**scapy**

# 介绍

本文主要讲解scapy的简单使用,包括怎么发包,收到包后怎么分析和取得我们所需要的字段。

**本文档需要阅读,需要有以下基础知识需要掌握**：

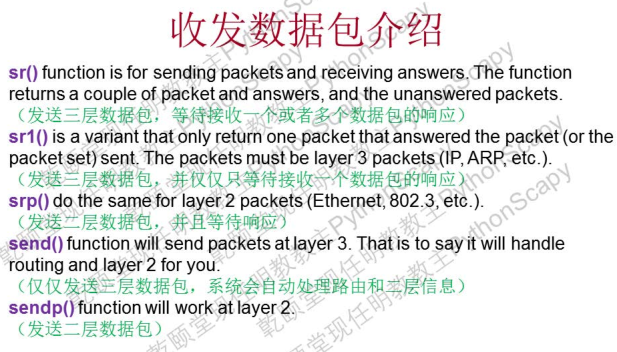
1. TCP/ip协议层次基础,如基础5层模型有哪些,ip是哪一层,arp是哪一层,icmp是哪一层。
2. 构建一个包需要哪些层次的东西。
3. Arp毒化arp原理。
4. Tcp 3次连接,4次断开。

# scapy 交互界面基础使用

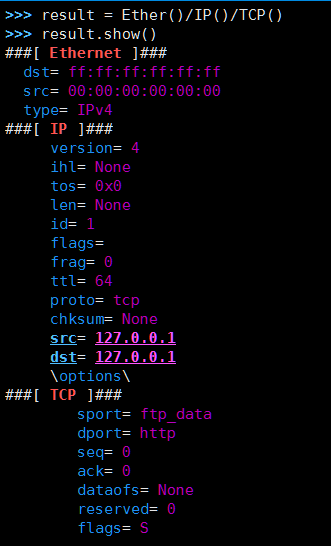
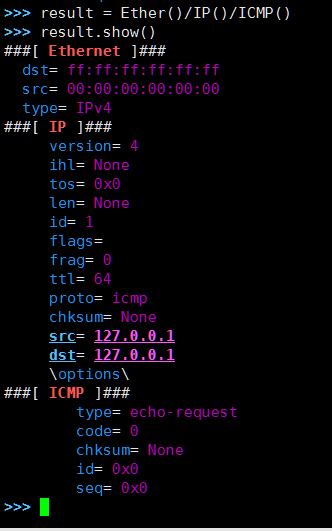
scapy可以作为一个单独的程序单独发包和收包,但是python可以给与它灵魂。

下面是scapy人工交互界面的操作。

首先贴出一张课程的图片怎么发2层包或者3层包。



1. **构建一个包**

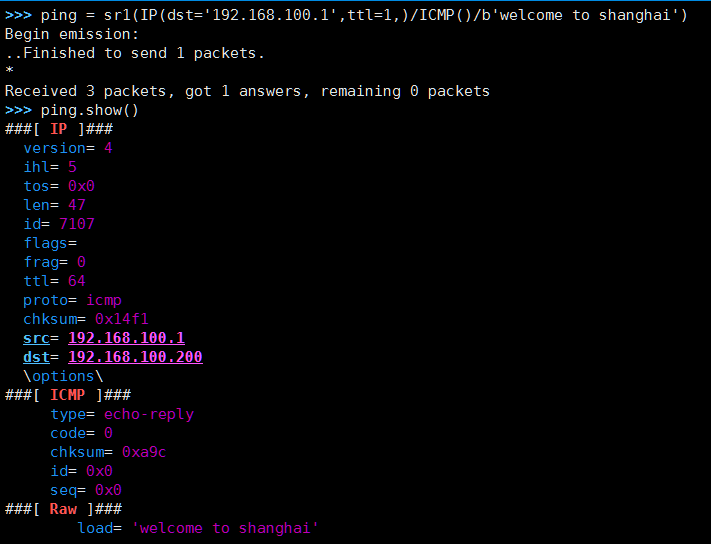


以上构建了2个包,一个为ICMP,一个为TCP,

为什么要这么写:

