Honeynet.org Scan of the Month 30

By Ken Lee (*ahken@ahken.net*)

2004-03-25

Table of Contents

A.Acknowledgments

Many thanks to Nawapong Nakjang for telling me about this challenge and giving me many suggestions on the answers.

B.Prologue

In order to analyze the data more efficiently, a *gawk* script has been written to convert the *iptables* log entries into SQL INSERT statements which were then executed on a *MySQL* database. 4 tables have been designed to encapsulate the log entries. The *gawk* script and database schema can be found in Appendix A and B respectively.

After establishing the database, a couple of basic analysis on the honeynet environment were carried out. First of all, the IPs of the DNS servers were determined:

```
mysql> select distinct addr from ip where addr like '22.22.22.%' or addr like
'23.23.23.%';
+-------------+
| addr        |
+-------------+
| 22.22.22.40 |
| 23.23.23.60 |
+-------------+
2 rows in set (0.29 sec)
```

Then the incoming and outgoing interfaces were identified:

```
mysql> select  PHYSIN, count(*) from common where src like '11.11.11.%' group
by physin;
+--------+----------+
| PHYSIN | count(*) |
+--------+----------+
| eth0   |      193 |
| eth1   |    24161 |
+--------+----------+
2 rows in set (3.88 sec)

mysql> select  PHYSIN, count(*) from common where dst like '11.11.11.%' group
by physin;
+--------+----------+
| PHYSIN | count(*) |
+--------+----------+
| eth0   |   283363 |
| eth1   |     3929 |
+--------+----------+
2 rows in set (26.79 sec)
```

Hence It is obvious that the Internet interface was *eth0* while the intranet interface was *eth1*.

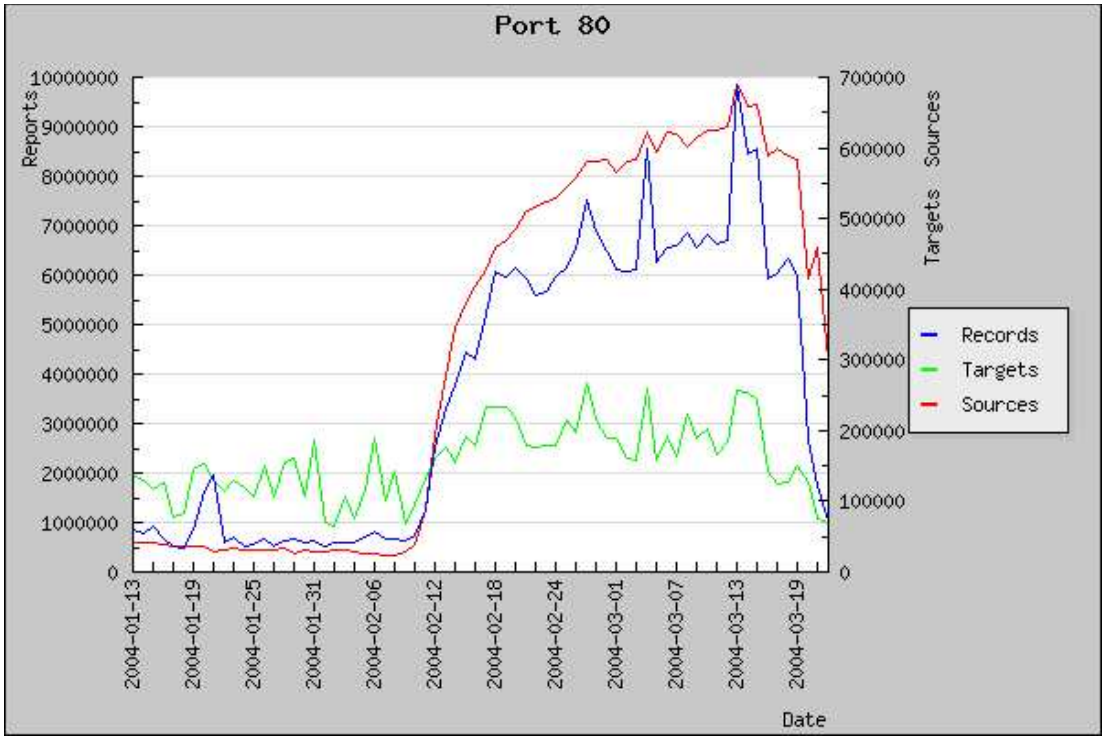
C.Answers to questions

1. What are the high-level trends in connectivity to/from the honeynet? What was growing/decreasing? How does that match global statistics from DShield and other sources?

Certain trends have been identified in incoming traffic on some ports. Here we shall look at

two of them:

According to DShield, in mid-February there has been a dramatic increase in attacks on TCP port 80:

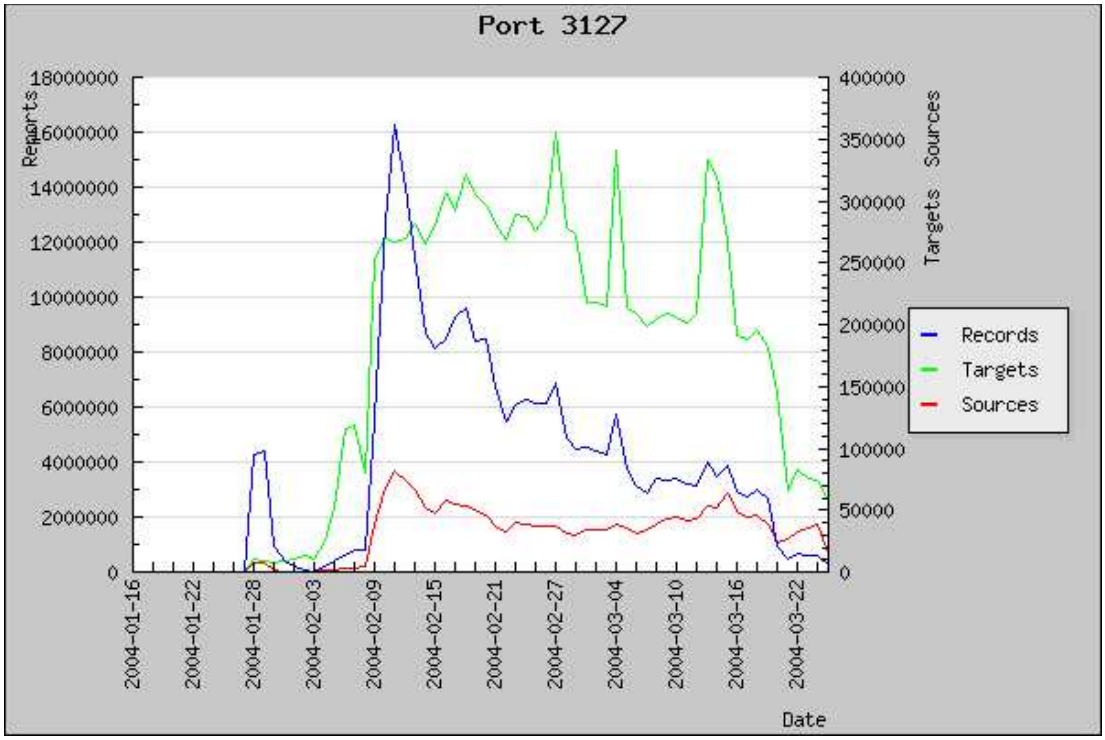


The same trend is observed in the honeynet logs as the attack count increased from an average of 319 probes per day to 430:

```
mysql> select avg(cnt) from (select date(dt), count(*) cnt from common,tcp
where tcp.log_id=common.log_id and  dpt=80 and physin='eth0' and flags like
'%SYN%' and date(dt)< '0000-02-12' group by date(dt)) tbl;
+----------+
| avg(cnt) |
+----------+
| 318.7273 |
+----------+
1 row in set (2.61 sec)

mysql> select avg(cnt) from (select date(dt), count(*) cnt from common,tcp
where tcp.log_id=common.log_id and  dpt=80 and physin='eth0' and flags like
'%SYN%' and date(dt)>= '0000-02-12' group by date(dt)) tbl;
+----------+
| avg(cnt) |
+----------+
| 429.8750 |
+----------+
1 row in set (4.85 sec)
```

Again according to DShield, there has been a sharp increase in activity on TCP port 3127 after Feb 8th:



The same is observed from the honeynet logs, with average traffic increasing from 70 per day to 1342. The near-linear decrease in traffic after Feb 8th observed by DShield is however not seen in the logs:

```
mysql> select avg(cnt) from (select date(dt), count(*) cnt from common,tcp
where tcp.log_id=common.log_id and  dpt=3127 and physin='eth0' and flags like
'%SYN%' and date(dt)< '0000-02-09' group by date(dt)) tbl;

+----------+
| avg(cnt) |
+----------+
|  69.7500 |
+----------+
1 row in set (9.56 sec)

mysql> select avg(cnt) from (select date(dt), count(*) cnt from common,tcp
where tcp.log_id=common.log_id and  dpt=3127 and physin='eth0' and flags like
'%SYN%' and date(dt)>= '0000-02-09' group by date(dt)) tbl;

+-----------+
| avg(cnt)  |
+-----------+
| 1342.2105 |
+-----------+
1 row in set (2.74 sec)
```

Traffic on some other popular TCP ports (135, 443, 445) have also been studied but no obvious trend has been observed in the honeynet logs.

