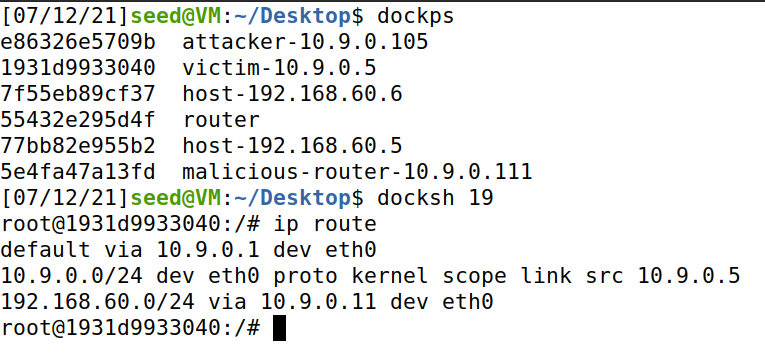
**ICMP Redirect Attack Lab**

**学号：57118223 姓名：齐天一**

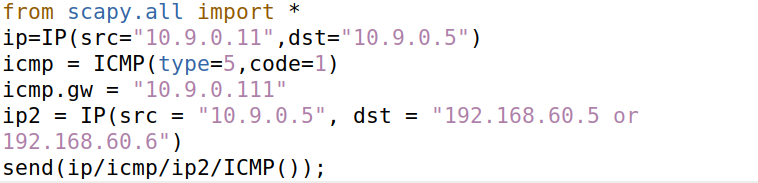
Task1:

**（0）进行重定向攻击**

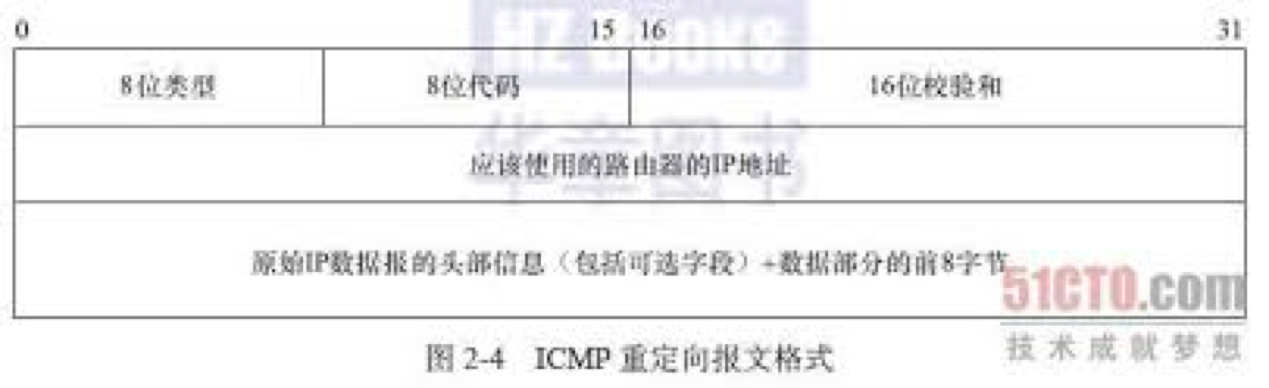
①登录受害者容器，查看路由，可以看到路由是正常，其中192.168.60.0子网的路由指向10.9.0.11



②编写如下代码用于重定向攻击

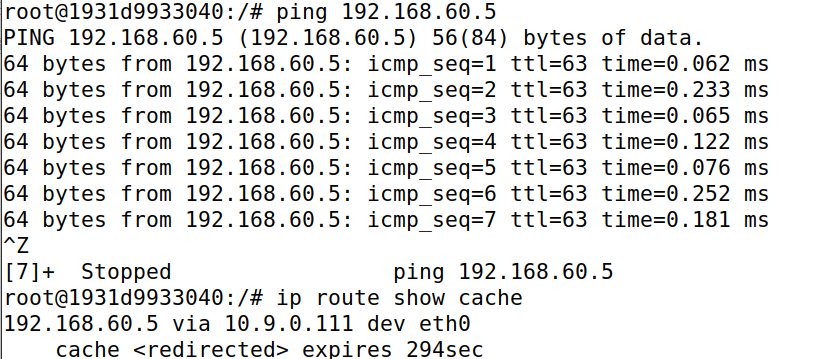


其中第一个ip对象对应于该报文的源ip和目标ip，前者为了伪装成路由而写作10.9.0.11，后者为受害者主机；icmp.gw表示重定向的网关，将其设为恶意路由器；第二个ip对象ip2对应原来报文的源ip和目标ip，前者为受害者主机，后者为192.168.60.5/192.168.60.6。

