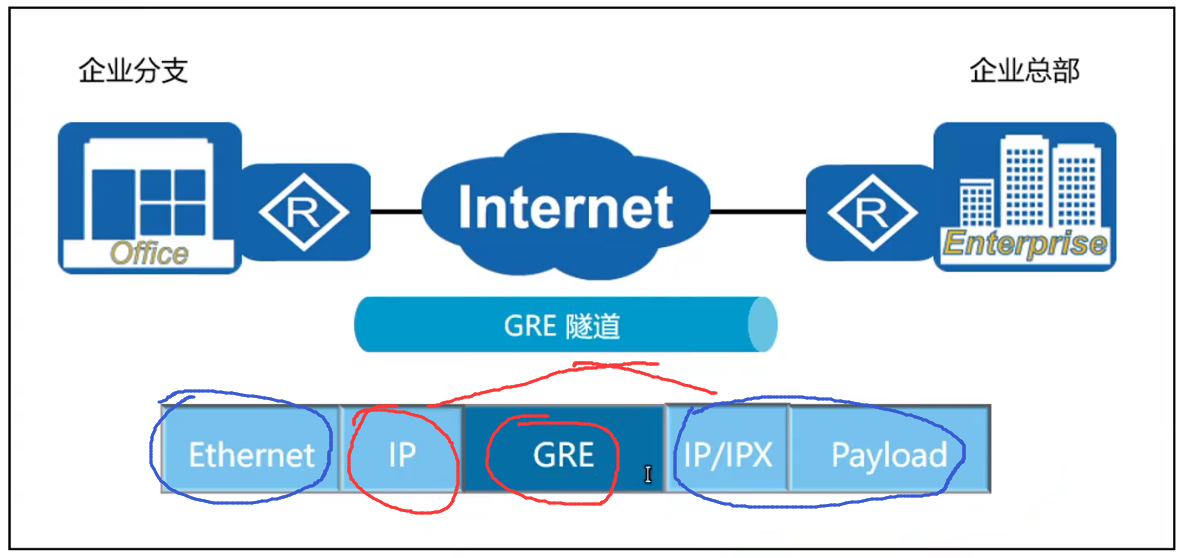
* 

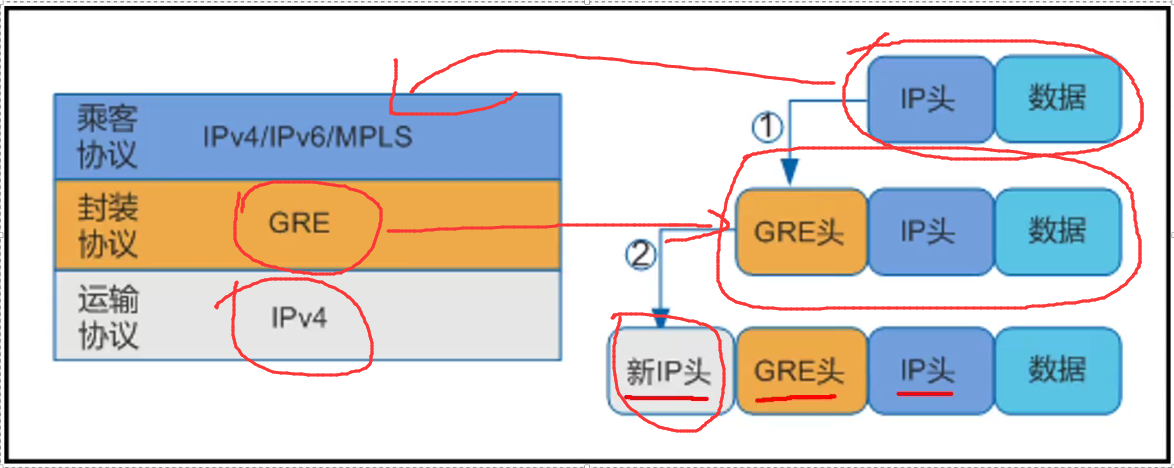
|  |  |
| --- | --- |
| GRE优点 | GRE缺点 |
| 建立隧道 | 点到点隧道  （一个点到一个点，多个点就要多个隧道口） |
| 支持多种网络层协议 | 静态配置隧道参数 |
| 支持组播路由 | 部署负责，连接关系时代价巨大 |
| 配置简单，容易部署 | 缺乏安全性（最大的致命伤） |