2. What possible evidence of malware is there? what types? what are the malware trends you can observe?

i. MyDoom

One very obvious evidence of malware presence is the activity on TCP port 3127 seen around Feb 8th. Evidences of this being malware activity are:

1. This port has no legitimate use.
2. This port is used by MyDoom.
3. There has been a sudden increase in traffic after Feb 8th which was around the time when MyDoom was released.

One trend of such malware activity is that the increase is very sharp, in this case from below 200 probes before Feb 9th to 1272 probes on Feb 9th itself and more than a 1000 on the next 3 days peaking at 2161 probes. This is a good evidence of the astonishing speed at which worms like MyDoom can propagate. Below is the number of incoming connections on TCP 3127 on each day of the logs:

```
mysql> select date(dt), count(*) from common,tcp where tcp.log_id=common.log_id
and  dpt=3127 and physin='eth0' and flags like '%SYN%' group by date(dt);

+------------+----------+
| date(dt)   | count(*) |
+------------+----------+
| 0000-02-03 |       21 |
| 0000-02-05 |       62 |
| 0000-02-06 |      180 |
| 0000-02-07 |       16 |
| 0000-02-09 |     1272 |
| 0000-02-10 |     1310 |
| 0000-02-11 |     1144 |
| 0000-02-12 |     2161 |
| 0000-02-13 |      930 |
| 0000-02-14 |      678 |
| 0000-02-15 |     1634 |
| 0000-02-16 |     1415 |
| 0000-02-17 |      785 |
| 0000-02-18 |     1170 |
| 0000-02-19 |     1271 |
| 0000-02-20 |      854 |
| 0000-02-21 |     4913 |
| 0000-02-22 |     1630 |
| 0000-02-23 |      685 |
| 0000-02-24 |      873 |
| 0000-02-25 |     1426 |
| 0000-02-26 |     1019 |
| 0000-02-27 |      332 |
+------------+----------+
23 rows in set (17.76 sec)
```

ii. Blaster

There have been a huge number of traffic on TCP port 135:

```
mysql> select count(*) from common, tcp where common.log_id=tcp.log_id and
physin='eth0' and dpt=135;
+----------+
| count(*) |
+----------+
|    86634 |
+----------+
1 row in set (4.99 sec)
```

This might suggest *Blaster* activity but since it could also be from port scanners, there is no solid proof of it being malware activity.

iii. NetBus

```
mysql> select src, count(*) from common, tcp where common.log_id=tcp.log_id and
dpt=12345 group by src order by src;
+---------------+----------+
| src           | count(*) |
+---------------+----------+
| 66.76.165.22  |       71 |
| 67.37.178.253 |       50 |
| 68.20.10.54   |       61 |
+---------------+----------+
3 rows in set (0.26 sec)
```

iv. Kuang2

```
mysql> select src, count(*) from common, tcp where common.log_id=tcp.log_id and
dpt=17300 group by src order by src;
+-----------------+----------+
| src             | count(*) |
+-----------------+----------+
| 141.153.150.33  |       63 |
| 172.158.160.96  |        6 |
| 172.181.1.44    |        3 |
| 172.181.56.142  |       71 |
| 172.211.62.73   |       52 |
| 200.244.46.200  |        6 |
| 200.56.85.164   |       41 |
| 200.70.171.139  |       69 |
| 201.5.52.80     |       72 |
| 210.216.129.142 |       45 |
| 211.186.116.19  |       43 |
| 211.187.127.51  |       64 |
| 211.190.102.125 |       40 |
| 211.244.171.181 |       71 |
| 211.40.233.200  |       42 |
| 211.59.62.116   |       41 |
| 217.129.27.148  |       13 |
| 217.132.9.229   |        8 |
| 218.10.140.216  |       22 |
| 219.74.114.249  |       70 |
| 220.255.85.212  |      196 |
| 24.240.242.177  |       20 |
| 4.8.222.111     |       30 |
| 61.110.150.34   |       72 |
| 61.249.164.76   |        4 |
```

```
| 64.173.104.79   |       60 |
| 65.70.14.229    |       70 |
| 66.169.174.139  |       72 |
| 66.177.147.150  |       32 |
| 66.203.183.137  |       52 |
| 66.25.127.73    |       36 |
| 67.125.27.6     |       47 |
| 67.160.185.102  |       70 |
| 68.161.242.167  |       60 |
| 68.226.148.161  |       70 |
| 68.237.62.208   |       47 |
| 68.47.3.189     |        3 |
| 68.80.233.162   |       59 |
| 68.93.101.207   |       70 |
| 69.132.40.215   |       18 |
| 81.133.145.97   |       70 |
| 81.185.49.218   |       47 |
| 81.218.250.166  |        1 |
| 81.223.242.164  |       36 |
| 81.226.105.5    |       18 |
| 81.250.182.138  |        8 |
| 82.166.90.209   |       37 |
+-----------------+----------+
47 rows in set (1.46 sec)
```

v. SubSeven

```
mysql> select src, count(*) from common, tcp where common.log_id=tcp.log_id and
dpt=27374 group by src order by src;
+-----------------+----------+
| src             | count(*) |
+-----------------+----------+
| 12.44.237.134   |       53 |
| 203.210.200.226 |       72 |
| 24.166.205.144  |       70 |
| 62.178.148.173  |       45 |
| 66.191.22.199   |       69 |
| 67.37.178.253   |       49 |
| 67.4.174.149    |       46 |
| 68.20.10.54     |       62 |
| 82.72.185.133   |       70 |
+-----------------+----------+
9 rows in set (0.40 sec)
```

3. What types of reconnaissance activity you notice? What do you think they were looking for? What are some of the notorious sources of such activity in the files?

Many incoming ICMP type 8 (ECHO-REQUEST) packets have been seen throughout the logs. This pinging action is a quick way to determine whether a machine is up. Some attackers observed in the logs actually repeatedly pinged the honeypot machines everyday during the period covered by the logs. The reason of doing this is probably to constantly check for machines that are brought up recently. In particular, four such sources are identified:

```
mysql> select src, count(*) cnt from common, icmp where
common.log_id=icmp.log_id and physin='eth0' group by src having cnt > 100 order
by cnt desc limit 10;
+----------------+------+
| src            | cnt  |
+----------------+------+
| 63.123.38.103  | 3928 |
| 63.123.70.166  | 3219 |
| 63.125.10.7    | 3087 |
| 63.126.190.227 | 1031 |
+----------------+------+
4 rows in set (2.09 sec)
```

The following query takes a closer look at the top entry, namely 63.123.38.103:

```
mysql> select date(dt), count(distinct dst), count(*) from common where
src='63.123.38.103' and proto='ICMP' group by date(dt) order by dt;
+------------+----------------------+----------+
| date(dt)   | count(distinct dst)  | count(*) |
+------------+----------------------+----------+
| 0000-02-01 |                   24 |      144 |
| 0000-02-02 |                   24 |      168 |
| 0000-02-03 |                   24 |      120 |
| 0000-02-04 |                   24 |       96 |
| 0000-02-05 |                   24 |      192 |
| 0000-02-06 |                   24 |      192 |
| 0000-02-07 |                   24 |      145 |
| 0000-02-08 |                   24 |      144 |
| 0000-02-09 |                   24 |      144 |
| 0000-02-10 |                   24 |      100 |
| 0000-02-11 |                   24 |       96 |
| 0000-02-12 |                   24 |      168 |
| 0000-02-13 |                   24 |      144 |
| 0000-02-14 |                   24 |      144 |
| 0000-02-15 |                   24 |      192 |
| 0000-02-16 |                   24 |       96 |
| 0000-02-17 |                   24 |      192 |
| 0000-02-18 |                   24 |      168 |
| 0000-02-19 |                   24 |      168 |
| 0000-02-20 |                   24 |      120 |
| 0000-02-21 |                   24 |      179 |
| 0000-02-22 |                   24 |      168 |
| 0000-02-23 |                   24 |       96 |
| 0000-02-24 |                   24 |      120 |
| 0000-02-25 |                   24 |      168 |
| 0000-02-26 |                   24 |      144 |
| 0000-02-27 |                   24 |      120 |
+------------+----------------------+----------+
27 rows in set (0.38 sec)
```

```
mysql> select count(distinct dst) from common where src='63.123.38.103' and
proto='ICMP';

+----------------------+
| count(distinct dst) |
+----------------------+
|                   24 |
+----------------------+
1 row in set (0.26 sec)
```

The attacker has actually been pinging the honeypot machines from the first day (Feb 1st) to the last day (Feb 27th). Interestingly, the pings seem to be targeted at the same 24 machines throughout the whole month, day after day.