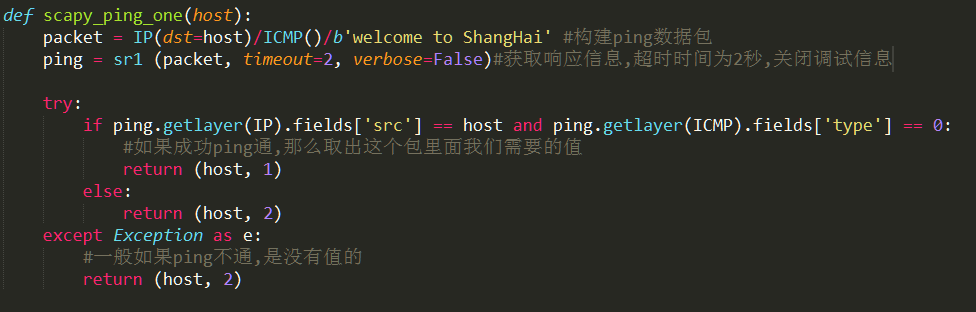
Ether() 一般在构建2层包的时候需要,而IP则是网络层的,TCP或者ICMP则是传输层的,一般这些()里面是有参数的,但是一般有一些默认值,比如dst,src等,我们也可以手动指定来达到伪造的目的，具体的在每一个分支会详细讲解。

1. **产生一个ping包。**



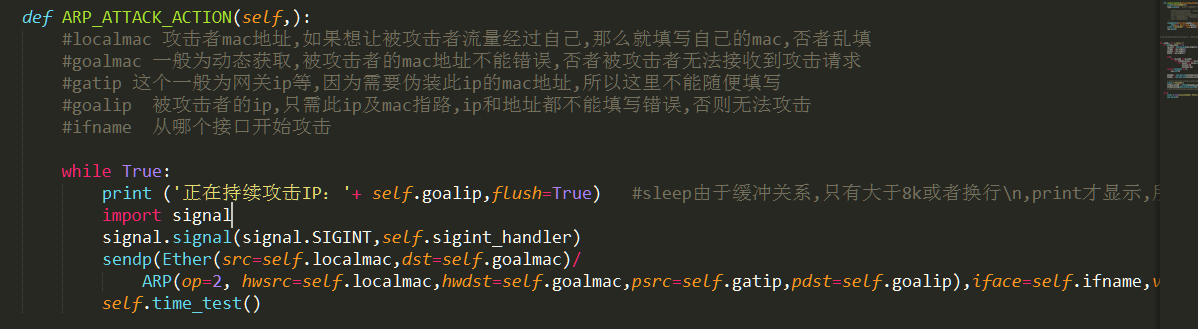
**如上所述,ICMP不是二层的,所以不需要加Ether(),一般ping通了才会有show()这个方法**