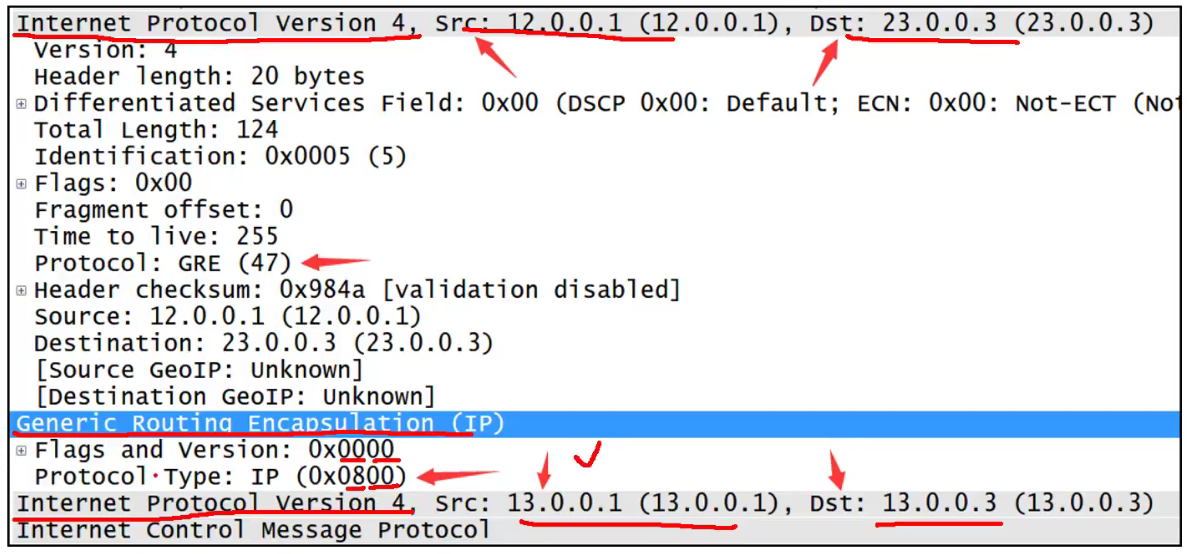
GRE核心功能：建立隧道，打通私网

* 隧道一般配置在出口且有公网的设备上。
* 
* 有了隧道以后，相当于在出口设备和对端出口设备接了一条专线，因为VPN就是用来代替专线的，可以理解为他是一条虚拟的专线，而且从路由层面上来说，正常我们数据从出口设备发出，到达目的网络会有多跳，而我们建立VPN后，只会算一跳，这些细节也会被隐藏。相当于也是减少了路径的跳数。
* 
* GRE over IPsec是实际应用场景较多的，GRE隧道一般很少单独使用，因为不安全， 所有用IPsec来保护，当然IPsec也可以单独使用。IPsec虽然很强大，有自己的隧道和加密技术，但是IPsec不支持路由组播，而GRE支持，所以使用IPsec保护GRE进行安全传输。所以是先有GRE然后再由IPsec来保护GRE，这也就是GRE over IPsec，这是实际需求的一种用法。
* 

