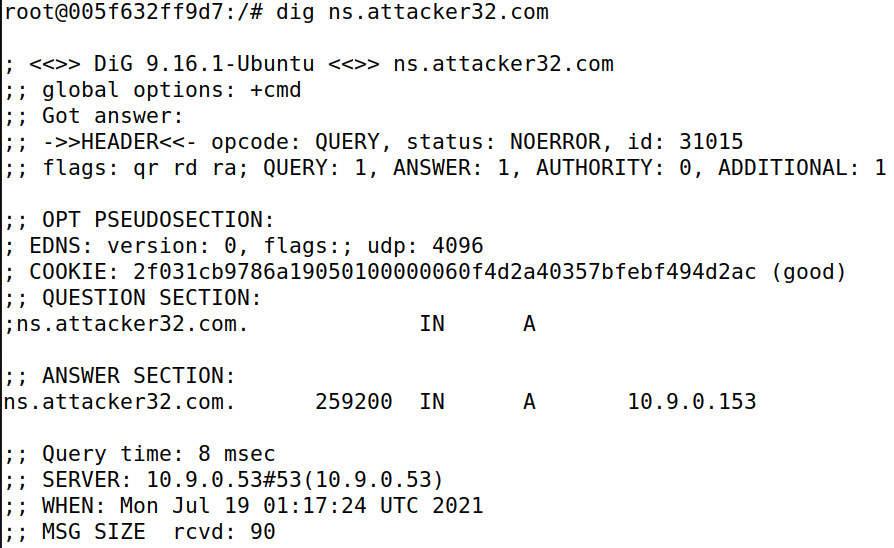
**Local DNS Attack Lab**

**57118215程文泽**

**Testing the DNS Setup**

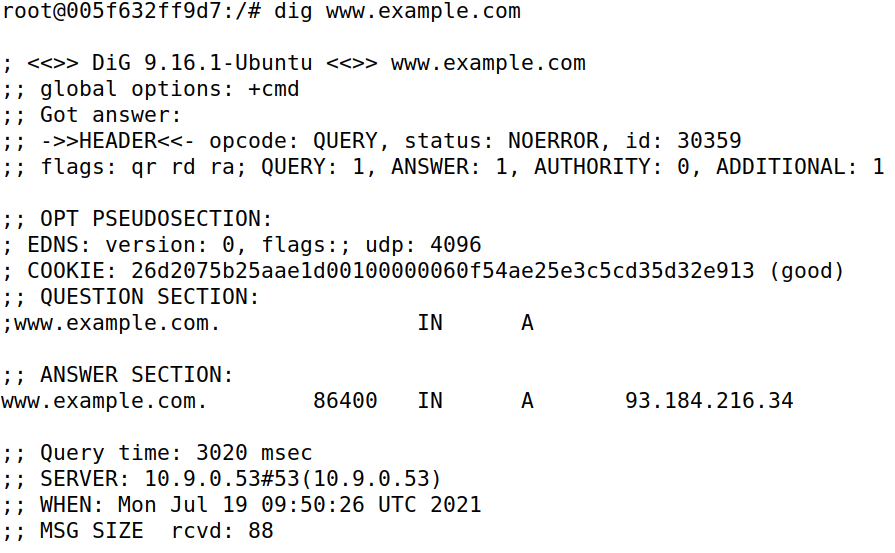
Get the IP address of ns.attacker32.com：

查询ns.attacker32.com的IP地址，因为本地服务器存在转发attacker32.com区域，所以DNS请求会发给对应IP为10.9.0.153的域名服务器，最后成功得到回复。