Another kind of reconnaissance activity is also seen in the logs once:

```
mysql> select dt, src, dst, proto, flags, spt, dpt  from common, tcp where
tcp.log_id=122836 and common.log_id=122836;;
+---------------------+---------------+-------------+-------+--------+------+------+
| dt                  | src           | dst         | proto | flags  | spt  | dpt  |
+---------------------+---------------+-------------+-------+--------+------+------+
| 0000-02-11 14:43:47 | 64.218.200.74 | 11.11.11.75 | TCP   | DF,SYN | 3958 |   79 |
+---------------------+---------------+-------------+-------+--------+------+------+
1 row in set (0.17 sec)
```

This is a attempted connection to the finger service, whose presence can potentially give useful information about valid usernames to the attacker.

4. What are the different scan patterns (sequential, etc) you can notice? Do you think all come from different attack tools? Any long term ("low and slow") scanning activity?

Some active attackers' activity are shown in the following queries:

```
mysql> select dt, dst, dpt, flags from tcp, common where
common.log_id=tcp.log_id and src='200.208.28.39' order by dt;
+---------------------+-------------+-------+------------+
| dt                  | dst         | dpt   | flags      |
+---------------------+-------------+-------+------------+
| 0000-02-01 05:46:30 | 11.11.11.80  | 12986 | DF,ACK,SYN |
| 0000-02-01 13:38:01 | 11.11.11.75  | 42842 | DF,ACK,SYN |
| 0000-02-01 17:37:27 | 11.11.11.83  | 56969 | DF,ACK,SYN |
| 0000-02-01 17:51:25 | 11.11.11.110 | 47887 | DF,ACK,SYN |
.......... << Lines removed for brevity >> ...........
| 0000-02-18 05:41:08 | 11.11.11.81  | 16279 | DF,ACK,SYN |
| 0000-02-18 07:45:59 | 11.11.11.85  | 53897 | DF,ACK,SYN |
| 0000-02-18 09:47:16 | 11.11.11.87  | 48856 | DF,ACK,SYN |
| 0000-02-18 23:58:47 | 11.11.11.81  | 25408 | DF,ACK,SYN |
+---------------------+-------------+-------+------------+
59 rows in set (0.44 sec)

mysql> select dt, src, dst, dpt, flags from tcp, common where
common.log_id=tcp.log_id and src='218.22.13.10' order by dt;

+---------------------+-------------+--------------+-------+------------+
| dt                  | src         | dst          | dpt   | flags      |
+---------------------+-------------+--------------+-------+------------+
| 0000-02-01 03:02:47 | 218.22.13.10 | 11.11.11.82  | 37677 | DF,ACK,SYN |
| 0000-02-01 08:01:16 | 218.22.13.10 | 11.11.11.87  | 43275 | DF,ACK,SYN |
| 0000-02-01 10:02:16 | 218.22.13.10 | 11.11.11.89  | 23170 | DF,ACK,SYN |
| 0000-02-01 10:45:44 | 218.22.13.10 | 11.11.11.125 | 21976 | DF,ACK,SYN |
.......... << Lines removed for brevity >> ...........
| 0000-02-04 11:45:09 | 218.22.13.10 | 11.11.11.73  | 11843 | DF,ACK,SYN |
| 0000-02-04 12:39:30 | 218.22.13.10 | 11.11.11.85  | 20502 | DF,ACK,SYN |
| 0000-02-04 14:56:47 | 218.22.13.10 | 11.11.11.72  | 24379 | DF,ACK,SYN |
| 0000-02-04 17:51:58 | 218.22.13.10 | 11.11.11.82  | 37677 | DF,ACK,SYN |
+---------------------+-------------+--------------+-------+------------+
60 rows in set (0.05 sec)

mysql> select dt, src, dst, dpt, flags from tcp, common where
common.log_id=tcp.log_id and src='220.113.34.16' order by dt;

+---------------------+--------------+--------------+------+--------+
| dt                  | src          | dst          | dpt  | flags  |
+---------------------+--------------+--------------+------+--------+
| 0000-02-19 21:04:03 | 220.113.34.16 | 11.11.11.64  |   80 | DF,SYN |
| 0000-02-19 21:04:03 | 220.113.34.16 | 11.11.11.64  | 8080 | DF,SYN |
| 0000-02-19 21:04:03 | 220.113.34.16 | 11.11.11.64  | 8000 | DF,SYN |
| 0000-02-19 21:04:04 | 220.113.34.16 | 11.11.11.64  | 3128 | DF,SYN |
| 0000-02-19 21:04:06 | 220.113.34.16 | 11.11.11.64  |   81 | DF,SYN |
| 0000-02-19 21:04:06 | 220.113.34.16 | 11.11.11.64  |   88 | DF,SYN |
| 0000-02-19 21:04:06 | 220.113.34.16 | 11.11.11.64  | 8081 | DF,SYN |
| 0000-02-19 21:04:06 | 220.113.34.16 | 11.11.11.64  | 8888 | DF,SYN |
| 0000-02-19 21:04:07 | 220.113.34.16 | 11.11.11.64  |   82 | DF,SYN |
| 0000-02-19 21:04:07 | 220.113.34.16 | 11.11.11.64  | 8002 | DF,SYN |
| 0000-02-19 21:04:07 | 220.113.34.16 | 11.11.11.67  |   80 | DF,SYN |
| 0000-02-19 21:04:07 | 220.113.34.16 | 11.11.11.64  | 8001 | DF,SYN |
| 0000-02-19 21:04:08 | 220.113.34.16 | 11.11.11.67  | 8888 | DF,SYN |
| 0000-02-19 21:04:08 | 220.113.34.16 | 11.11.11.67  |   81 | DF,SYN |
| 0000-02-19 21:04:08 | 220.113.34.16 | 11.11.11.67  | 8080 | DF,SYN |
| 0000-02-19 21:04:08 | 220.113.34.16 | 11.11.11.67  | 8000 | DF,SYN |
| 0000-02-19 21:04:08 | 220.113.34.16 | 11.11.11.67  | 3128 | DF,SYN |
.......... << Lines removed for brevity >> ...........
| 0000-02-19 21:04:24 | 220.113.34.16 | 11.11.11.120 | 8002 | DF,SYN |
| 0000-02-19 21:04:24 | 220.113.34.16 | 11.11.11.120 | 8000 | DF,SYN |
| 0000-02-19 21:04:24 | 220.113.34.16 | 11.11.11.120 |   82 | DF,SYN |
| 0000-02-19 21:04:24 | 220.113.34.16 | 11.11.11.110 | 8081 | DF,SYN |
+---------------------+--------------+--------------+------+--------+
110 rows in set (0.15 sec)
```

It can be seen that many different patterns and methods of scanning were used. In particular, the first attacker seemed to be probing random ports with TCP SYN-ACK packets. The second attacker probed a few specific high TCP ports, again with SYN-ACK packets but in a random sequence. Meanwhile, the third attacker probed with SYN packets in a sequential patterns on each IP and a few specific ports.

Furthermore, the following query reveals a typical scan performed by a port-scan tool:

```
mysql> select time(dt), src, dst, proto, dpt, flags from common left join tcp
on common.log_id=tcp.log_id where src='66.186.83.178' order by dt;
+----------+---------------+--------------+-------+------+---------+
| time(dt) | src           | dst          | proto | dpt  | flags   |
+----------+---------------+--------------+-------+------+---------+
| 06:45:42 | 66.186.83.178 | 11.11.11.64  | ICMP  | NULL |         |
| 06:45:43 | 66.186.83.178 | 11.11.11.64  | ICMP  | NULL |         |
| 06:45:44 | 66.186.83.178 | 11.11.11.67  | ICMP  | NULL |         |
| 06:45:45 | 66.186.83.178 | 11.11.11.64  | ICMP  | NULL |         |
| 06:45:45 | 66.186.83.178 | 11.11.11.69  | ICMP  | NULL |         |
| 06:45:48 | 66.186.83.178 | 11.11.11.70  | ICMP  | NULL |         |
| 06:45:49 | 66.186.83.178 | 11.11.11.71  | ICMP  | NULL |         |
| 06:45:49 | 66.186.83.178 | 11.11.11.72  | ICMP  | NULL |         |
| 06:45:49 | 66.186.83.178 | 11.11.11.73  | ICMP  | NULL |         |
| 06:45:53 | 66.186.83.178 | 11.11.11.75  | ICMP  | NULL |         |
| 06:45:57 | 66.186.83.178 | 11.11.11.67  | TCP   |  445 | DF,SYN  |
| 06:45:57 | 66.186.83.178 | 11.11.11.67  | TCP   |  139 | DF,SYN  |
| 06:45:58 | 66.186.83.178 | 11.11.11.67  | TCP   |  445 | DF,SYN  |
| 06:45:58 | 66.186.83.178 | 11.11.11.67  | TCP   |  445 | DF,SYN  |
| 06:45:58 | 66.186.83.178 | 11.11.11.69  | TCP   |  445 | DF,SYN  |
| 06:45:58 | 66.186.83.178 | 11.11.11.69  | TCP   |  139 | DF,SYN  |
| 06:45:58 | 66.186.83.178 | 11.11.11.69  | TCP   |  445 | DF,SYN  |
| 06:45:58 | 66.186.83.178 | 11.11.11.69  | TCP   |  445 | DF,SYN  |
| 06:46:01 | 66.186.83.178 | 11.11.11.70  | TCP   |  445 | DF,SYN  |
| 06:46:02 | 66.186.83.178 | 11.11.11.70  | TCP   |  139 | DF,SYN  |
..... << Lines removed for brevity >> .....
| 07:22:51 | 66.186.83.178 | 11.11.11.125 | TCP   |  139 | DF,SYN  |
| 07:22:52 | 66.186.83.178 | 11.11.11.125 | TCP   |  445 | DF,SYN  |
| 07:22:52 | 66.186.83.178 | 11.11.11.125 | TCP   |  445 | DF,SYN  |
| 07:22:52 | 66.186.83.178 | 11.11.11.125 | TCP   |  445 | DF,SYN  |
| 07:22:53 | 66.186.83.178 | 11.11.11.125 | TCP   |  445 | DF,SYN  |
+----------+---------------+--------------+-------+------+---------+
10217 rows in set (1.72 sec)
```