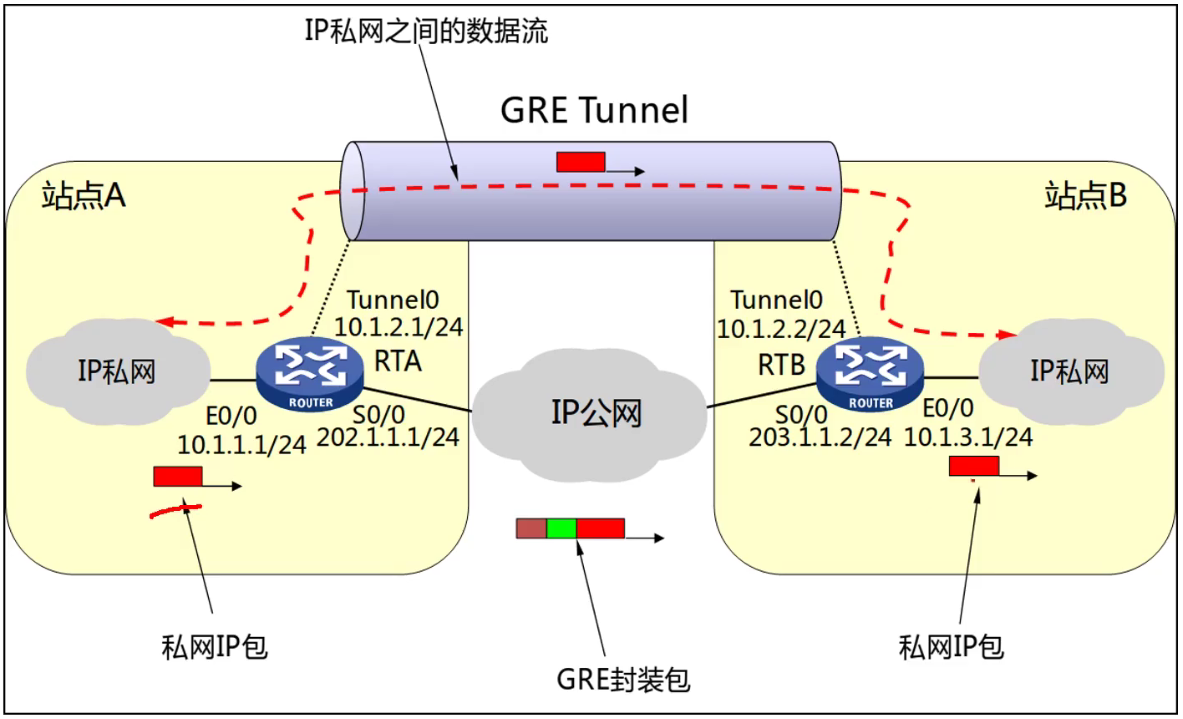
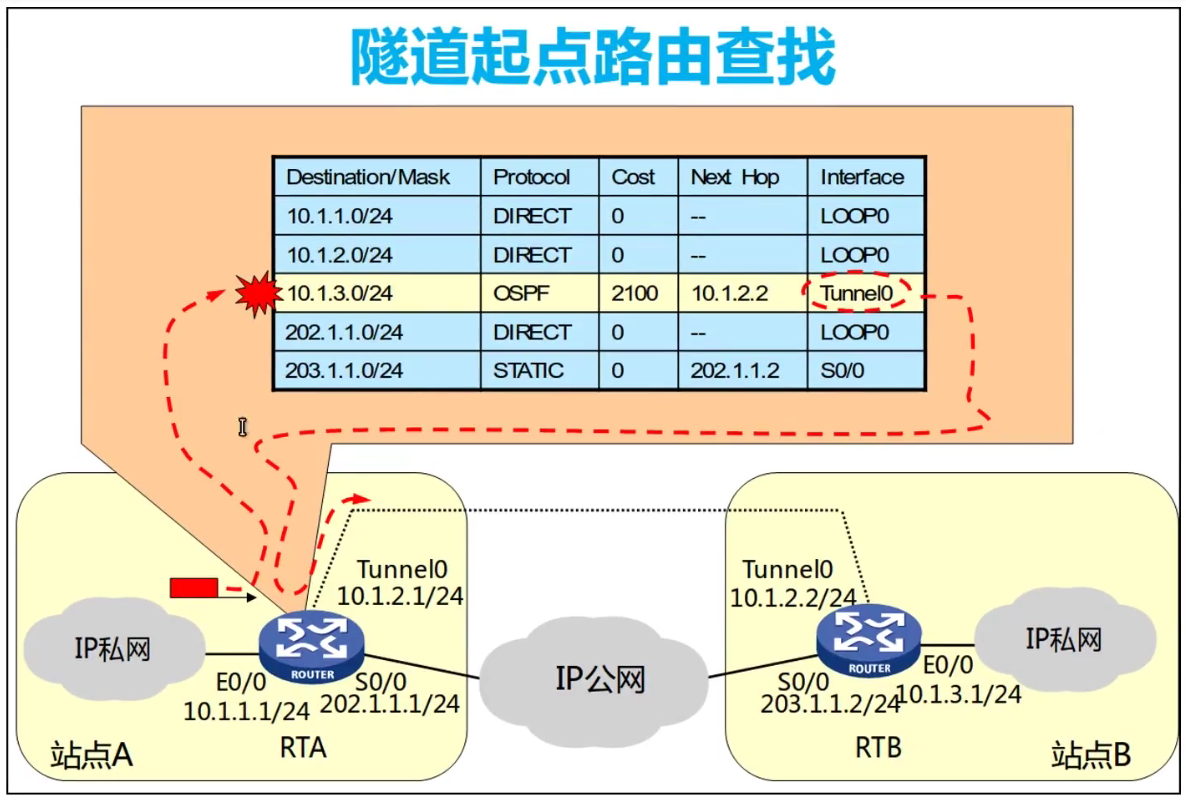
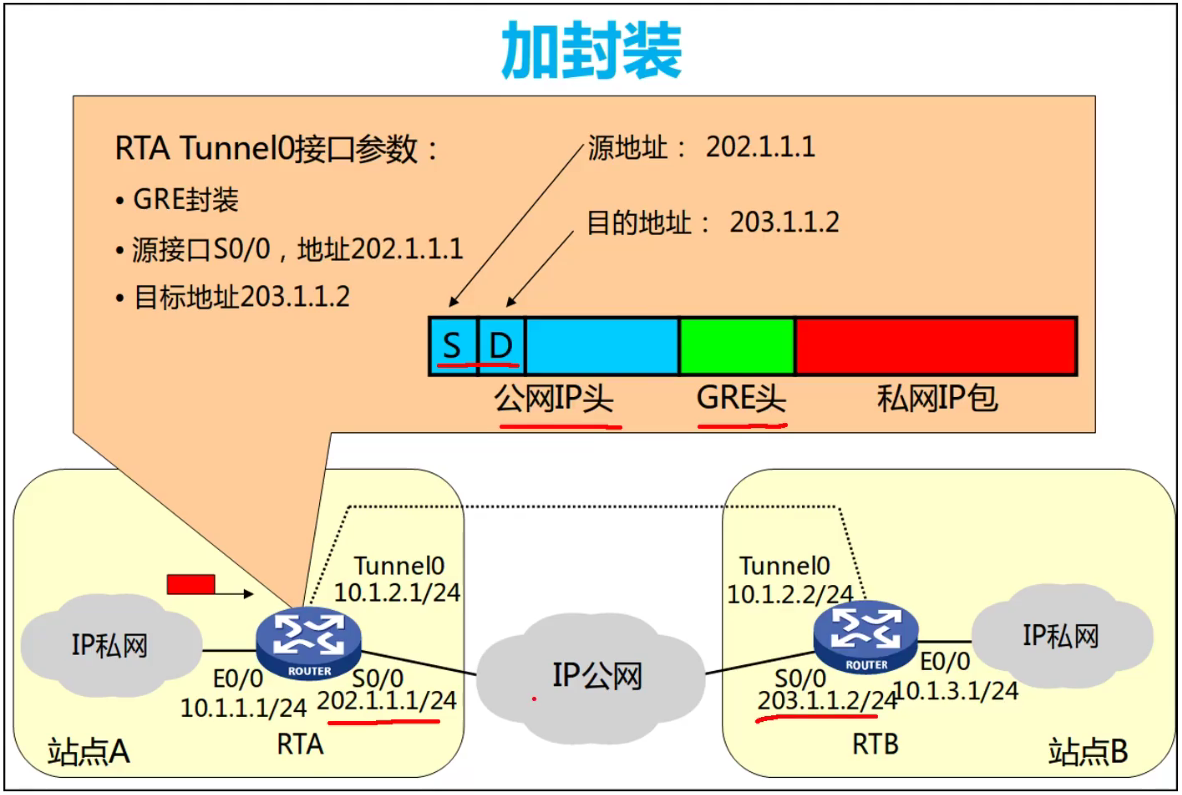