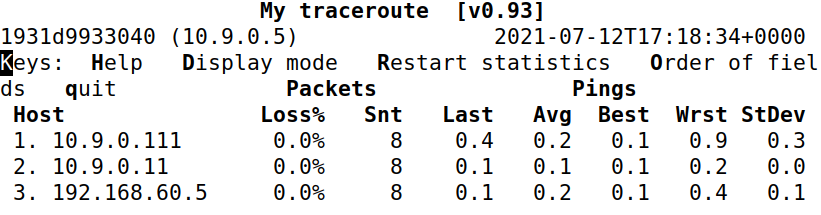


（网上找到的ip重定向报文格式）

③受害者主机ping 192.168.60.5,同时攻击者运行攻击程序，发送重定向报文



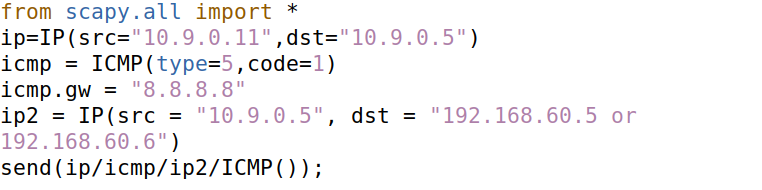
随后停止ping，并查看路由缓存，可以看到，前往192.168.60.5的路由被指向为10.9.0.111（即恶意路由）。