It can be seen that the scan began with a phase of sending out ICMP ping packets to the target machines, followed by the actual probes on designated ports with SYN packets.

Another interesting attacker did a much slower scan:

```
mysql> select dt, dst, dpt, flags from common, tcp where
common.log_id=tcp.log_id and src='194.128.177.225' order by dt, dst, dpt;
+---------------------+-------------+------+------------+
| dt                  | dst         | dpt  | flags      |
+---------------------+-------------+------+------------+
| 0000-02-13 09:02:29 | 11.11.11.90 |   80 | DF,SYN     |
| 0000-02-13 09:07:03 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 09:11:33 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 09:16:03 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 09:20:33 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 09:25:03 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 09:29:34 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 09:34:04 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 09:38:34 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 09:43:04 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 09:47:34 | 11.11.11.90 |   80 | DF,ACK,FIN |
```

```
| 0000-02-13 09:52:04 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 09:56:34 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 10:01:04 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 10:05:35 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 10:10:05 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 10:14:35 | 11.11.11.90 |   80 | DF,ACK,FIN |
```

```
| 0000-02-13 10:19:05 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 10:23:35 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 10:28:05 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 10:32:35 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-13 10:37:05 | 11.11.11.90 |   80 | DF,ACK,FIN |
.......... << Lines removed for brevity >> ...........
| 0000-02-19 10:53:48 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 10:58:19 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:02:49 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:07:19 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:11:49 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:16:20 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:20:49 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:25:19 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:29:50 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:34:20 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:38:50 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:43:20 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:47:50 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:52:20 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 11:56:50 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 12:01:20 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 12:05:51 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 12:10:21 | 11.11.11.90 |   80 | DF,ACK,FIN |
| 0000-02-19 12:11:25 | 11.11.11.90 |   80 | DF,ACK,FIN |
+---------------------+-------------+------+------------+
1960 rows in set (0.61 sec)
```

It can be observed that the probes spaced out in intervals of about 5 minutes which is quite slow in terms of port scans. Incidentally, there is an option '-T Paranoid' in *nmap* which sends out probes every 5 minutes. Another observation is that the probes were dedicated to TCP port 80 (HTTP) and that the scan method was with FIN-ACK packets.

The following is an attempt to identify the use of *nmap*:

```
mysql> select dt, src, dst, flags, window from common, tcp where
common.log_id=tcp.log_id and window=3072;
+---------------------+-----------------+--------------+--------+--------+
| dt                  | src             | dst          | flags  | window |
+---------------------+-----------------+--------------+--------+--------+
| 0000-02-16 01:47:21 | 220.215.103.131 | 11.11.11.125 | DF,SYN |   3072 |
| 0000-02-16 01:47:22 | 220.215.103.131 | 11.11.11.125 | DF,SYN |   3072 |
| 0000-02-16 01:47:23 | 220.215.103.131 | 11.11.11.125 | DF,SYN |   3072 |
+---------------------+-----------------+--------------+--------+--------+
3 rows in set (0.25 sec)
```

According to the *p0f* fingerprints, only *nmap* sends out SYN packets with window size 3072. The above is one attacker who was apparently using *nmap* SYN scans. The estimation is no more than an educated guess though, as the stack fingerprints are not comprehensive.

5. What other common internet noise types do you see?

1. Multicast (IGMP) messages:

```
mysql> select date(dt), src, dst from common where proto='IGMP';
+------------+-------------+-----------+
| date(dt)   | src         | dst       |
+------------+-------------+-----------+
| 0000-02-03 | 11.11.11.67 | 224.0.0.2 |
| 0000-02-03 | 11.11.11.67 | 224.0.1.1 |
| 0000-02-03 | 11.11.11.67 | 224.0.1.1 |
| 0000-02-03 | 11.11.11.67 | 224.0.1.1 |
| 0000-02-19 | 11.11.11.67 | 224.0.1.1 |
| 0000-02-19 | 11.11.11.67 | 224.0.1.1 |
| 0000-02-19 | 11.11.11.67 | 224.0.1.1 |
+------------+-------------+-----------+
7 rows in set (3.16 sec)
```

2. Windows NetBIOS (UDP) broadcasts:

```
mysql> select distinct src, dst, dpt from common, udp  where
common.log_id=udp.log_id and dst like '%.255';
+-------------+--------------+------+
| src         | dst          | dpt  |
+-------------+--------------+------+
| 11.11.11.67 | 11.11.11.255 |  138 |
| 11.11.11.67 | 11.11.11.255 |  137 |
| 11.11.11.69 | 11.11.11.255 |  137 |
| 11.11.11.69 | 11.11.11.255 |  138 |
+-------------+--------------+------+
4 rows in set (2.20 sec)
```

6. Any unidentified/anomalous traffic observed? Please suggest hypothesis for why it is there and what it indicates.

One kind of anomalous traffic were packets with private addresses as the source. As can be seen from below, there were at least 37 such addresses observed in the incoming traffic:

```
mysql> select addr, in_count from ip where addr like '127.%' or addr like
'10.%' or addr like '169.254.%' or addr like '172.__.%' order by addr;
+-----------------+----------+
| addr            | in_count |
+-----------------+----------+
| 10.0.1.36       |        1 |
| 10.0.13.17      |        1 |
| 10.0.56.65      |        1 |
| 10.1.2.206      |        1 |
| 10.10.129.20    |        1 |
| 10.10.136.114   |        1 |
| 10.10.228.18    |       59 |
| 10.10.4.7       |        2 |
| 10.10.51.99     |        1 |
| 10.100.70.2     |        1 |
| 10.11.4.117     |        1 |
| 10.11.4.203     |        1 |
| 10.165.241.41   |        2 |
| 10.187.0.151    |        1 |
| 10.21.4.50      |        1 |
| 10.25.5.43      |        1 |
| 10.250.230.151  |        2 |
| 10.250.230.201  |        1 |
| 10.30.101.157   |        3 |
| 10.39.3.33      |        3 |
| 10.55.100.210   |        1 |
| 10.63.214.3     |        1 |
| 10.63.214.5     |        1 |
| 10.65.1.7       |        1 |
| 127.0.0.1       |     6394 |
| 169.254.185.195 |       51 |
| 169.254.233.189 |      169 |
| 172.16.1.42     |        1 |
| 172.16.25.206   |        1 |
| 172.16.26.76    |        1 |
| 172.16.3.74     |        4 |
| 172.16.30.7     |        1 |
| 172.17.24.153   |        1 |
| 172.17.3.59     |       72 |
| 172.20.10.166   |        1 |
| 172.24.255.37   |        1 |
| 172.31.37.99    |        1 |
+-----------------+----------+
37 rows in set (0.49 sec)
```

Since the addresses are unroutable on the Internet, they were of little use to the attackers in enumerating or attacking a machine since the reply of such packets were not sent back. Considering the fact that many of the above sources had only a few packet counts, it is fair to guess that they were used as decoys in a port scan, such as those used by *nmap* when the -D parameter is specified.