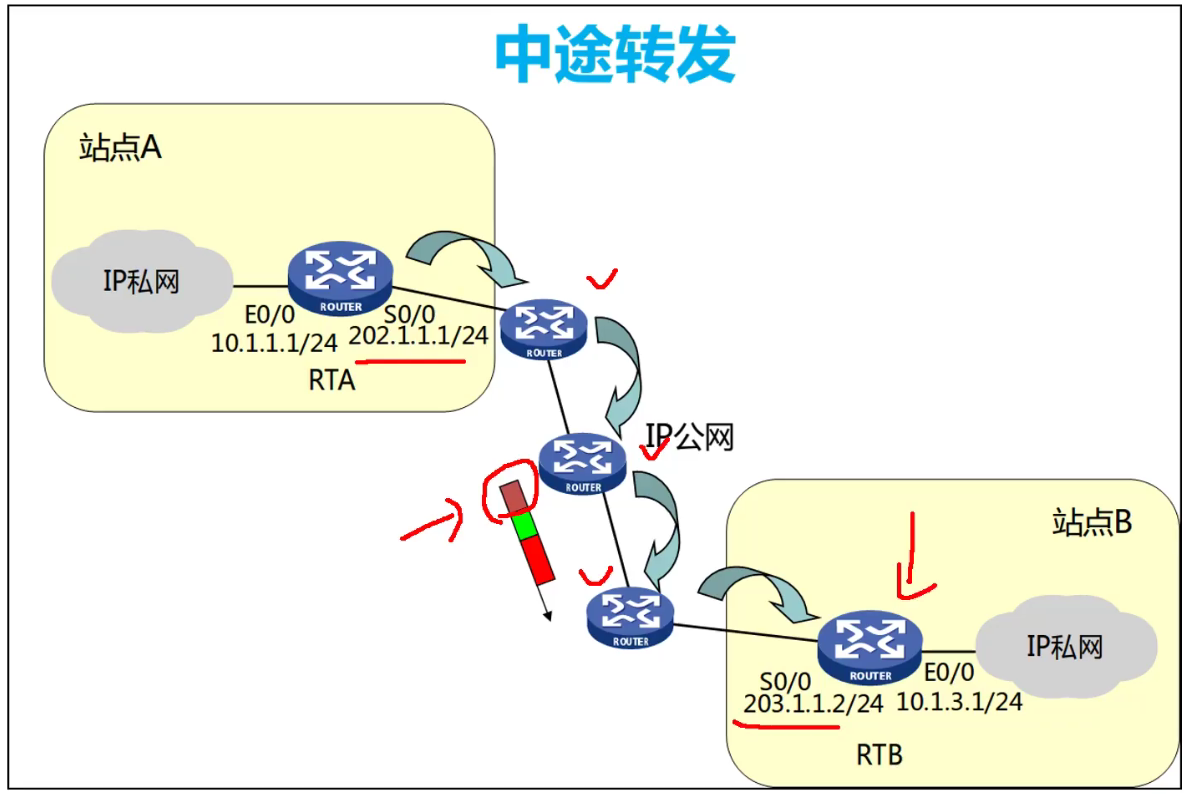
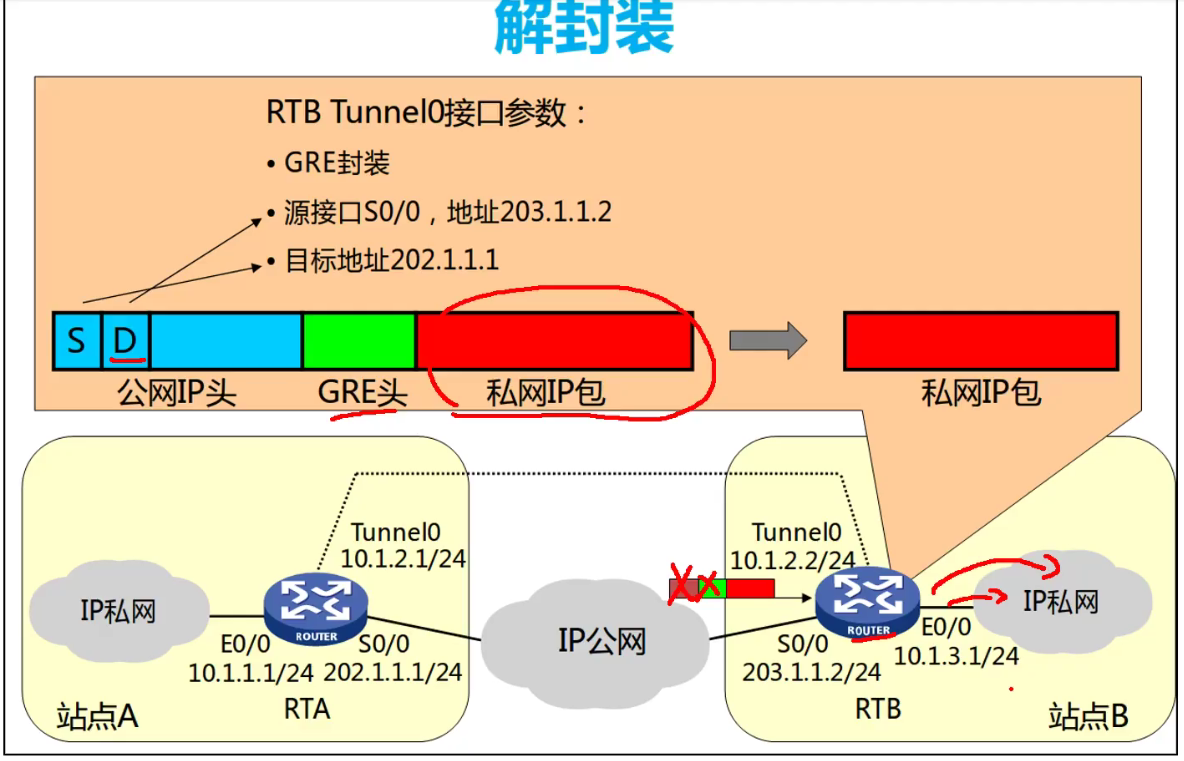
GRE封装：穿马甲