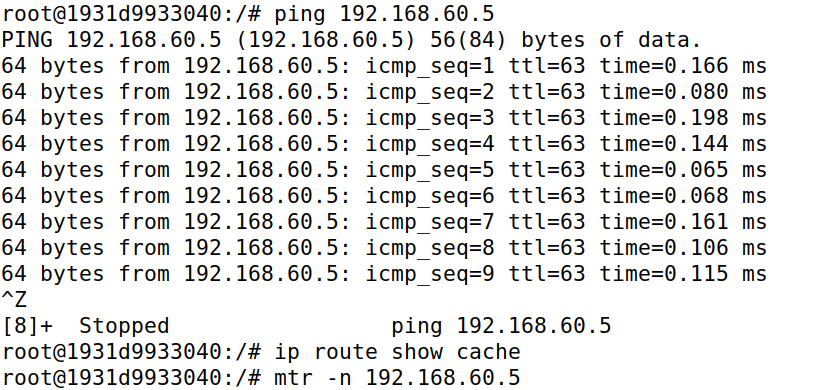
输入mtr命令进行查看，可以看到路径被重新路由了，说明成功实施了攻击。

1. **问题1：能否重定向到远程计算机**

①修改重定向的网关ip为8.8.8.8

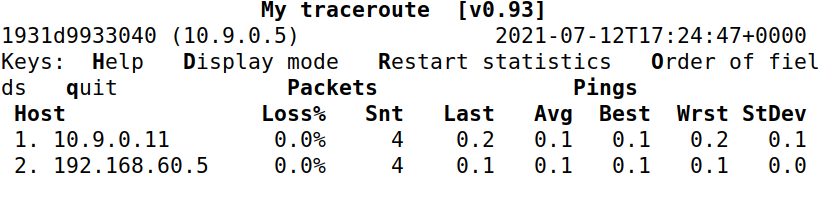


②重新执行攻击过程（在此之前清空受害者ip路由缓存）



可以看到查看缓存，并没有变化。

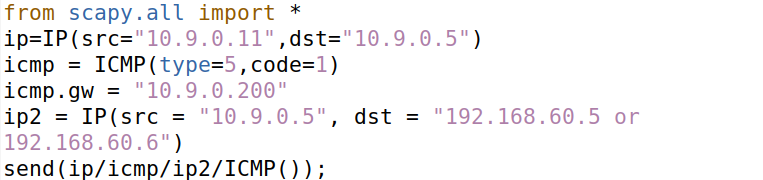
再次输入mtr指令：



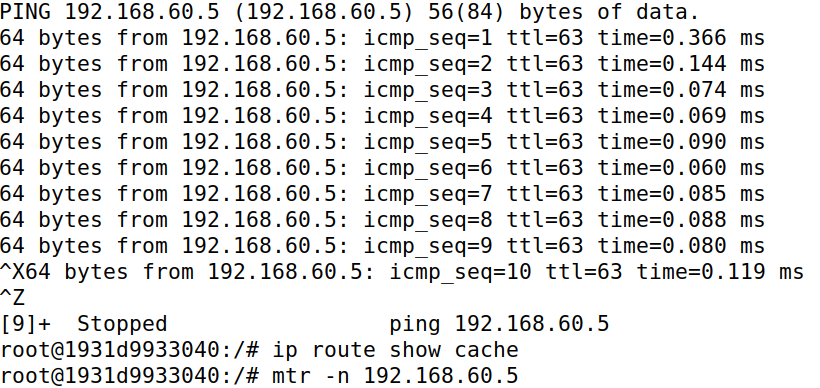
路径并未被导向8.8.8.8，说明**不能**重定向到远程计算机。

1. **问题2：能否重定向到本网络不存在主机**

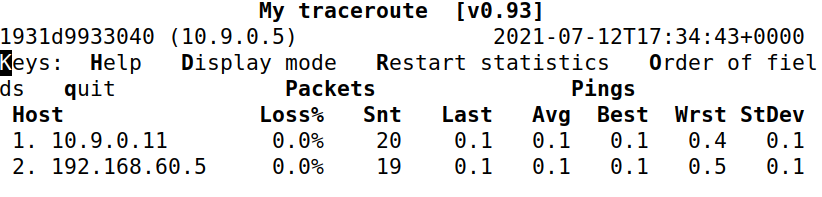
①构造不存的主机10.9.0.200，修改gw值



②重新执行攻击，并查看路由缓存，没有出现改动



输入指令mtr查看：

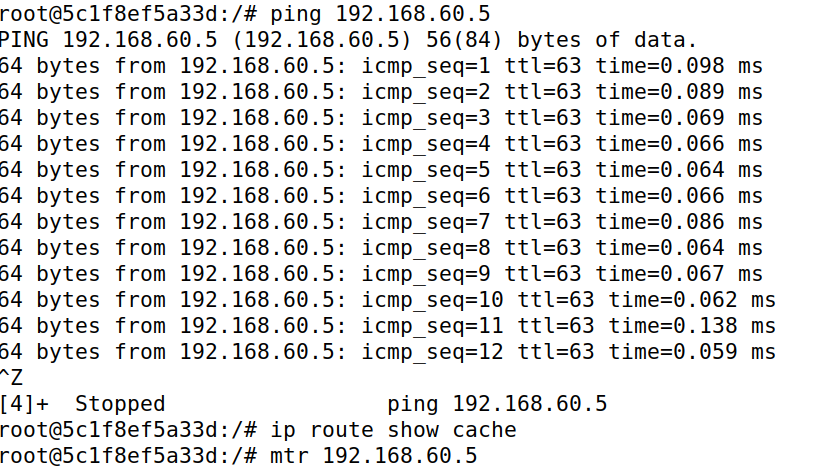