Another strange finding is that the honeypot machines never replied to SYN packets with SYN-ACK packets but with ACK packets instead, and that the ACK packets were sent a long time after the SYN had been received. One hypothesis is that the honeypot machines were running a non-standard TCP/IP stack, or one that is emulated by a low-interaction honeypot. This is backed up by the fact that the outgoing TCP window size for SYN packets was always

5840, which, according to *p0f* fingerprints, identifies the honeypot machines as Zaurus PDAs, which is quite improbable:

```
mysql> select distinct window from common, tcp where common.log_id=tcp.log_id
and src like '11.11.11.%' and flags like '%SYN%';
+--------+
| window |
+--------+
|   5840 |
+--------+
1 row in set (4.63 sec)
```

In addition, a strange internal IP is observed: 11.11.11.65. Apparently no external traffic is directed to this machine and strange incoming/outgoing interfaces are observed in the logs:

```
mysql> select distinct src, dst, dpt, physin, in_intf, physout, out from
common, udp where common.log_id=udp.log_id and (src='11.11.11.65' or
dst='11.11.11.65');
+-------------+-------------+------+--------+---------+---------+------+
| src         | dst         | dpt  | physin | in_intf | physout | out  |
+-------------+-------------+------+--------+---------+---------+------+
| 11.11.11.67 | 11.11.11.65 |  514 | eth1   | br0     | eth0    | br0  |
| 11.11.11.69 | 11.11.11.65 |  514 | eth0   | eth1    | eth1    |      |
| 11.11.11.67 | 11.11.11.65 |  514 | eth0   | eth1    | eth1    |      |
+-------------+-------------+------+--------+---------+---------+------+
```

Since the only traffic on 11.11.11.65 is incoming on UDP port 514 (syslog), it is assumed that this is a log server behind the firewall and is thus inaccessible from the Internet.

7. Was the honeypot compromised during the observed time period? How do you know?

It has probably not been compromised during the period since no outgoing TCP SYN-ACK packet has been seen:

```
mysql> select count(*) from common where proto='TCP' and flags like '%SYN%' and
flags like '%ACK%' and physout='eth0';
+----------+
| count(*) |
+----------+
|        0 |
+----------+
1 row in set (5.85 sec)
```

Hence no attacker was able to successfully connect to any of the machines on a TCP port. As for UDP ports, even if someone did compromise a machine through one of them, it is still quite likely that a telnet/ssh backdoor would have been created and that the attacker would have connected back. There was no such traffic however, as no SYN-ACK has been seen. Hence it is quite unlikely that any of the honeypot machines has been compromised.

With that said, there are still doubts on some suspicious outgoing TCP SYN connections from the honeypot machines to TCP port 80:

```
mysql> select distinct dst, count(*) cnt from common, tcp where
common.log_id=tcp.log_id and dpt=80 and physout='eth0' group
by dst order by cnt desc;
+----------------+-----+
| dst            | cnt |
+----------------+-----+
| 207.66.155.21  |  23 |
| 62.211.66.12   |   4 |
| 195.27.176.155 |   3 |
| 209.63.57.10   |   3 |
+----------------+-----+
4 rows in set (1.00 sec)
```

The highlighted entry happens to be a Italian ISP which also provides web hosting service. The reason this is suspicious is that if the honeypot was compromised, the attacker might want to get his tools e.g. root-kits from somewhere. This is usually done through HTTP or FTP. But the log showed that only outgoing SYNs to port 80 were sent without incoming ACKs and hence the outgoing connection was unsuccessful. So this suspicion cannot be confirmed.

8. If you'd obtain such firewall logs from a production system, what source IPs or groups of such IPs you'd focus on as a highest threat?

The answer to this question very much depends on the production system itself. Here are a few factors to consider:

1. Open ports on the production servers

   Assuming that a perimeter firewall is in place and that it is correctly configured, most probes/attacks can be safely filtered out. The only traffic that pose a threat to the system are those targeted at an open port. For example, if the server is providing e-mail service, TCP port 25 has to be open and cannot be completely protected by the firewall. If the attacker is dedicating his attacks on ports that include TCP 25, the threat against the system is high. One such attacker is identified in the logs. The attacker's probes were exclusively on TCP 25:

```
mysql> select distinct proto from common where src='220.96.49.232';
+-------+
| proto |
+-------+
| TCP   |
+-------+
1 row in set (0.14 sec)

mysql> select distinct dpt from common, tcp where common.log_id=tcp.log_id and
src='220.96.49.232';
+------+
| dpt  |
+------+
|   25 |
+------+
1 row in set (0.15 sec)
```

2. Intent of the attacker

   The behaviour of attackers varies a lot. Some of them are just performing opportunistic scans in the hope of finding a live machine and some open ports that can be attacked. On the other hand, some attackers keep on probing a set of IPs persistently. The latter kind is potentially more of a threat to the system since they are more dedicated to attacking the system in question. It is also more likely that such attackers do have a target in mind than their being ordinary script kiddies. Several such attackers have been identified from the logs:

```
mysql> select date(dt), dpt, count(*) from common, tcp where
common.log_id=tcp.log_id and src='63.123.70.166' group by date(dt), dpt order
by date(dt), dpt;
+------------+------+----------+
| date(dt)   | dpt  | count(*) |
+------------+------+----------+
| 0000-02-01 |  135 |      220 |
| 0000-02-02 |  135 |      167 |
| 0000-02-03 |  135 |      150 |
| 0000-02-04 |  135 |      152 |
| 0000-02-05 |  135 |      173 |
| 0000-02-06 |  135 |      203 |
| 0000-02-07 |  135 |      117 |
| 0000-02-08 |  135 |      105 |
| 0000-02-09 |  135 |      171 |
```

```
| 0000-02-10 |  135 |       72 |
| 0000-02-11 |  135 |       53 |
| 0000-02-12 |  135 |      103 |
| 0000-02-13 |  135 |      229 |
| 0000-02-14 |  135 |      123 |
| 0000-02-15 |  135 |      197 |
| 0000-02-16 |  135 |      136 |
| 0000-02-17 |  135 |      119 |
| 0000-02-18 |  135 |      222 |
| 0000-02-19 |  135 |      194 |
| 0000-02-20 |  135 |       68 |
| 0000-02-21 |  135 |      108 |
| 0000-02-22 |  135 |      139 |
| 0000-02-23 |  135 |      179 |
| 0000-02-24 |  135 |      180 |
| 0000-02-25 |  135 |      204 |
| 0000-02-26 |  135 |      134 |
| 0000-02-27 |  135 |      100 |
+------------+------+----------+
27 rows in set (1.99 sec)

mysql> select date(dt), dpt, count(*) from common, tcp where
common.log_id=tcp.log_id and src='63.123.70.166' group by date(dt), dpt order
by date(dt), dpt;
+------------+------+----------+
| date(dt)   | dpt  | count(*) |
+------------+------+----------+
| 0000-02-01 |  135 |      220 |
| 0000-02-02 |  135 |      167 |
| 0000-02-03 |  135 |      150 |
| 0000-02-04 |  135 |      152 |
| 0000-02-05 |  135 |      173 |
| 0000-02-06 |  135 |      203 |
| 0000-02-07 |  135 |      117 |
| 0000-02-08 |  135 |      105 |
| 0000-02-09 |  135 |      171 |
| 0000-02-10 |  135 |       72 |
| 0000-02-11 |  135 |       53 |
| 0000-02-12 |  135 |      103 |
| 0000-02-13 |  135 |      229 |
| 0000-02-14 |  135 |      123 |
| 0000-02-15 |  135 |      197 |
| 0000-02-16 |  135 |      136 |
| 0000-02-17 |  135 |      119 |
| 0000-02-18 |  135 |      222 |
| 0000-02-19 |  135 |      194 |
| 0000-02-20 |  135 |       68 |
| 0000-02-21 |  135 |      108 |
| 0000-02-22 |  135 |      139 |
| 0000-02-23 |  135 |      179 |
| 0000-02-24 |  135 |      180 |
| 0000-02-25 |  135 |      204 |
| 0000-02-26 |  135 |      134 |
| 0000-02-27 |  135 |      100 |
+------------+------+----------+
27 rows in set (0.61 sec)

mysql> select date(dt), dpt, count(*) from common, tcp where
common.log_id=tcp.log_id and src='63.125.10.7' group by date(dt), dpt order by
date(dt), dpt;
+------------+------+----------+
| date(dt)   | dpt  | count(*) |
+------------+------+----------+
| 0000-02-01 |  135 |      142 |
| 0000-02-02 |  135 |      186 |
| 0000-02-03 |  135 |      118 |
| 0000-02-04 |  135 |      124 |
| 0000-02-05 |  135 |      146 |
| 0000-02-06 |  135 |      117 |
| 0000-02-07 |  135 |      121 |
| 0000-02-08 |  135 |      140 |
| 0000-02-09 |  135 |      122 |
| 0000-02-10 |  135 |       67 |
| 0000-02-11 |  135 |       99 |
```

```
| 0000-02-12 |   135 |        97 |
| 0000-02-13 |   135 |       174 |
| 0000-02-14 |   135 |       215 |
| 0000-02-15 |   135 |       167 |
| 0000-02-16 |   135 |       116 |
| 0000-02-17 |   135 |       185 |
| 0000-02-18 |   135 |       167 |
| 0000-02-19 |   135 |       123 |
| 0000-02-20 |   135 |       152 |
| 0000-02-21 |   135 |       121 |
| 0000-02-22 |   135 |       172 |
| 0000-02-23 |   135 |       154 |
| 0000-02-24 |   135 |       134 |
| 0000-02-25 |   135 |       172 |
| 0000-02-26 |   135 |       187 |
| 0000-02-27 |   135 |        77 |
+------------+-------+-----------+
27 rows in set (0.97 sec)
```

As can be seen from above the attackers kept coming back everyday throughout February. It is also worth mentioning that such activity might also have come from worms such as *Blaster*, though in many worm implementations the target IPs are randomly generated. Seldom are they dedicated to a predefined list of targets since this is against their purpose of achieving the widest propagation possible.