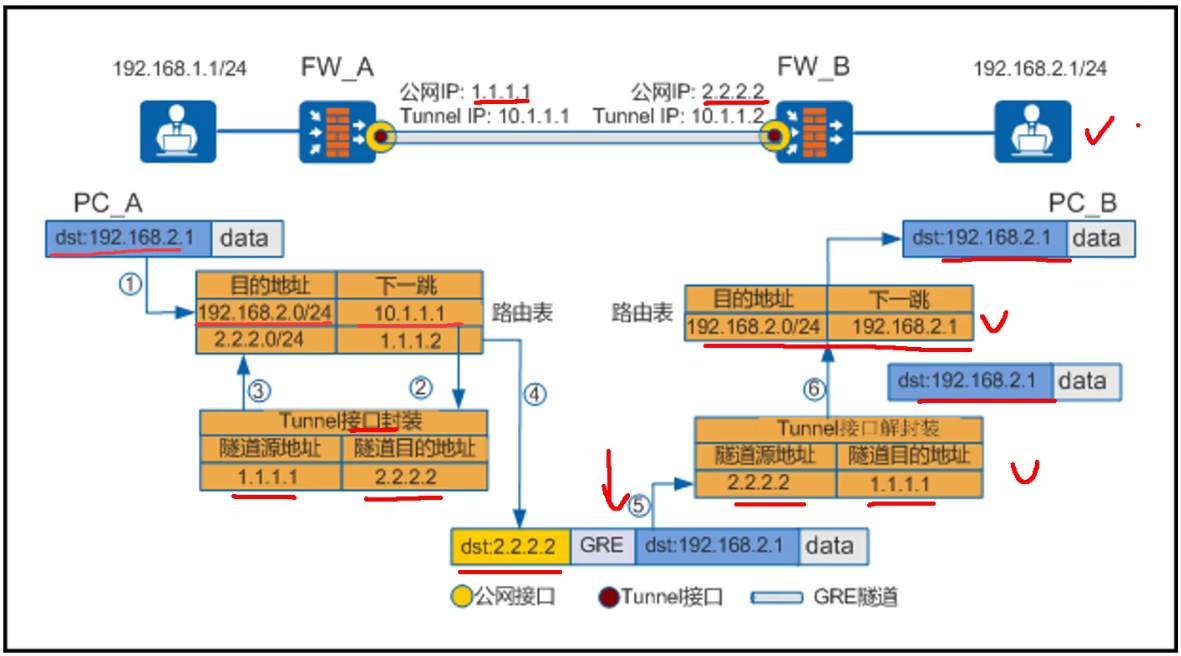
* 在原始封装的数据报文字段中加入新的IP头部，在新的IP报头和老的IP报头中间加一个协议，这就是GRE报头。
* 

