Scan of the month 30#

Submitted by All4Sharon from all4sharon@eyou.com

Prepare Stage:

1. Get the log file and verify it
   a) #wget http://www.honeynet.org/scans/scan30/honeynet-Feb1_FebXX.log.gz
   b) #md5sum honeynet-Feb1_FebXX.log.gz
   e002b1013f18dd42e17be919c2870081 honeynet-Feb1_FebXX.log.gz
   c) #tar zvxf honeynet-Feb1_FebXX.log.gz
   d) #md5sum honeynet-Feb1_FebXX.log
   8c0070ef51f6f764fde0551fa60da11b honeynet-Feb1_FebXX.log
2. Get an overview of the log content
   After skimming through the log file, I found the log can be divided into several parts:
   a) Normal TCP Traffic

```
Feb 1 00:00:02 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0
PHYSOUT=eth1 SRC=192.150.249.87 DST=11.11.11.84 LEN=40 TOS=0x00
PREC=0x00 TTL=110 ID=12973 PROTO=TCP SPT=220 DPT=6129
WINDOW=16384 RES=0x00 SYN URGP=0
```

   b) Normal UDP Traffic

```
Feb 1 09:06:16 bridge kernel: Legal DNS: IN=br0 PHYSIN=eth1 OUT=br0
PHYSOUT=eth0 SRC=11.11.11.67 DST=22.22.22.40 LEN=71 TOS=0x00 PREC=0x00
TTL=64 ID=24007 DF PROTO=UDP SPT=4250 DPT=53 LEN=51
```

   c) Broadcast Traffic

```
Feb 1 00:09:27 bridge kernel: Legal Broadcast: IN=br0 PHYSIN=eth1 OUT=br0
PHYSOUT=eth0 SRC=11.11.11.67 DST=11.11.11.255 LEN=241 TOS=0x00
PREC=0x00 TTL=64 ID=0 DF PROTO=UDP SPT=138 DPT=138 LEN=221
```

   d) Normal ICMP Traffic

```
Feb 1 00:04:19 bridge kernel: INBOUND ICMP: IN=br0 PHYSIN=eth0 OUT=br0
PHYSOUT=eth1 SRC=190.0.0.39 DST=11.11.11.105 LEN=92 TOS=0x00 PREC=0x00
TTL=107 ID=12037 PROTO=ICMP TYPE=8 CODE=0 ID=512 SEQ=73
```

   e) Inblock Message

```
Feb 10 13:50:22 bridge kernel: INBLOCK: IN=br0 PHYSIN=eth0 OUT=br0
PHYSOUT=eth1 SRC=4.22.106.52 DST=11.11.11.105 LEN=92 TOS=0x00
PREC=0x00 TTL=112 ID=17103 PROTO=ICMP TYPE=8 CODE=0 ID=21241
SEQ=46667
```

   f) TCP drop Traffic

```
Feb 1 09:05:50 bridge kernel: Drop TCP after 13 attempts IN=br0 PHYSIN=eth1
OUT=br0 PHYSOUT=eth0 SRC=11.11.11.67 DST=80.116.93.36 LEN=60 TOS=0x00
PREC=0x00 TTL=64 ID=55344 DF PROTO=TCP SPT=1167 DPT=113
```

```
WINDOW=5840 RES=0x00 SYN URGP=0
```

g) UDP drop Traffic

```
Feb  1 21:46:29 bridge kernel: Drop udp after 20 attempts IN=br0 PHYSIN=eth1
OUT=br0 PHYSOUT=eth0 SRC=11.11.11.67 DST=218.38.159.132 LEN=257
TOS=0x00 PREC=0x00 TTL=64 ID=0 DF PROTO=UDP SPT=137 DPT=55247
LEN=237
```

h) MultiCast Traffic

```
Feb  1 21:46:29 bridge kernel: Drop udp after 20 attempts IN=br0 PHYSIN=eth1
OUT=br0 PHYSOUT=eth0 SRC=11.11.11.67 DST=218.38.159.132 LEN=257
TOS=0x00 PREC=0x00 TTL=64 ID=0 DF PROTO=UDP SPT=137 DPT=55247
LEN=237
```

i) Other information

```
Feb 10 13:51:02 bridge kernel: eth0: 0 multicast blocks dropped.
Feb 10 13:51:03 bridge kernel: br0: port 2(eth0) entering disabled state
Feb 10 13:51:03 bridge kernel: eth1: 0 multicast blocks dropped.
Feb 10 13:51:03 bridge kernel: br0: port 1(eth1) entering disabled state
Feb 10 13:51:05 bridge kernel: br0: port 2(eth0) entering listening state
```

3. Use grep to separate the log and change the log into database style with some Perl scripts e.g.

1) Get TCP traffic usinig grep

```
#grep –i “bridge kernel: INBOUND TCP:” | grep –i “PROTO=TCP > tcp.log
```