During the course of attempting question 9, another dedicated attacker has also been found in the logs. His activity is studied in details in the answer to that question.

3. Skills of the attacker

   The skills of an attacker is also an important factor in determining his threat. This is however quite hard to tell from a firewall log like the one given.

4. The amount of incoming probes

   Even if the probes themselves are not harmless, a overwhelming of such traffic can still use up significant network/computer resources i.e. they can result in a denial-of-service attack to the production system. Hence it is still important that the administrator should block traffic from active port scanners, such as the following two:

```
mysql> select src, count(*) cnt from common where physin='eth0' group by src
having cnt > 10000 order by cnt desc;
+----------------+-------+
| src            | cnt   |
+----------------+-------+
| 66.60.166.84   | 21829 |
| 66.186.83.178  | 10217 |
+----------------+-------+
  2 rows in set (22.43 sec)
```

9. What honeypot systems were attacked the most? What ports were open on each of them? Why do you think a machines with close IP addresses were attacked differently?

Shown below are the frequency of attacks on each machine, sorted by number of packets received in descending order:

```
mysql> select dst, count(*) c from common where physin='eth0' group by dst
order by c desc;
+--------------+-------+
| dst          | c     |
+--------------+-------+
| 11.11.11.75  | 30730 |
| 11.11.11.80  | 14446 |
| 11.11.11.100 | 13316 |
| 11.11.11.105 | 13267 |
| 11.11.11.67  | 12981 |
| 11.11.11.110 | 12909 |
| 11.11.11.90  | 12527 |
| 11.11.11.71  | 11653 |
| 11.11.11.87  | 11580 |
| 11.11.11.70  | 11287 |
| 11.11.11.69  | 11107 |
| 11.11.11.115 | 11089 |
| 11.11.11.73  | 10916 |
| 11.11.11.72  | 10772 |
| 11.11.11.82  | 10657 |
| 11.11.11.81  | 10398 |
| 11.11.11.125 | 10272 |
| 11.11.11.95  | 10169 |
| 11.11.11.83  | 10011 |
| 11.11.11.89  |  9906 |
| 11.11.11.85  |  9896 |
| 11.11.11.84  |  9157 |
| 11.11.11.120 |  8365 |
| 11.11.11.64  |  5759 |
| 11.11.11.65  |   129 |
| 11.11.11.255 |    64 |
+--------------+-------+
26 rows in set (8.65 sec)
```

The open TCP ports on each machine is determined by looking at outgoing packets with the ACK flag set, as follows:

```
mysql> select src, spt, count(*) c from common, tcp where physout='eth0' and
common.log_id=tcp.log_id and flags like '%ACK%' group by src, spt order by src,
spt;
+-------------+------+-----+
| src         | spt  | c   |
+-------------+------+-----+
| 11.11.11.67 |   80 |   1 |
| 11.11.11.67 |  139 |  10 |
| 11.11.11.67 |  443 |   1 |
| 11.11.11.69 |   21 |  30 |
| 11.11.11.69 |   80 |   1 |
| 11.11.11.69 |  443 |   1 |
| 11.11.11.71 |   21 |  30 |
| 11.11.11.71 |   80 | 636 |
| 11.11.11.71 |  443 |   1 |
| 11.11.11.72 |   21 |  30 |
| 11.11.11.72 |   80 |  10 |
| 11.11.11.72 |  443 |   1 |
| 11.11.11.73 |   21 |  26 |
| 11.11.11.73 |   80 |   9 |
| 11.11.11.73 |  139 |   1 |
| 11.11.11.73 |  443 |  29 |
```

```
| 11.11.11.73 | 3128 |   1 |
| 11.11.11.75 |   21 |  28 |
| 11.11.11.75 |   80 |  11 |
| 11.11.11.75 |  443 |  48 |
| 11.11.11.80 |   21 |  26 |
| 11.11.11.80 |   80 | 389 |
| 11.11.11.80 |  139 |   1 |
| 11.11.11.80 |  443 |   1 |
+-------------+------+-----+
24 rows in set (9.25 sec)
```

Hence the system attacked the most was 11.11.11.75 and it had TCP ports 21, 80 and 443 open. To investigate why it had the highest attack rate, some queries are performed as below:

```
mysql> select dpt, count(*) c from tcp, common where common.log_id=tcp.log_id
and dst='11.11.11.75' group by dpt having c >
100 order by c desc;
+------+-------+
| dpt  | c     |
+------+-------+
|  443 | 20436 |
|  135 |  3610 |
|  445 |  2007 |
| 3127 |  1107 |
|  139 |   784 |
| 6129 |   150 |
| 1433 |   129 |
|  901 |   127 |
+------+-------+
8 rows in set (9.66 sec)

mysql> select src, count(*) c from tcp, common where common.log_id=tcp.log_id
and dst='11.11.11.75' and dpt=443 group by src having c > 100 order by c desc;
+----------------+-------+
| src            | c     |
+----------------+-------+
| 66.60.166.84   | 19584 |
| 61.120.200.227 |   481 |
| 218.103.70.82  |   310 |
+----------------+-------+
3 rows in set (1.87 sec)

mysql> select dst, dpt, count(*) from common, tcp where
common.log_id=tcp.log_id and src='66.60.166.84' group by dst, dpt order by dst,
dpt;
+--------------+------+----------+
| dst          | dpt  | count(*) |
+--------------+------+----------+
| 11.11.11.105 |  443 |      311 |
| 11.11.11.69  |  443 |      519 |
| 11.11.11.73  |  443 |        2 |
| 11.11.11.75  |  443 |    19584 |
| 11.11.11.82  |  443 |      478 |
| 11.11.11.87  |  443 |      439 |
| 11.11.11.89  |  443 |      496 |
+--------------+------+----------+
7 rows in set (2.31 sec)
```

As can be seen above, attacks on TCP port 443 accounted for more than 60% of the overall attacks on 11.11.11.75. In the meantime, 66.60.166.84 accounted for more than 90% of the attacks on 443 on that machine. A query on all attacks performed by 66.60.166.84 reveals that it was a dedicated TCP 443 attacker. In particular, the attacker seemed quite interested in 11.11.11.75. A further investigation on the attacker is carried out as follows:

```
mysql> select date(dt), hour(dt), dst, dpt, count(*) from common, tcp where
common.log_id=tcp.log_id and src='66.60.166.84' group by date(dt), hour(dt),
dst, dpt order by date(dt), hour(dt), dst, dpt;
+------------+----------+--------------+------+----------+
| date(dt)   | hour(dt) | dst          | dpt  | count(*) |
+------------+----------+--------------+------+----------+
| 0000-02-07 |       16 | 11.11.11.69  |  443 |      519 |
| 0000-02-07 |       16 | 11.11.11.75  |  443 |     2338 |
| 0000-02-07 |       16 | 11.11.11.87  |  443 |      439 |
| 0000-02-07 |       16 | 11.11.11.89  |  443 |      496 |
| 0000-02-07 |       17 | 11.11.11.75  |  443 |    11418 |
| 0000-02-08 |        3 | 11.11.11.75  |  443 |     4281 |
| 0000-02-08 |        4 | 11.11.11.105 |  443 |      311 |
| 0000-02-08 |        4 | 11.11.11.75  |  443 |      359 |
| 0000-02-08 |        5 | 11.11.11.75  |  443 |      600 |
| 0000-02-08 |        6 | 11.11.11.75  |  443 |      588 |
| 0000-02-08 |        7 | 11.11.11.73  |  443 |        2 |
| 0000-02-08 |        7 | 11.11.11.82  |  443 |      478 |
+------------+----------+--------------+------+----------+
12 rows in set (2.11 sec)
```

One interesting piece of information gained is the working hours of the attacker. As can be seen by the colour partitioning, the attacker attacked the honeynet machines in two different sessions, with about 10 hours of gap between the two sessions. One reasonable guess is that he was sleeping during that period and since the gap started at 5 p.m., it is highly likely that he was from a different timezone. If the guess is right, it can also be deduced that this attacker was quite dedicated in attacking the honeynet machines since the attack pattern (time and number of probes on different machines) reveals that he was manually interacting with the scanner instead of just issuing a scan on a large range of IPs. This should be considered an imminent threat in a production system, especially when 443 is an open port.