* GRE头部
* 
* 把原本的IP报头当成货物，这个被称为乘客协议，GRE负责承载乘客协议的IP报文，可以理解为车辆，然后由新的IP报头在公网进行传输，公网设备只会查看新的IP报头，然后进行转发。

* GRE报头详解
* 协议号为47，经过GRE封装后，新的IP报头的协议号为47。GRE报头里面还有载荷协议，用于判断上层是上面类型。
* 
* GRE封装后报头
* 

GREVPN转发过程详解：

* 私网到私网的路由不走缺省路由，要走隧道口，当我们收到一个私网路由时，这时设备会查找路由表。
* 
* 查询路由表后，发现要去的地址的下一跳是隧道口，设备发现我们要走的是隧道口，所以会对数据报文进行GRE封装。
* 
* 设备加了一个GRE头部，同时会有新的IP头部，这个新的IP头部，会有新的源和目标IP地址，就是建立时的隧道源和隧道目标。
* 隧道口的意义在于设备查找路由时找到出接口是隧道口，或者下一跳是隧道目标。
* 因此，私网数据报文保留，加上GRE报头和新的IP报头（隧道源和隧道目标）。
* 
* 数据报文发送出去后，中途运营商的设备会根据外层的IP报头进行转发。
* 
* 目标设备收到数据报文后，进行解封装，看到GRE报文后，会继续解封装，然后在看到原先发送的数据报文，发现是私网包头，进行查找路由，发现有就进行转发，回包也是如此，这就完成通讯。
* 