2) Create MySQL table tblTCPTraffic

```
use sotm;
create table tblTCPTraffic(
        time datetime,
        direction varchar(1),
        datain int,
        dataout int,
        phyin int,
        phyout    int,
        srcip varchar(15),
        dstip varchar(15),
        sport int,
        dport int,
        len int,
        tos int,
        prec int,
        ttl int,
        id int,
        df int,
        window int,
        res int,
        flags int,
        urgp int,
);
```

3) Put the traffic into database using Perl script

```
#!/usr/bin/perl
use strict;
use DBI;

my $tcpfile = "tcp.log";
my $dbh;
open TCPF, $tcpfile || die "Can't open TCP file.\n";
$dbh = DBI->connect('DBI:mysql:sotm','root',",) || die "Can't open DB\n";

my $line;
while(<TCPF>){
if(/...( +)(\d+) (\d\d:\d\d:\d\d) bridge kernel: INBOUND TCP: IN=br(\d) PHYSIN=eth(\d)
OUT=br(\d) PHYSOUT=eth(\d) SRC=(\d+\.\d+\.\d+\.\d+) DST=(\d+\.\d+\.\d+\.\d+)
LEN=(\d+) TOS=(0x..) PREC=(0x..) TTL=(\d+) ID=(\d+) (|DF )PROTO=TCP SPT=(\d+)
DPT=(\d+) WINDOW=(\d+) RES=(0x..)
(|CWR )(|ECE )(|ACK )(|RST )(|SYN )(|PSH )(|FIN )URGP=(\d+)/){
        my $date = "2004-2-" . $2 . " " . $3;
        my $datain = $4;
        my $phyin = $5;
        my $dataout = $6;
        my $phyout = $7;
        my $srcip = $8;
        my $dstip = $9;
        my $len = $10;
        my $tos = $11;
        my $prec = $12;
        my $ttl = $13;
        my $id = $14;
        my $df;
        if($15 eq undef){
            $df = 0;
        }else{
            $df = 1;
        }
        my $sport = $16;
        my $dport = $17;
        my $window = $18;
        my $res = $19;
        my $flags;
        $flags |= 0x80 if($20 ne undef); #CWR Flag
        $flags |= 0x40 if($21 ne undef); #ECE Flag
        $flags |= 0x10 if($22 ne undef); #ACK Flag
        $flags |= 0x04 if($23 ne undef); #RST Flag
        $flags |= 0x02 if($24 ne undef); #SYN Flag
        $flags |= 0x08 if($25 ne undef); #PSH Flag
        $flags |= 0x01 if($26 ne undef); #FIN Flag
        my $urgp = $27;
        $dbh->do("INSERT INTO tblTCPTraffic(
                time, direction, datain, dataout, phyin,
                phyout, srcip, dstip, sport, dport, len,
                tos, prec, ttl, id, df, window, res, flags,
                urgp,)
        VALUES( '$date', 'I', $datain, $dataout, $phyin,
                $phyout, '$srcip', '$dstip', $sport,
                $dport, $len, $tos, $prec, $ttl, $id,
                $df, $window, $res, $flags, $urgp )" );
 }else{
        print "Unmatch Entry: $_";
 }
}
$dbh->disconnect();
```

With the same procedure, I create other tables such as tblUDPTraffic, tblICMPTraffic, tblBroadcast, tblAllTraffic.

Analysis Stage:

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?

The first thing is to find the honeynet hosts and DNS hosts.

```
SELECT DISTINCT srcip FROM tblAllTraffic WHERE dstip LIKE “11.11.11.%” ORDER BY dstip;
```

```
| 11.11.11.100 |
| 11.11.11.105 |
| 11.11.11.110 |
| 11.11.11.115 |
| 11.11.11.120 |
| 11.11.11.125 |
| 11.11.11.64    |
| 11.11.11.67    |
| 11.11.11.69    |
| 11.11.11.70    |
| 11.11.11.71    |
| 11.11.11.72    |
| 11.11.11.73    |
| 11.11.11.75    |
| 11.11.11.80    |
| 11.11.11.81    |
| 11.11.11.82    |
| 11.11.11.83    |
| 11.11.11.84    |
| 11.11.11.85    |
| 11.11.11.87    |
| 11.11.11.89    |
| 11.11.11.90    |
| 11.11.11.95    |
```

Notice: **The IPs above may contain spoofed IPs**

SELECT DISTINCT dstip FROM tblAllTraffic WHERE dstip LIKE “22.22.22.%”;

```
| 22.22.22.40    |
```