并没有讲路由导向 10.9.0.200，说明**不能**导向不存在主机。

1. **问题3：修改三个参数后重新进行攻击**

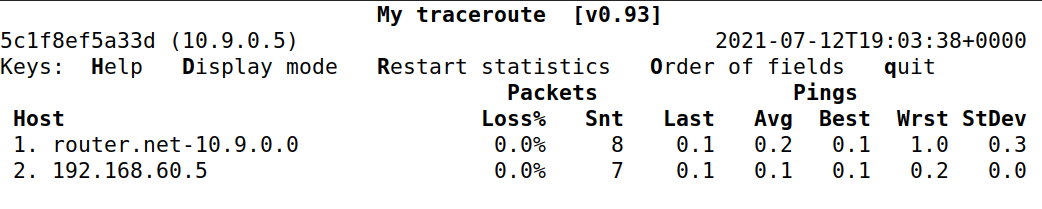
①在配置文件中修改参数：  


②在受害者机进行ping操作时，运行攻击程序



可以看到，路由缓存中并未出现修改后的内容

使用mtr命令查看：



路径中没有出现恶意路由，说明重定向失败了。

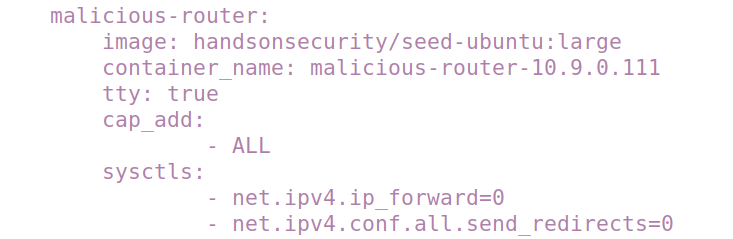
原因：

通过查阅资料，我们可以得知这三个标记位是关于重定向报文转发的，值为0的时候表示禁止重定向，值为1的时候表示允许重定向。如果将三个都置为1，表示转发功能打开。此时，**当恶意路由器受到受害者ip报文时，会发现该报文必须经过路由器10.9.0.11时，会向受害者发送一个重定向报文，将其路由重定向为10.9.0.11，这样原来伪造的重定向到10.9.0.111报文就被覆盖了，从而导致无法完成攻击。**这说明如果想要完成重定向攻击，就需要关闭正常的重定向功能。