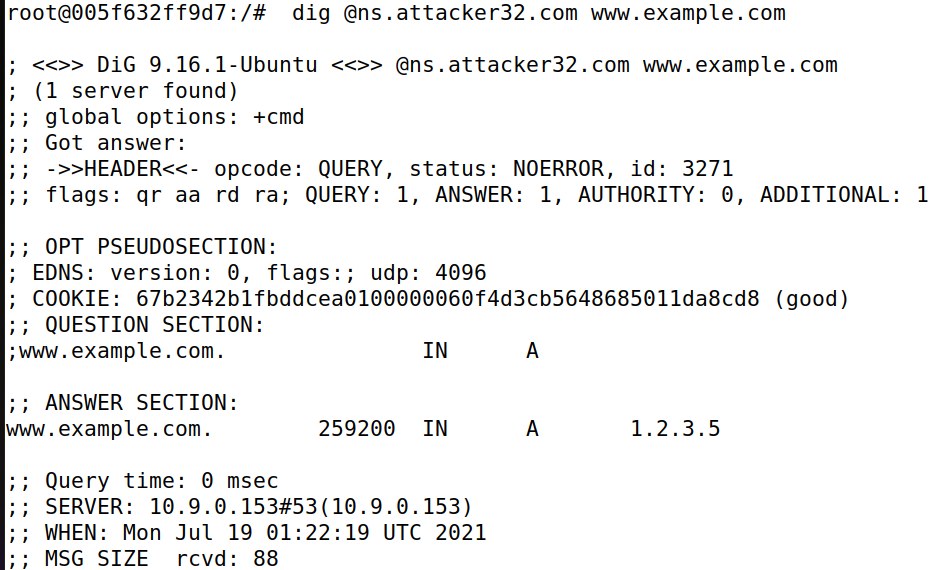


Get the IP address of www.example.com：

查询www.example.com的IP地址，因为本地服务器没有该域名的记录，它会向网络中真正的根域名服务器查询，最后得到答复，

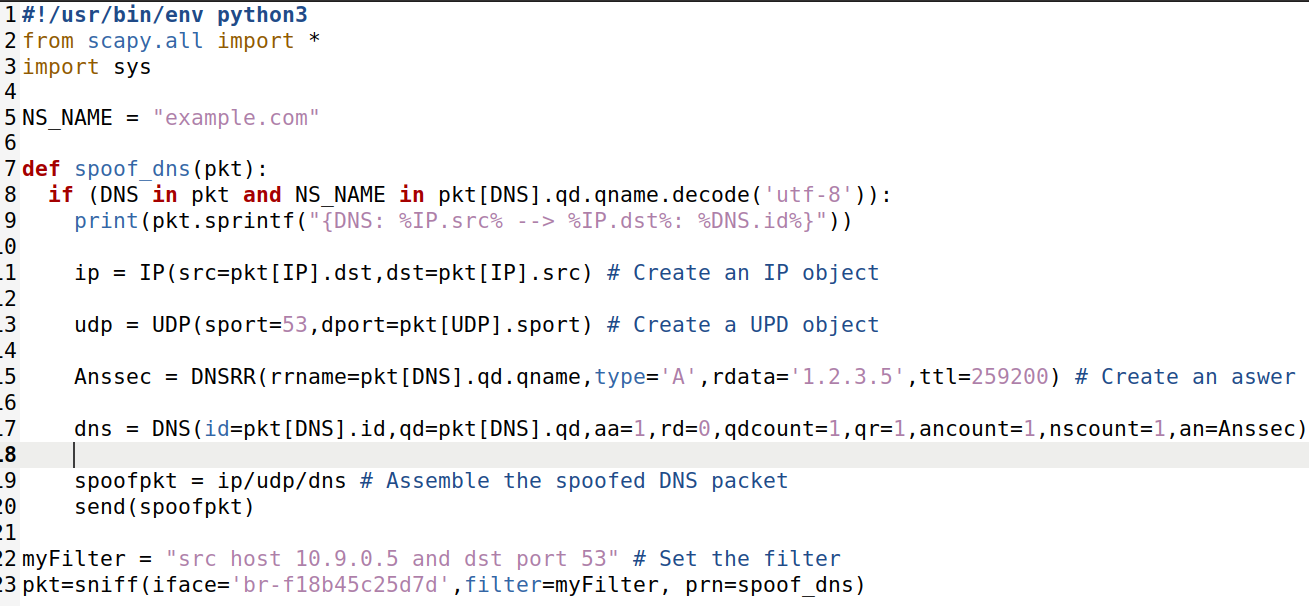


将查询www.example.com的DNS请求，发送给attacker.com的域名服务器，成功得到回答。

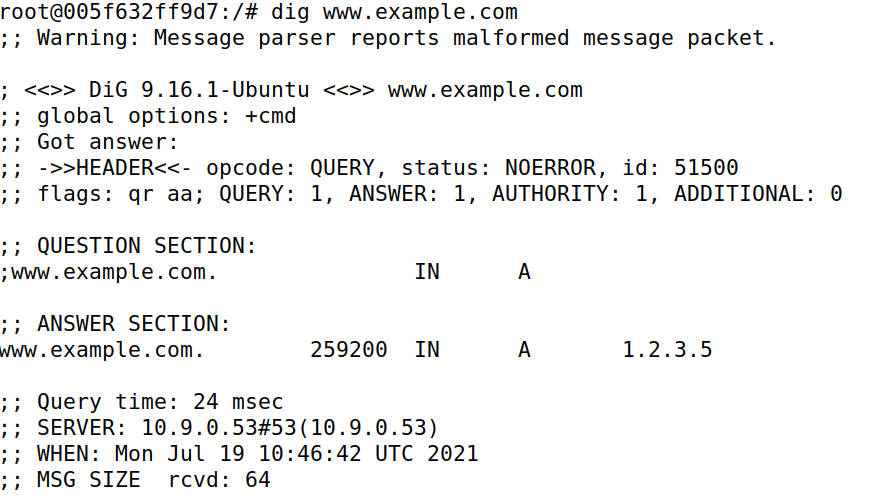


**Task 1: Directly Spoofing Response to User**

伪造DNS回复脚本如下：