* GRE通讯流程图
* 

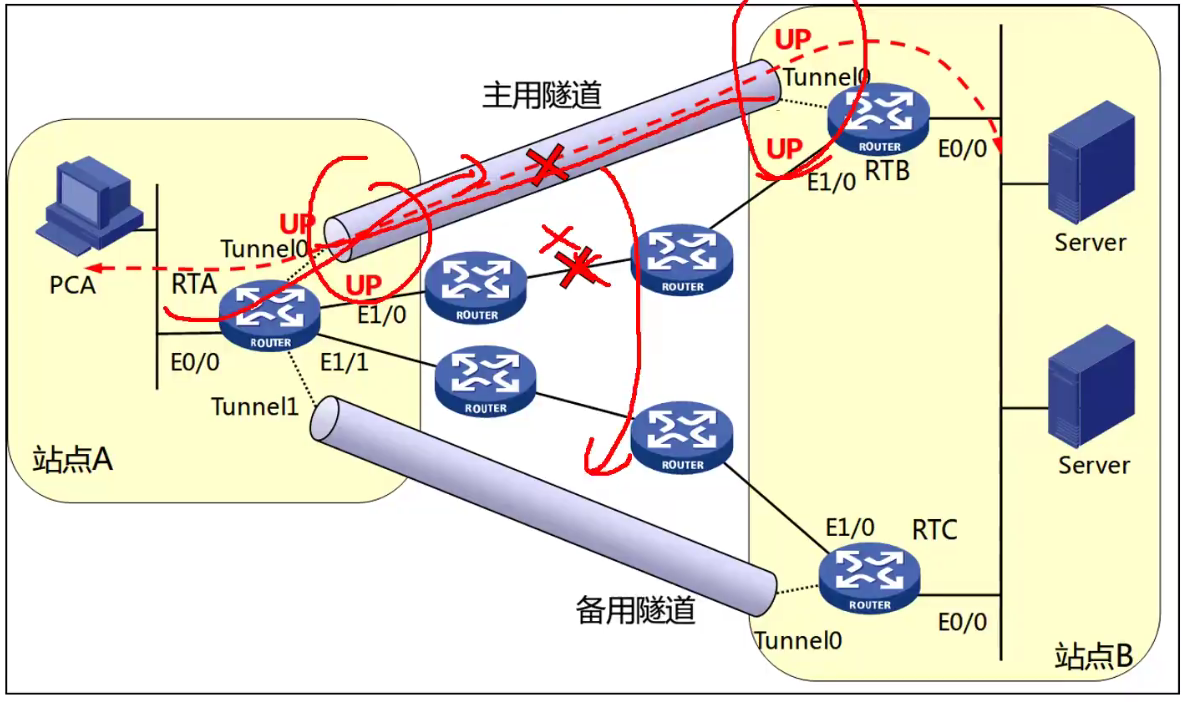
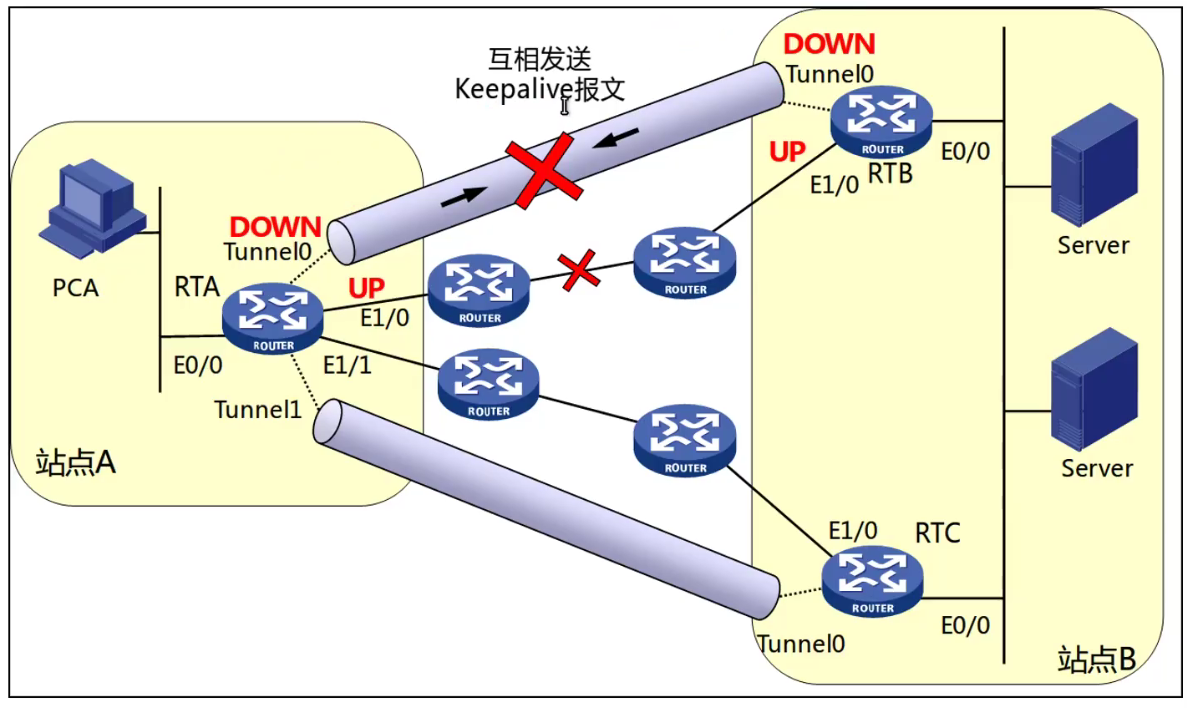
GREVPN配置：