To investigate why 11.11.11.75 is so appealing to the attacker, a query is issued:

```
mysql> select src, flags, count(*) from common where dst='66.60.166.84' group
by src, flags order by src, flags;
+-------------+--------+----------+
| src         | flags  | count(*) |
+-------------+--------+----------+
| 11.11.11.69 | DF,ACK |        1 |
| 11.11.11.75 | DF,ACK |       22 |
+-------------+--------+----------+
2 rows in set (3.56 sec)
```

This shows that 11.11.11.75 was one of the two machines which ever responded to the probes of that attacker. Perhaps that was why the attacker returned to probing the machine on Feb 8th. It is however hard to tell why the attacker decided to attempt so many probes on 11.11.11.75 before hour 18 since, as shown below, the ACK packets were only returned starting from hour 18:

```
mysql> select date(dt), hour(dt), src, dst, dpt, count(*) from common, tcp
where common.log_id=tcp.log_id and (src='66.60.166.84' or dst='66.60.166.84')
group by date(dt), hour(dt), dst, dpt order by date(dt), hour(dt), dst, dpt;
+------------+----------+--------------+--------------+-------+----------+
| date(dt)   | hour(dt) | src          | dst          | dpt   | count(*) |
+------------+----------+--------------+--------------+-------+----------+
| 0000-02-07 |       16 | 66.60.166.84 | 11.11.11.69  |   443 |      519 |
| 0000-02-07 |       16 | 66.60.166.84 | 11.11.11.75  |   443 |     2338 |
| 0000-02-07 |       16 | 66.60.166.84 | 11.11.11.87  |   443 |      439 |
| 0000-02-07 |       16 | 66.60.166.84 | 11.11.11.89  |   443 |      496 |
| 0000-02-07 |       17 | 66.60.166.84 | 11.11.11.75  |   443 |    11418 |
| 0000-02-07 |       18 | 11.11.11.75  | 66.60.166.84 | 42580 |        1 |
| 0000-02-07 |       18 | 11.11.11.75  | 66.60.166.84 | 43074 |        1 |
| 0000-02-07 |       18 | 11.11.11.69  | 66.60.166.84 | 54494 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 32831 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 33576 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 34561 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 37330 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 45913 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 46173 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 49428 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 49479 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 50149 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 52051 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 56252 |        1 |
| 0000-02-07 |       19 | 11.11.11.75  | 66.60.166.84 | 59595 |        1 |
| 0000-02-08 |        3 | 66.60.166.84 | 11.11.11.75  |   443 |     4281 |
| 0000-02-08 |        4 | 66.60.166.84 | 11.11.11.105 |   443 |      311 |
| 0000-02-08 |        4 | 66.60.166.84 | 11.11.11.75  |   443 |      359 |
| 0000-02-08 |        5 | 66.60.166.84 | 11.11.11.75  |   443 |      600 |
| 0000-02-08 |        6 | 66.60.166.84 | 11.11.11.75  |   443 |      588 |
| 0000-02-08 |        6 | 11.11.11.75  | 66.60.166.84 | 46935 |        1 |
| 0000-02-08 |        6 | 11.11.11.75  | 66.60.166.84 | 46988 |        1 |
| 0000-02-08 |        6 | 11.11.11.75  | 66.60.166.84 | 47004 |        1 |
| 0000-02-08 |        6 | 11.11.11.75  | 66.60.166.84 | 47006 |        1 |
| 0000-02-08 |        6 | 11.11.11.75  | 66.60.166.84 | 48320 |        1 |
| 0000-02-08 |        6 | 11.11.11.75  | 66.60.166.84 | 51833 |        1 |
| 0000-02-08 |        7 | 66.60.166.84 | 11.11.11.73  |   443 |        2 |
| 0000-02-08 |        7 | 66.60.166.84 | 11.11.11.82  |   443 |      478 |
| 0000-02-08 |        8 | 11.11.11.75  | 66.60.166.84 | 34400 |        1 |
| 0000-02-08 |        8 | 11.11.11.75  | 66.60.166.84 | 39675 |        1 |
+------------+----------+--------------+--------------+-------+----------+
35 rows in set (11.02 sec)
```

10.Provide some high-level metrics about the data (such as most frequently targeted ports, etc) and make some conclusions based on them.

- Total incoming packet count

```
mysql> select count(*) from common where physin='eth0';
+----------+
| count(*) |
+----------+
|   283363 |
+----------+
1 row in set (3.32 sec)
```

- Total number of different attacker IPs seen

```
mysql> select count(distinct src) from common where physin='eth0';
+---------------------+
| count(distinct src) |
+---------------------+
|               20377 |
+---------------------+
1 row in set (9.83 sec)
```

- Attack count on each day

```
mysql> select date(dt), count(*) from common where physin='eth0' group by
date(dt) order by date(dt);
+------------+----------+
| date(dt)   | count(*) |
+------------+----------+
| 0000-02-01 |     7216 |
| 0000-02-02 |     6955 |
| 0000-02-03 |    17816 |
| 0000-02-04 |    10500 |
| 0000-02-05 |     6633 |
| 0000-02-06 |     5342 |
| 0000-02-07 |    20950 |
| 0000-02-08 |    14496 |
| 0000-02-09 |    10493 |
| 0000-02-10 |     4765 |
| 0000-02-11 |    10953 |
| 0000-02-12 |     9761 |
| 0000-02-13 |     9641 |
| 0000-02-14 |     9328 |
| 0000-02-15 |     9731 |
| 0000-02-16 |     9860 |
| 0000-02-17 |     9443 |
| 0000-02-18 |     9559 |
| 0000-02-19 |    11028 |
| 0000-02-20 |     8275 |
| 0000-02-21 |    13662 |
| 0000-02-22 |    11201 |
| 0000-02-23 |     9074 |
| 0000-02-24 |    12403 |
| 0000-02-25 |    10849 |
| 0000-02-26 |    18291 |
| 0000-02-27 |     5138 |
+------------+----------+
27 rows in set (6.18 sec)
```

- Top ten targeted TCP ports

```
mysql> select dpt, count(*) cnt from common, tcp where common.log_id=tcp.log_id
group by dpt order by cnt desc limit 10;
+-------+-------+
| dpt   | cnt   |
+-------+-------+
|   135 | 86634 |
|   445 | 46442 |
|   443 | 26444 |
|  3127 | 25782 |
|   139 | 15000 |
|    80 | 13310 |
|  6129 |  3428 |
|   901 |  3097 |
|  1433 |  2791 |
| 17300 |  2147 |
+-------+-------+
10 rows in set (4.91 sec)
```

- Top ten targeted UDP ports

```
mysql> select dpt, count(*) cnt from common, udp where common.log_id=udp.log_id
group by dpt order by cnt desc limit 10;
+------+-------+
| dpt  | cnt   |
+------+-------+
|   53 | 18127 |
|  137 |  8755 |
| 1434 |  5905 |
|  138 |  3827 |
| 1026 |  2394 |
|  135 |  1525 |
| 1027 |   290 |
|  514 |   266 |
| 1812 |   146 |
|  111 |    28 |
+------+-------+
10 rows in set (2.93 sec)
```

- Top ten attackers

```
mysql> select src, count(*) cnt from common where physin='eth0' group by src
order by cnt desc limit 10;
+----------------+-------+
| src            | cnt   |
+----------------+-------+
| 66.60.166.84   | 21829 |
| 66.186.83.178  | 10217 |
| 63.13.135.27   |  8121 |
| 63.123.70.166  |  7237 |
| 63.125.10.7    |  6882 |
| 127.0.0.1      |  6394 |
| 63.123.38.103  |  3928 |
| 63.126.133.117 |  2801 |
| 63.126.190.227 |  2442 |
| 67.123.234.132 |  2351 |
+----------------+-------+
10 rows in set (30.43 sec)
```

- Top ten aggressive scanners (average packet count per unit time)

```
mysql> select * from ip order by (in_count/(last_seen-first_seen))  desc limit
10;
+----------------+---------------------+---------------------+----------+
| ADDR           | first_seen          | last_seen           | in_count |
+----------------+---------------------+---------------------+----------+
| 67.68.37.182   | 0000-02-22 18:56:45 | 0000-02-22 18:56:46 |       43 |
| 213.37.222.124 | 0000-02-19 06:48:17 | 0000-02-19 06:48:18 |       30 |
| 200.90.105.56  | 0000-02-21 10:54:04 | 0000-02-21 10:54:05 |       29 |
| 158.83.137.173 | 0000-02-12 16:33:32 | 0000-02-12 16:33:34 |       49 |
| 195.8.173.35   | 0000-02-12 03:30:34 | 0000-02-12 03:30:35 |       24 |
| 210.180.220.53 | 0000-02-14 20:31:44 | 0000-02-14 20:31:45 |       24 |
| 210.180.220.62 | 0000-02-16 23:08:39 | 0000-02-16 23:08:40 |       24 |
| 210.92.104.250 | 0000-02-26 00:45:49 | 0000-02-26 00:45:50 |       24 |
| 62.119.134.82  | 0000-02-01 02:38:31 | 0000-02-01 02:38:32 |       24 |
| 66.193.75.57   | 0000-02-23 23:04:22 | 0000-02-23 23:04:23 |       24 |
+----------------+---------------------+---------------------+----------+
10 rows in set (0.43 sec)
```

From the statistics the following can be concluded:

1. There are many attackers out there: more than 20000 different attacker IPs are in the logs, this is a huge number even considering the fact that some of the IPs were spoofed.