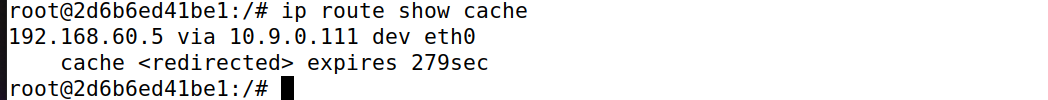
Task2：

1. 尝试MITM攻击

①修改标记位，关闭恶意路由的路由转发功能。

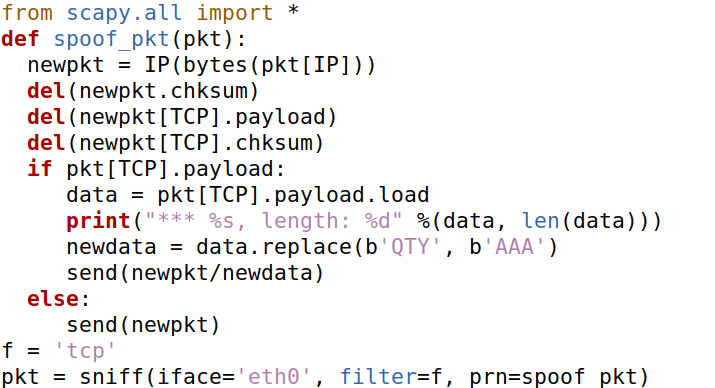


②发动icmp重定向攻击，将路由重定向到恶意路由器

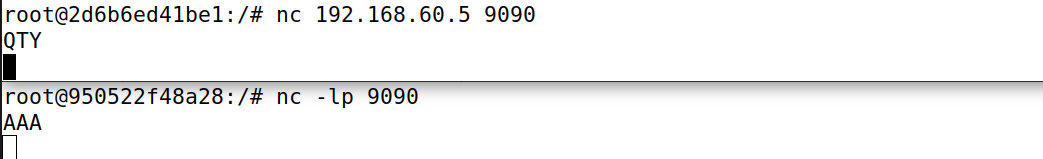


③在目标主机，打开9090端口监听；在受害者主机，使用nc连接到目标主机的端口。在连接后，攻击者运行MITM.py程序，嗅探并重发报文。

程序如下：



输入字符串“QTY”：



可以看到接受端成功收到了字符串并将其转换为了AAA

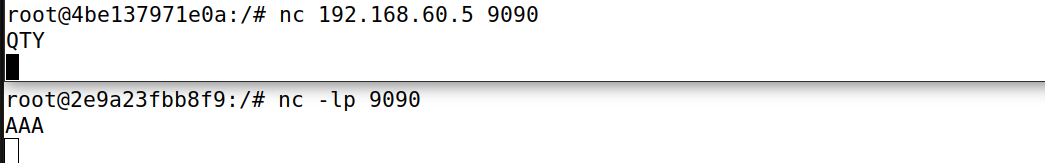
1. **问题1：应该选择哪个方向的报文**

使用mac地址

①受害者->目标主机

过滤策略如下（源mac地址为受害者）：

MBU7F`B8WC~J6B$3GP~JNL4

运行程序：  


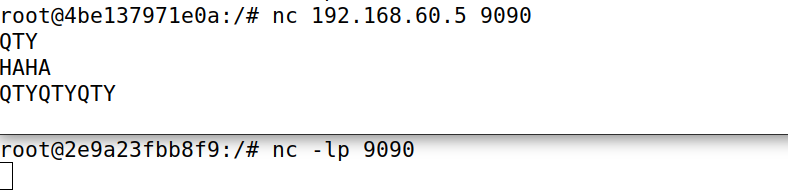
可以看到，接收端成功输出了结果，并完成了字符串的替换。

②目标主机->受害者

过滤策略如下（源mac地址为目标主机）：

BDJ[(MJ6GL42OB}3_(BF[R4

运行程序并开始攻击时的现象：

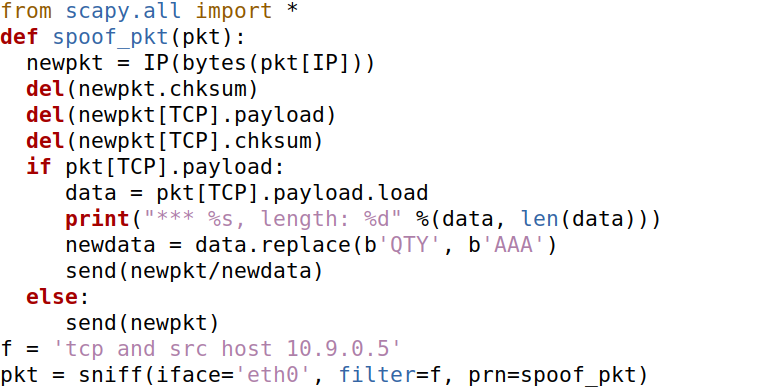


可以看到，接收端没有反映，说明连接失败。

分析：可以看到，只捕获“受害者->目标主机”方向的报文就可以完成通信，反之则不行，可以认为只需要捕获“受害者->目标主机”方向的报文即可。

1. **问题2：使用ip过滤和使用mac地址过滤的不同**  
   IP地址过滤

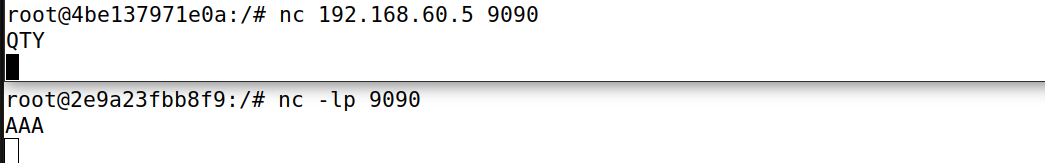
①修改过滤器内容。将过滤器修改为“tcp and src host 10.9.0.5”其中10.9.0.5是受害者主机发送的，表示筛选出来自受害者的tcp报文：