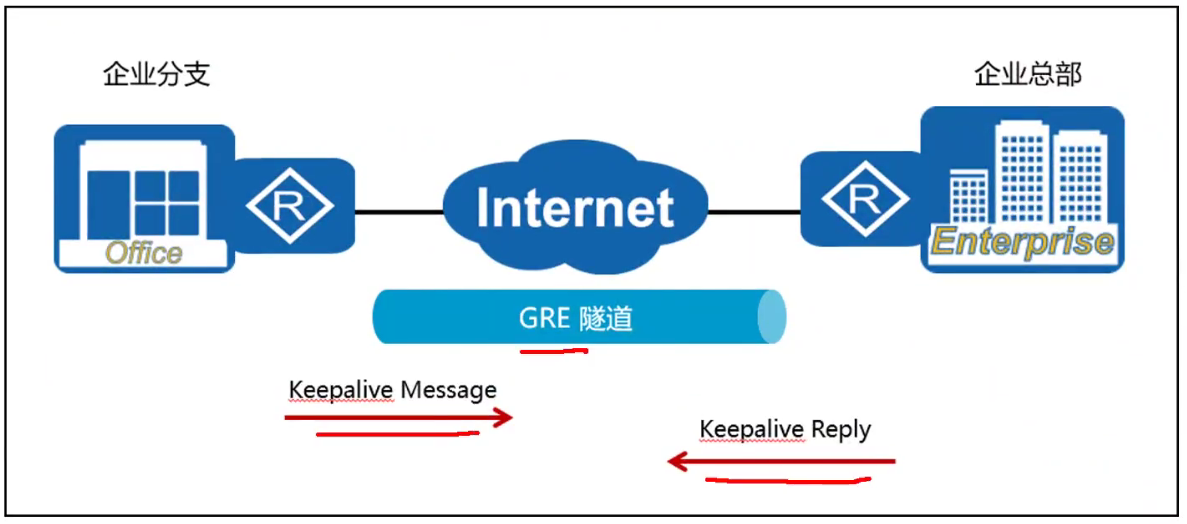
|  |  |
| --- | --- |
| 命令 | 备注 |
| Interface tunnel 0/x/x | 创建隧道口。 |
| Tunnel-protocol x | 指定协议。 |
| source | 指定隧道源/接口/IP。 |
| destination | 指定隧道目地/IP。 |
| Ip address x | 添加隧道口地址（必须） |
| Ip route-static d m tp | 配置静态隧道口。  d=目的 m=掩码  tp=隧道接口/宣告口 |
| Dis int tunnel x | 查看隧道口信息。 |
| Keepalive period 5 retry-times 3 | 开启Keepalive避免数据黑洞。 |
| Gre checksum/key | 配置简单校验/验证 |

GREVPN路由宣告注意事项：使用动态路由宣告时，千万不能宣告公网接口（只能宣告私网路由）。

* 如果a宣告了公网地址，那么b就会通过a的隧道口学习到这个公网地址，这个时候b的隧道目标就出现问题了，有两条路由，一是物理口的缺省，二是往隧道口的隧道目标，b向a发送数据报文时，会进行封装，如何通过隧道目标发送出去，然而隧道目标的下一跳还是隧道口，这样子就有问题了，会一直递归封装。
* 

GREVPN隧道口虚假状态：只要设备上有到隧道目标的路由，那么隧道口即可激活。

* 隧道口激活UP状态很简单，只要有去隧道目标的路由，即可启用隧道口，他不判断隧道目标链路是否正常，没有检测机制，如果中间链路出现故障，那么会造成业务中断。
* 
* 解决办法：开启keepalive，开启后，会通过隧道发送keepalive的数据包让对端回复，周期性发送，用于检测对端隧道是否可达。会发送keepalive message ，对端隧道会进行keepalive reply 回复。
* 

* 

* 