2. The attacks never stop: attackers are from around the world and they work round the clock. It goes on forever.

3. Most attacks are targeted at ports specific to Microsoft Windows platforms. Examples are TCP 135, 137-139, 445, 1433 and 3127. Together they accounted for 5 among the top 10 targeted TCP ports.

4. Worms play a significant role in the overall attacking activities. In this particular log file, the activity on TCP port 3127 indicates *MyDoom* activity and it ranks 4th in the most targeted TCP ports.

5. Among all UDP ports, port 53 (DNS) is the most targeted. It needs extra protection (e.g. by filtering source IPs of queries) in production systems.

6. Many opportunistic attackers are out there. They just scan a broad range of IPs and go away if there is no response. They appear and then disappear in the logs in a matter of seconds.

7. The Internet is evil :)

D.Epilogue

The amount of logs involved in this challenge is overwhelming. Even with the help of a database software it is not trivial to gain intelligence from the logs unless you know what you are looking for. Due to the lack of time, some of the answers given above are quite superficial. In many cases there might have been a lot more findings if one can dedicate more time in the analysis. For instance, it will be useful to reverse-resolve the IPs to domain names to better correlate the IPs e.g. analyzing the traffic in terms of domains instead of separate IPs. Something to learn from this challenge is that A. a honeypot is not a fire-and-forget solution to security – gathering the logs is just the beginning of the headache and B. the more you try to find answers from the logs, the more questions you will find.

E.Appendix

A) *gawk* script for converting *iptables* logs to SQL INSERT statements

```
############################
# iptables2sql.awk
# =================
#
# Copyright 2004 Ken Lee
############################

function handle_error(error, line) {
  print error ": " line > "/dev/stderr";
}

function print_stats(values) {
  for (_i in values) {
    print " - " _i " = " values[_i];
  }
  print "************************************";
}

function gen_sql(id, table, schema, values,                              _fields,
_values) {
  _fields = "";
  _values = "";
  for (_i in schema) {
    _fields = _fields "," _i;
    if (schema[_i] == "int") {
      _values = _values "," strtonum(values[_i]);
    }
    else if (schema[_i] == "char") {
      _values = _values ",'" values[_i] "'";
    }
    else if (schema[_i] == "datetime") {
      _values = _values ",STR_TO_DATE('" values[_i] "','%b %e %H:%i:%s')";
    }
    else {
      print "Unknown field type '" _i "'. Aborting" > "/dev/stderr";
      nextfile;
    }
  }
  return "INSERT INTO " table " (LOG_ID" _fields ") VALUES (" id _values ");";
}

BEGIN {
  COMMON_TABLE = "common";
  TCP_TABLE = "tcp";
  UDP_TABLE = "udp";
  ICMP_TABLE = "icmp";

  # Fields common to all protocols
  COMMON_FIELDS["DT"] = "datetime";
  COMMON_FIELDS["IN_INTF"] = "char";
  COMMON_FIELDS["PHYSIN"] = "char";
  COMMON_FIELDS["OUT"] = "char";
  COMMON_FIELDS["PHYSOUT"] = "char";
  COMMON_FIELDS["SRC"] = "char";
  COMMON_FIELDS["DST"] = "char";
  COMMON_FIELDS["LEN"] = "int";
  COMMON_FIELDS["TOS"] = "int";
  COMMON_FIELDS["PREC"] = "int";
  COMMON_FIELDS["TTL"] = "int";
  COMMON_FIELDS["ID"] = "int";
  COMMON_FIELDS["PROTO"] = "char";
  COMMON_FIELDS["FLAGS"] = "char";
  TCP_FIELDS["SPT"] = "int";
  TCP_FIELDS["DPT"] = "int";
  TCP_FIELDS["WINDOW"] = "int";
  TCP_FIELDS["RES"] = "int";
  TCP_FIELDS["URGP"] = "int";
  TCP_FIELDS["LEN"] = "int";
```

```
  UDP_FIELDS["SPT"] = "int";
  UDP_FIELDS["DPT"] = "int";
  UDP_FIELDS["LEN"] = "int";
  ICMP_FIELDS["ID"] = "int";
  ICMP_FIELDS["TYPE"] = "int";
  ICMP_FIELDS["CODE"] = "int";
  ICMP_FIELDS["SEQ"] = "int";

  next_line = 1;
}

{
  _original = $0;
  _pos = 1;

# Date/Time
  values["DT"] = $(_pos++) " " $(_pos++) " " $(_pos++);

# Skip hostname and rule description
  _tmp_counter = 0;
  while (_pos <= NF && _tmp_counter < 2) {
    if (match($_pos, ":$") != 0) {
      _tmp_counter++;
    }
    _pos++;
  }

# Process the rest as either name-value pairs or flags
  flags = "";
  while (_pos <= NF) {
    # A flag
    if (split($_pos, _tmp_arr, "=") < 2) {
      flags = flags (flags == "" ? "" : ",") $_pos;
    }
    # A name-value pair
    else {
      if (_tmp_arr[1] == "IN") {
           _tmp_arr[1] = "IN_INTF";
      }
      values[_tmp_arr[1]] = _tmp_arr[2];
    }
    _pos++;
  }
  values["FLAGS"] = flags;

# Generate SQL statement for specific protocol
  if (values["PROTO"] == "TCP") {
    proto_sql = gen_sql(next_line, TCP_TABLE, TCP_FIELDS, values);
  }
  else if (values["PROTO"] == "UDP") {
    proto_sql = gen_sql(next_line, UDP_TABLE, UDP_FIELDS, values);
  }
  else if (values["PROTO"] == "ICMP") {
    proto_sql = gen_sql(next_line, ICMP_TABLE, ICMP_FIELDS, values);
  }
# IGMP
  else if (values["PROTO"] == "2") {
    proto_sql = "";
# Fix protocol name
    values["PROTO"] = "IGMP";
  }
  else {
    handle_error("Unknown protocol", _original);
    next;
  }
# Generate generic SQL statement for the line
  common_sql = gen_sql(next_line, COMMON_TABLE, COMMON_FIELDS, values);

  print common_sql;
  print proto_sql;
  print "";
  next_line++;
}
```

B) Database schema

- Table <common>

```
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| LOG_ID  | int(11)      |      | PRI | 0       |       |
| DT      | datetime     | YES  |     | NULL    |       |
| IN_INTF | varchar(10)  | YES  |     | NULL    |       |
| PHYSIN  | varchar(10)  | YES  |     | NULL    |       |
| OUT     | varchar(10)  | YES  |     | NULL    |       |
| PHYSOUT | varchar(10)  | YES  |     | NULL    |       |
| SRC     | varchar(20)  | YES  | MUL | NULL    |       |
| DST     | varchar(20)  | YES  |     | NULL    |       |
| LEN     | int(11)      | YES  |     | NULL    |       |
| TOS     | int(11)      | YES  |     | NULL    |       |
| PREC    | int(11)      | YES  |     | NULL    |       |
| TTL     | int(11)      | YES  |     | NULL    |       |
| ID      | int(11)      | YES  |     | NULL    |       |
| PROTO   | varchar(10)  | YES  |     | NULL    |       |
| FLAGS   | varchar(255) | YES  |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
```

- Table <tcp>

```
+--------+---------+------+-----+---------+-------+
| Field  | Type    | Null | Key | Default | Extra |
+--------+---------+------+-----+---------+-------+
| LOG_ID | int(11) |      | PRI | 0       |       |
| SPT    | int(11) | YES  |     | NULL    |       |
| DPT    | int(11) | YES  |     | NULL    |       |
| WINDOW | int(11) | YES  |     | NULL    |       |
| RES    | int(11) | YES  |     | NULL    |       |
| URGP   | int(11) | YES  |     | NULL    |       |
| LEN    | int(11) | YES  |     | NULL    |       |
+--------+---------+------+-----+---------+-------+
```

- Table <udp>

```
+--------+---------+------+-----+---------+-------+
| Field  | Type    | Null | Key | Default | Extra |
+--------+---------+------+-----+---------+-------+
| LOG_ID | int(11) |      | PRI | 0       |       |
| SPT    | int(11) | YES  |     | NULL    |       |
| DPT    | int(11) | YES  |     | NULL    |       |
| LEN    | int(11) | YES  |     | NULL    |       |
+--------+---------+------+-----+---------+-------+
```

- Table <icmp>

```
+--------+---------+------+-----+---------+-------+
| Field  | Type    | Null | Key | Default | Extra |
+--------+---------+------+-----+---------+-------+
| LOG_ID | int(11) |      | PRI | 0       |       |
| ID     | int(11) | YES  |     | NULL    |       |
| TYPE   | int(11) | YES  |     | NULL    |       |
| CODE   | int(11) | YES  |     | NULL    |       |
| SEQ    | int(11) | YES  |     | NULL    |       |
+--------+---------+------+-----+---------+-------+
```