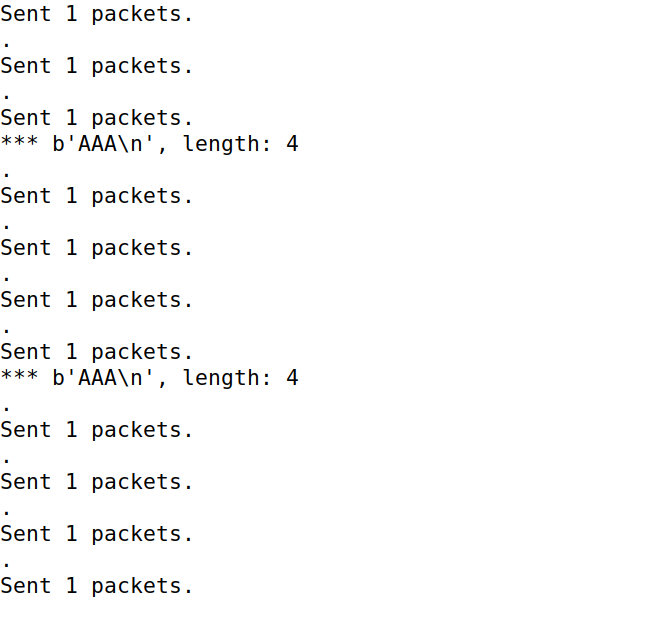
②受害者和目标主机建立tcp连接，攻击者运行程序来伪造受害者报文

（过程与第0部分一样）

③输入指令QTY，可以看到在另一端输出了AAA，可以看出伪造成功：



查看攻击端，会发现它一直在发送报文：



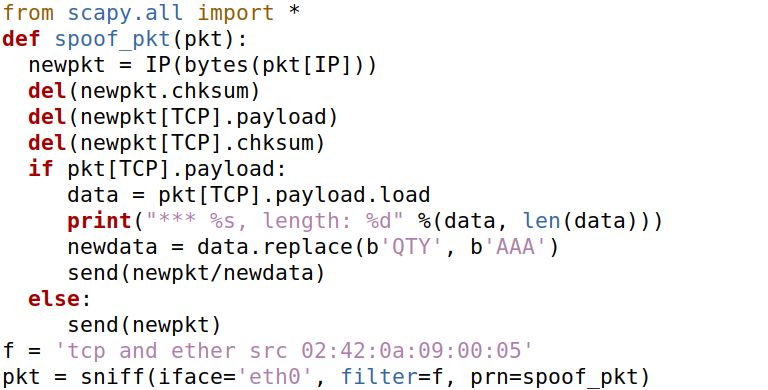
分析：

一直在发送报文，说明嗅探程序也一直在捕获报文，而受害者只发送了几个报文，显然不可能时受害者造成的这一现象。唯一可能的原因是：嗅探程序它会对攻击者重发的报文也进行了捕获和重发，这就导致了程序陷入了“嗅探到一个报文->对其进行重发->捕获到重发的报文->对重发的报文进行重发”这一循环，造成了资源的浪费和效率的低下。

为什么会捕获自己发送的报文？这是因为路由器不会修改报文的ip地址，所以嗅探程序重发的报文其ip地址和原来一样，同样符合嗅探程序的过滤策略，导致其会被再次捕获。因此需要有一种标识来区分报文是否由嗅探程序所在的攻击者发送，而mac地址非常符合这一点。

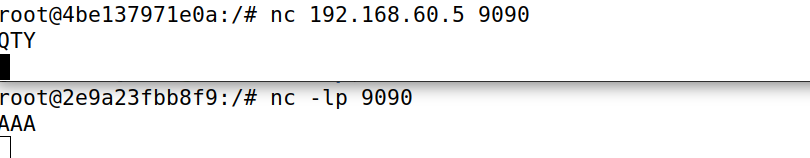
MAC地址过滤

①修改过滤器内容。将过滤器修改为“tcp and ether src 02:42:0a:09:00:05”其中02:42:0a:09:00:05是受害者的mac地址的，表示筛选来自受害者的tcp报文：