# ping



Ping 程序的代码主要核心就是上述,如getlayer拿到哪个层的包，以及ICMP如果主机不在线程序就回出错等问题,其余的完全是围绕该程序编写。

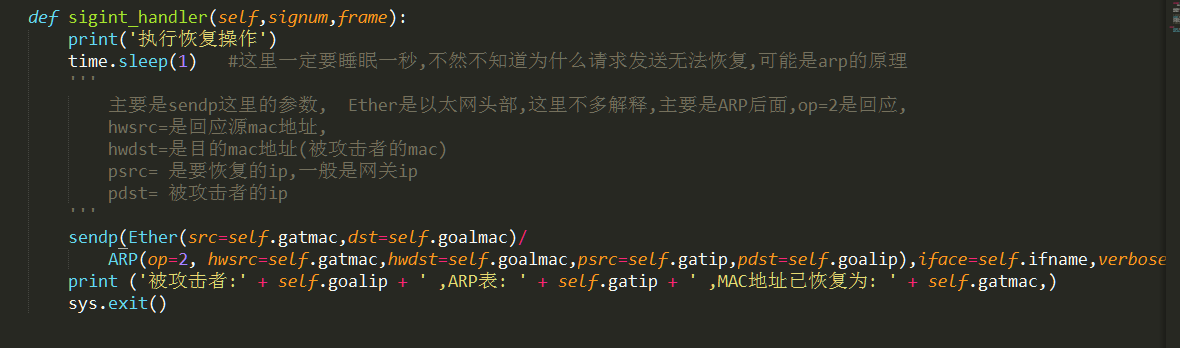
# ARP毒化攻击



Arp毒化攻击原理简单,不需要太多介绍，主要是上述代码中：

Arp不能跨三层发送,所以这里sendp发送2层包,回应op为2,发送为1,格式为Ether()/ARP()即可。

以及如下代码中,恢复的时候，源和目标需要是正确的ip和mac来回应的。



# TCP扫描与DoS攻击

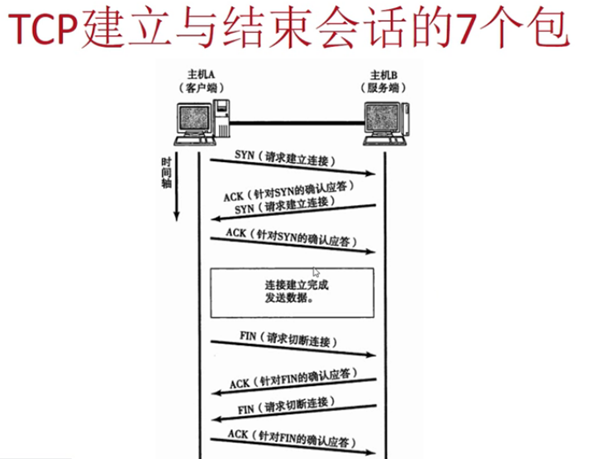
## TCP原理介绍

因为本节需要做TCP的全开攻击和半开攻击,所以本节原理一定要弄清楚。

1. 全开攻击,建立大量三次握手正常连接,来阻塞服务器。
2. 半开攻击,建立 大量ACK 请求连接,而不管回应的包,意思就是只做第一步。

Tcp是可靠连接,如果要传输数据,必须3次握手,而udp只要请求就可以连接。

TCP 连接与断开总共7个包,这里简单解释一波。

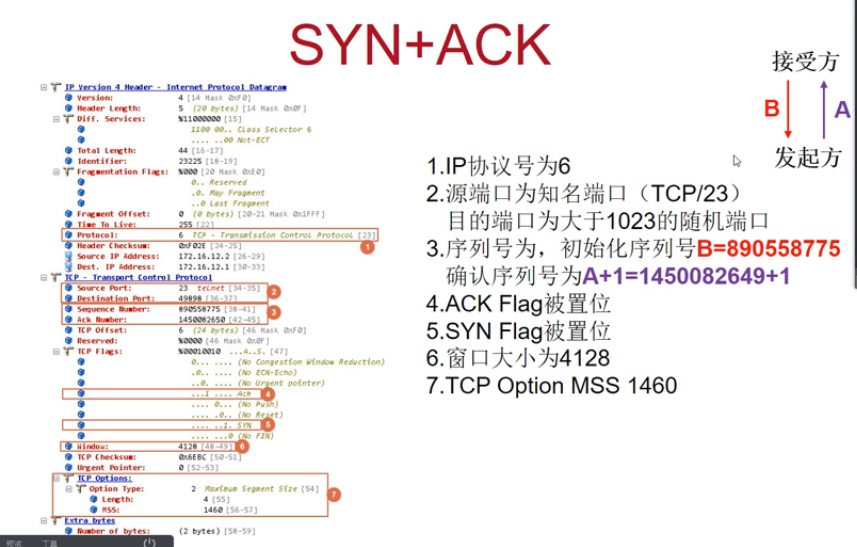


1. 首先,断开的时候也有可能是3个包,有时候ACK+FIN在一个包里面发送了。

下面简单看看SYN的一些包里面有什么



然后主机B的回应包



最后一个ACK,确认序列号在+1。



## 开始前准备

**在TCP与DoS正式开始前,还需要知道二点:**

Syn,fin,rst,ack都可以用数字来表示。

1 fin 2syn 4 reset 8 push 16ack 比如回ack+fin 那么就是1+16 = 17



Linux主机开启防火墙:

在tcp扫描及Dos攻击时, 我的攻击客户端一般为linux,这时,使用scapy来像被攻击端发送了ACK报文之后,被攻击端就回给供给端SYN+ACK,这里linux的系统首先会比scapy这个程序先拿到请求,看到是SYN+ACK的包,认为我没有请求连接,便会把这个包给RST掉,那么scapy便拿不到这个包了,所以,在linux主机上要禁止发送RST的包。

Linux 防火墙处理：

Centos7

systemctl start firewalld

firewall-cmd --direct --add-rule ipv4 filter OUTPUT 1 -p tcp --tcp-flags RST RST -s 192.168.100.200 -j DROP

firewall-cmd --direct --add-rule ipv4 filter OUTPUT 1 -p icmp -s 192.168.100.200 -j DROP

## tcp端口扫描及tcp DoS攻击

本节tcp端口及tcpdos攻击原理及注意事项在5.1-5.3已讲解清楚,本小节主要记录代码的注意事项。

本小节主要有以下代码：

1. **random\_library**

创建随机的ip,port,sn的随机类文件。

1. **thread\_result**

多线程可以取得线程执行结果的自定义类,本人想测试拿到thread的返回值,所以使用了此函数。

1. **tcp\_handshake**

tcp 扫描及攻击的核心,后面的tcp\_scan以及tcp\_dos都是调用里面的函数来实现扫描和攻击效果

1. **tcp\_scan**

tcp端口的扫描脚本,主要是调用tcp\_handshake来实现功能。

1. **tcp\_dos**

tcpdos攻击的脚本，包含：

伪造IP地址攻击tcp半开攻击。

不伪造IP地址tcp全开建立正常连接攻击。

简单来说 tcp\_handshake 调用 random\_library

tcp\_scan及tcp\_dos调用 tcp\_handshake.

核心代码handshake逻辑如下：

首先要构建一个包

if 伪装源IP

生成的包使用随机IP来发送

else:

使用本机IP来发送

If 发送包不需要拿到结果：

把包发送出去即可,那么就是syn半开攻击

else:

try:

就发送并且接收一个包

If 这个包正常给了回应

If 我们不需要建立tcp全开连接：

直接返回

else:

将计算好的ack包返回给服务器

else:

直接返回结果

Except：

如果遇到错误,直接返回连接错误

## 流量分析

1. **分析pcap包中的字段。**

本节代码: flow\_analyze

1. Python 读取PCAP文件。
2. Python 自动找到TCP源端口为21,TCP负载有 “pass”关键字的数据包
3. Python 返回匹配的数据包。

本章理论：

每个读取出来的包,都有如下层次。

　　　　物理层 --> 网络层 --> 传输层 --> 应用层

　　每一层的数据都可以根据相应层的协议名获取，然后再通过字段名获取具体层具体字段的信息，代码如下：

print p['Ether'].name

print p['IP'].name

print p['TCP'].name

也可以用一下方式来获取,本章代码就是用这种方式

p.getlayer(IP)

这里一定要注意fields和res之间的代码转换,还有Try的错误处理,因为有些包会出现None,这样就会直接报错,所以一定要处理,这里一定要注意。

具体的解释在代码有详细解释。

1. **Dos 分析**

本节代码: dos\_analyze

1. 当一个连接的源和目的多次为同一个时,我们判定为dos攻击。

**本章的取值方式和1小节没太大区别,主要注意：**

1. try的报错处理,没有值的时候会报错。

2. sorted的使用。