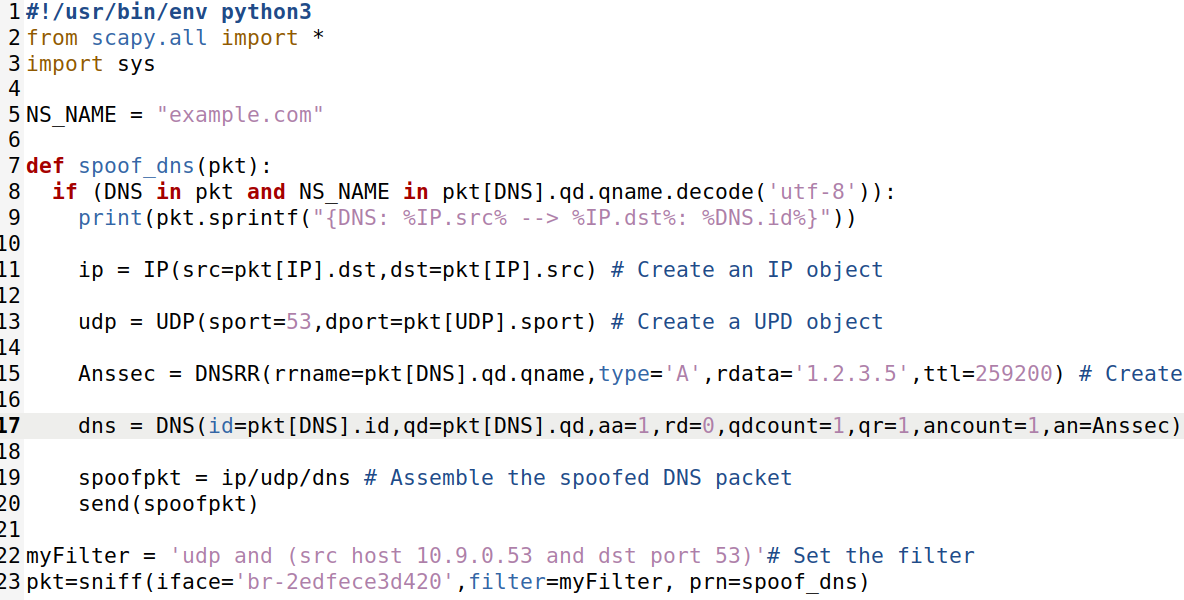


在运行攻击脚本后，在用户主机里运行dig命令，结果如下，可以看到显示域名的IP地址是1.2.3.5，与脚本里设置的一样，且与未攻击之前得到的IP地址完全不同。

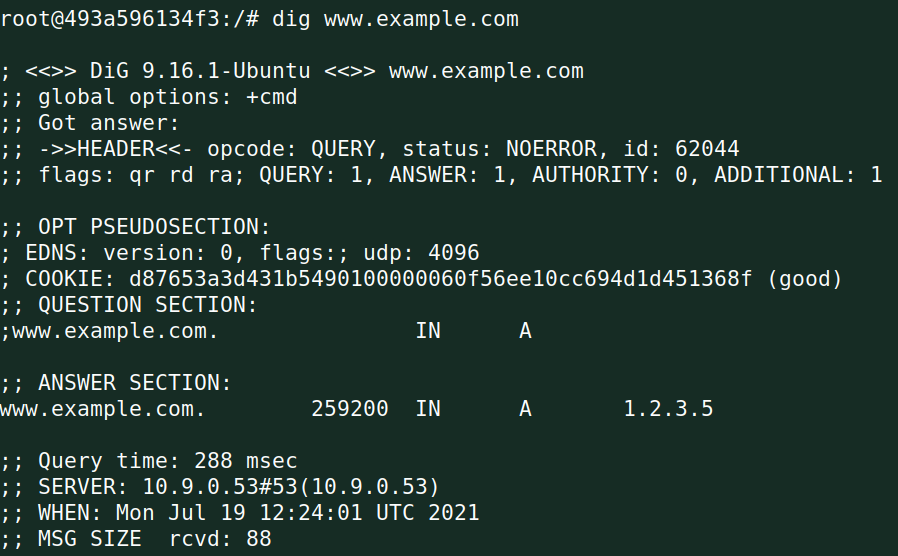


**Task 2: DNS Cache Poisoning Attack – Spoofing Answers**

伪造DNS回复脚本如下：



在运行攻击脚本后，在用户主机里运行dig命令，结果如下，可以看到显示域名的IP地址是1.2.3.5，与脚本里设置的一样，随后查看本地域名服务器的缓存，可以看到伪造回复被本地域名服务器缓存了，而后停止攻击脚本，结果仍然不变。



