

- 对抗：攻击方用 **nmap** 扫描（达到特定目的），防守方 **tcpdump** 监听，用 **wireshark** 分析（分析出攻击方扫描目的）
- 环境：vsphere  
攻击机：BackTrack4，172.31.4.178，工具 **nmap**  
靶机：Metasploitable\_Linux，172.31.4.188，工具 **tcpdump**  
侦察机：Ubuntu\_10.04，用 **tcpdump** 记录数据(由于演示需要，设置的靶机 **tcpdump** 显示数据包但是没有存储至文件，故使用侦察机记录)，记录使用的命令 **tcpdump -w listen.pcap host 172.31.4.188**
- 扫描：  
扫描过程及结果：参见 **attack\_defend.htm**，**nmap.txt**
- 监听结果：listen.pcap
- 分析参考：
  - 此次需要注意会有一些其他流量数据包，但是很少
  - 同时由于是在虚拟机里，数据报不向作业那 5 次那样规整，会有一堆请求和应答堆在一起的情况，但是不影响分析
  - 扫描工具的确定可以使用 **snort** 分析
  - 攻击者也可以从流量统计中确定

IPv4 Conversations: listen.pcap

IPv4 Conversations: 4

Address A	Address B	Packets	Bytes	Packets A->B	Bytes A->B	Packets A<-B	Bytes A<-B	Rel Start	Duration	bps A->B	bps A<-B
172.31.4.188	172.31.4.255	3	276	3	276	0	0	0.000000000	606.0209	3.64	N/A
172.31.4.2	172.31.4.188	3	312	3	312	0	0	0.000586000	606.0208	4.12	N/A
172.31.4.188	202.106.0.20	6	579	3	255	3	324	117.190435000	0.0709	28755.07	36535.86
172.31.4.178	172.31.4.188	135549	8148840	67855	4075596	67694	4073244	56.746593000	106.6988	305577.60	305401.25

Help Copy Close

- 扫描次数的鉴定可以参考时间间隔的分析，以及 **arp** 的分析(虚拟里 **arp** 的刷新好像很快，扫描前会发送 **arp** 请求)，但是注意并没有上次的那些 **ICMP** 和 **TCP ACK** 确定活跃性(**ARP** 也可以确认活跃)。
- 其他诸如开放端口的与作业的五次扫描类似，不多说
- 最后关于服务的扫描，可以发现不仅扫描是否开放，还进一步尝试连接，并可以观察到应用层协议的一些通信信息

No. ↓	Time	Source	Destination	Protocol	Info
9	56.746593	172.31.4.178	172.31.4.188	TCP	57738 > mysql [SYN] s
10	56.746636	172.31.4.188	172.31.4.178	TCP	mysql > 57738 [SYN, s
13	56.747127	172.31.4.178	172.31.4.188	TCP	57738 > mysql [RST] s
2073	84.186714	172.31.4.178	172.31.4.188	TCP	52374 > mysql [SYN] s
2083	84.190185	172.31.4.188	172.31.4.178	TCP	mysql > 52374 [SYN, s
2088	84.190499	172.31.4.178	172.31.4.188	TCP	52374 > mysql [RST] s
133248	115.881985	172.31.4.178	172.31.4.188	TCP	36892 > mysql [SYN] s
133249	115.881991	172.31.4.188	172.31.4.178	TCP	mysql > 36892 [SYN, s
133250	115.881995	172.31.4.178	172.31.4.188	TCP	36892 > mysql [RST] s
135262	117.167769	172.31.4.178	172.31.4.188	TCP	39890 > mysql [SYN] s
135268	117.170779	172.31.4.188	172.31.4.178	TCP	mysql > 39890 [SYN, s
135270	117.171025	172.31.4.178	172.31.4.188	TCP	39890 > mysql [ACK] s
135272	117.175396	172.31.4.188	172.31.4.178	MySQL	Server Greeting prot
135273	117.175411	172.31.4.178	172.31.4.188	TCP	39890 > mysql [ACK] s
135274	117.176026	172.31.4.178	172.31.4.188	TCP	39890 > mysql [FIN, s
135280	117.179032	172.31.4.188	172.31.4.178	TCP	mysql > 39890 [FIN, s
135281	117.179040	172.31.4.178	172.31.4.188	TCP	39890 > mysql [ACK] s
⊞ Frame 135272 (132 bytes on wire, 90 bytes captured)					
⊞ Ethernet II, Src: Vmware_94:35:d1 (00:50:56:94:35:d1), Dst: Vmware_94:15:a0 (00:50:56:94:15:a0)					
⊞ Internet Protocol, Src: 172.31.4.188 (172.31.4.188), Dst: 172.31.4.178 (172.31.4.178)					
⊞ Transmission Control Protocol, Src Port: mysql (3306), Dst Port: 39890 (39890), Seq: 1, Ack: 1,					
⊞ MySQL Protocol					
Packet Length: 62					
Packet Number: 0					
⊞ Server Greeting					
Protocol: 10					
Version: 5.0.51a-3ubuntu5					
[Packet size limited during capture: MySQL truncated]					