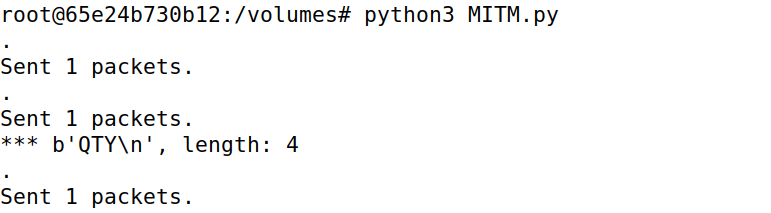
  
②受害者和目标主机建立tcp连接，攻击者运行上面的程序来伪造受害者报文

（过程与前面类似，不再赘述）

③输入指令QTY，可以看到在另一端输出了AAA，可以看出伪造成功：



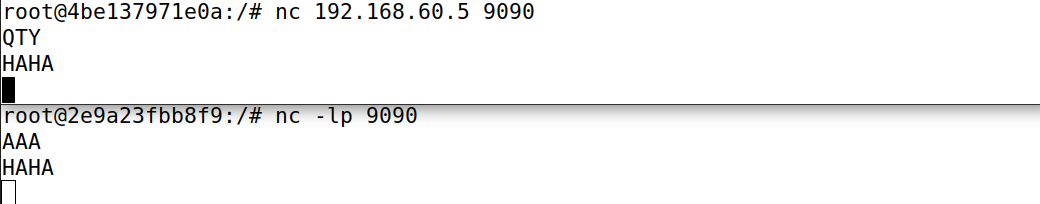
此时攻击者端显示捕获了三个报文：



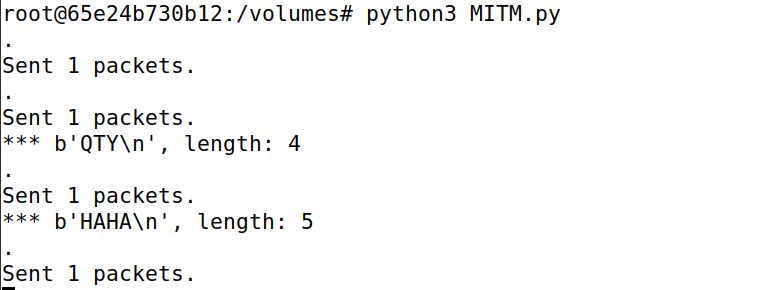
其中前两个是，攻击者并未处理，只是将其转发出去；

最后一个报文是数据报文，其中包含了发送的字符串“QTY”，攻击者将其改为“AAA”后重新发送出去。

④输入一个无关的字符，可以看到在接收者那里也输出了这一字符串：

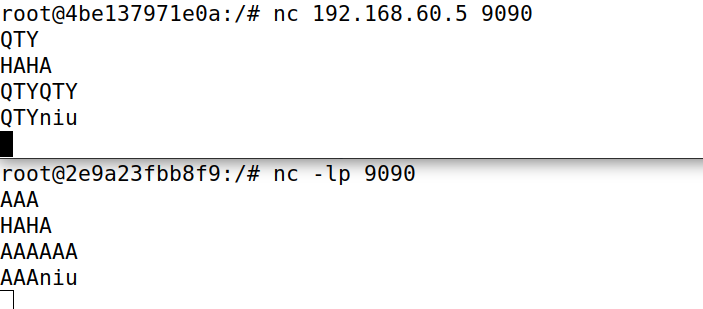


攻击者端捕获了一个报文：

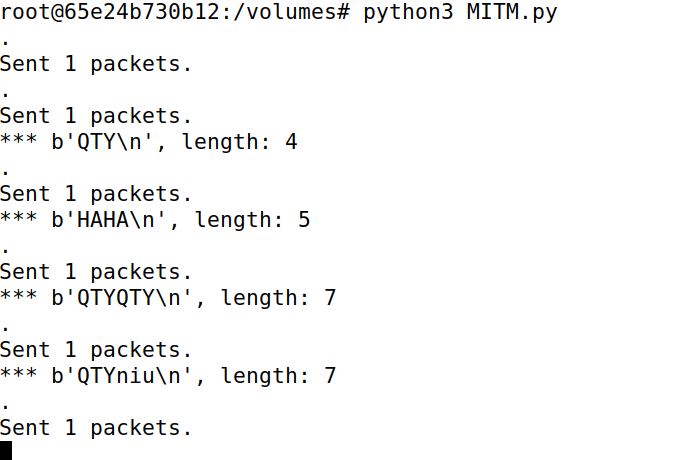


该报文即是包含“HAHA”的数据报文，因为不包含“QTY”，所以不予以修改。

⑤输入一些带有“QTY”的字符串，可以看到再接收端将字符串内部的“QTY”都修改为了“AAA”



攻击者端可以看到，对于每个字符串，都捕获了一个数据报文，然后攻击者将他们修改后分别发出去：



分析：

可以看到，对于攻击者而言，使用Mac地址过滤只需要捕获和重发很少的报文，除了最开始发送了两个用于应答的报文，后面每个字符串都只需重发一个报文，其效果远远好于使用ip进行过滤。

造成这种现象的原因是再报文转发的过程中，路由不会对ip地址进行修改，也就是说在其传输路径上，源ip和目标ip是不变的；但mac地址则不同，它在报文的每次转发时都会改变，取决于当前路由端口的mac地址，因此Max地址能表示报文**最近**的发送者（或转发者）。所以，**我们只需要筛选来自受害者mac地址的报文，就能有效防止嗅探程序捕获自身发送的报文，避免了“捕获自己发送的报文->发送该报文->再次捕获该报文”的循环，从而提高了效率。**