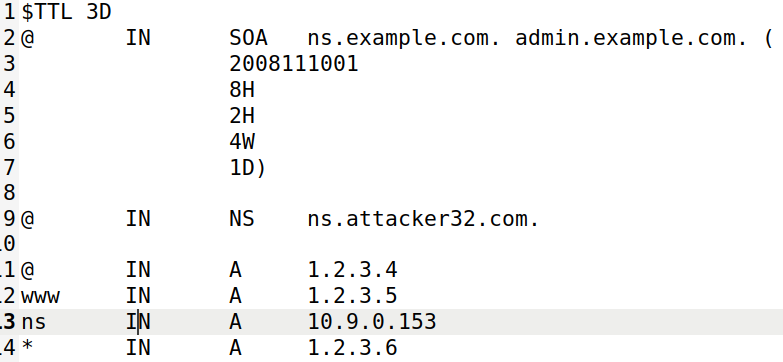


**Task 3: Spoofing NS Records**

伪造DNS回复脚本如下，在task2脚本中增加了NS记录。

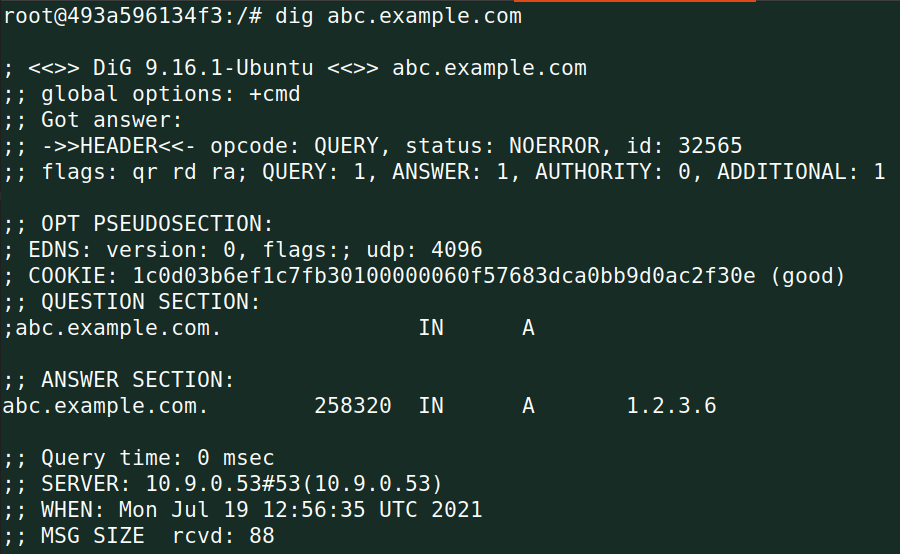


查看攻击者在恶意DNS服务器对example.com域中所有域名对应的伪造IP，如下



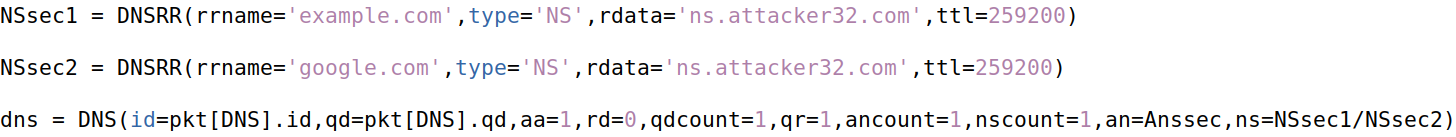
随后运行脚本，在用户主机里运行dig命令，查看本地DNS服务器缓存，可以看到，伪造NS记录被缓存了，最后在用户主机上对example.com域中的任意主机名运行dig命令，得到如下结果，可以看到，得到IP地址与上面的伪造IP地址相同，攻击成功。



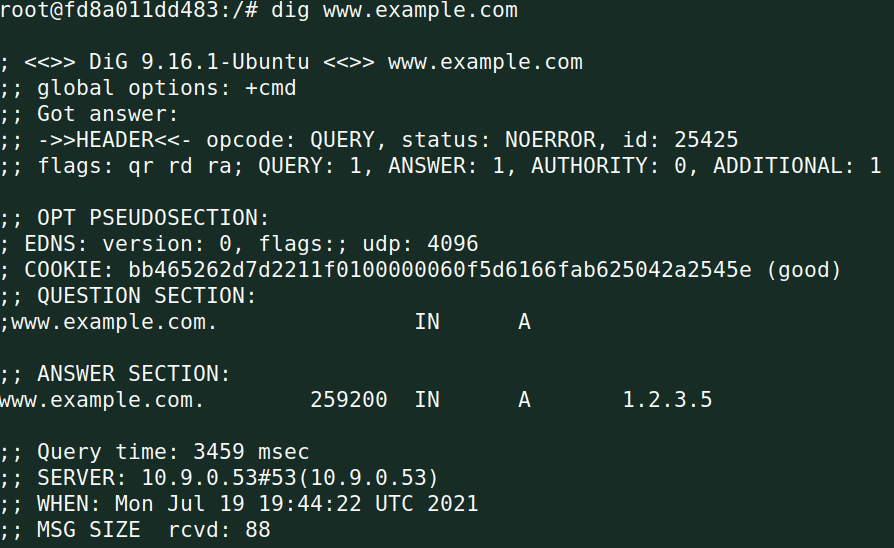


**Task 4: Spoofing NS Records for Another Domain**

伪造DNS回复脚本如下，在task3脚本中再增加了一个NS记录



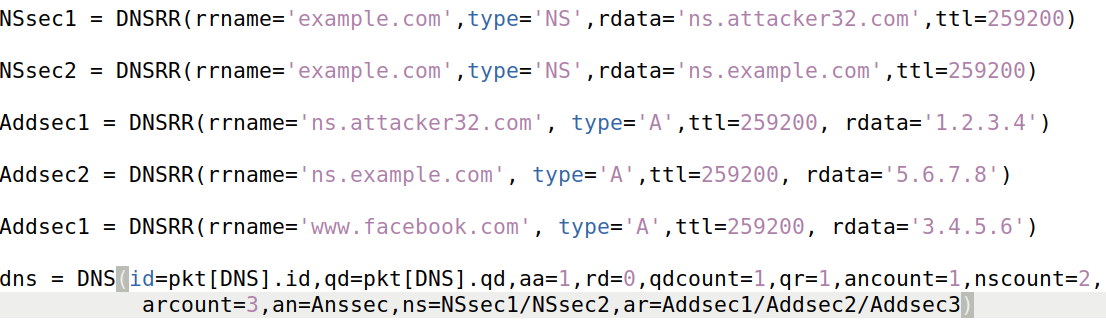
随后运行脚本，结果如下，攻击成功，然后查看本地DNS服务器缓存如下，可以看到，只有ns.attacker32.com记录被缓存了，这是因为第一条记录是合法的所以会被缓存，而第二条goole.com记录是伪造的并不合法，所以本地DNS服务器没有接收第二条记录。





**Task 5: Spoofing Records in the Additional Section**

修改脚本如下：



随后运行脚本，结果如下，攻击成功，然后查看本地DNS服务器缓存如下，可以看到，两个NS记录成功被缓存下来，附加记录中只有ns.example.com被记录下来了，对于附加记录www.face.book没有出现在NS记录中，因此会被忽略掉。

对于附加记录ns.attack32.com，我觉得是因为本地DNS服务器知道ns.attack32.com真正的IP地址，所以对不合法的附加记录ns.attack32.com不会写进缓存，