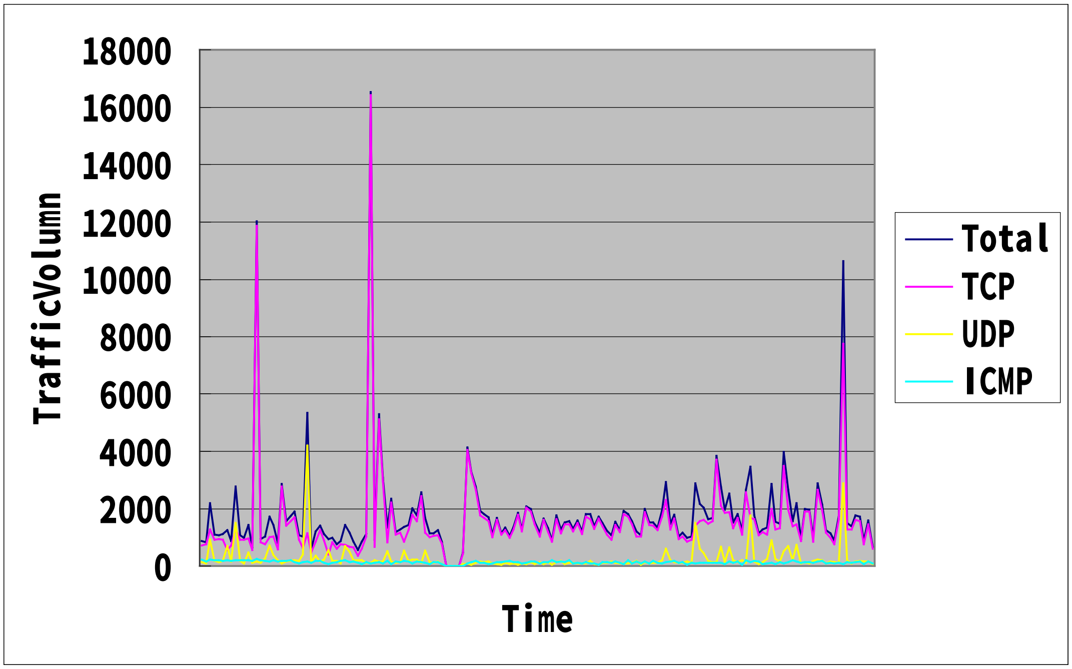
SELECT DISTINCT dstip FROM tblAllTraffic WHERE dstip LIKE “23.23.23.%”;

```
| 23.23.23.60    |
```

Use the following script to analyze the traffic

```
#!/usr/bin/perl

use strict;
use DBI;

my $dbh;
my $sth;
my $basetime = '2004-02-01 00:00:00';

$dbh = DBI->connect("DBI:mysql:sotm","root","",);

#-----------------Analyzing Total Traffic---------------------#
print "Total Traffic per 4 hours.";
for(my $count=0; $count < 6 * 26; $count++){
   $sth = $dbh->prepare("SELECT count(*) FROM tblAllTraffic where "
                . "time between
                date_add('$basetime', interval (4*$count) hour)
                and
                date_add('$basetime', interval (4*($count+1)) hour) ");
   $sth->execute();
   my $num = $sth->fetchrow_array();
   print "$num\n";
}
#----------------Analyziing TCP Traffic----------------------#
print "TCP Traffic per 4 hours.";
for(my $count=0; $count < 6 * 26; $count++){
   $sth = $dbh->prepare("SELECT count(*) FROM tblAllTraffic where "
                . "time between
                date_add('$basetime', interval (4*$count) hour)
                and
                date_add('$basetime', interval (4*($count+1)) hour)"
                . " and proto=1 ");
    $sth->execute();
    my $num = $sth->fetchrow_array();
    print "$num\n";
}
#----------------Analyziing UDP Traffic-----------------------#
print "UDP Traffic per 4 hours.";
for(my $count=0; $count < 6 * 26; $count++){
    $sth = $dbh->prepare("SELECT count(*) FROM tblAllTraffic where "
                . "time between
                date_add('$basetime', interval (4*$count) hour)
                and
                date_add('$basetime', interval (4*($count+1)) hour)"
                . " and proto=0 ");
    $sth->execute();
    my $num = $sth->fetchrow_array();
    print "$num\n";
}
#----------------Analyziing ICMP Traffic-----------------------#
print "ICMP Traffic per 4 hours.";
for(my $count=0; $count < 6 * 26; $count++){
    $sth = $dbh->prepare("SELECT count(*) FROM tblICMPTraffic where "
                . "time between
                date_add('$basetime', interval (4*$count) hour)
                and
                date_add('$basetime', interval (4*($count+1)) hour)");
    $sth->execute();
    my $num = $sth->fetchrow_array();
    print "$num\n";
}
$sth->finish();
$dbh->disconnect();
```

Then make a traffic graph as below.



From the graph we can see three TCP peaks at

Feb 03 4:00AM – 8:00AM,
Feb 07 16:00PM – 20:00 PM,
Feb 26 8:00AM – 12:00AM

and two UDP peaks at

Feb 05 4:00AM – 8:00Am
Feb 26 8:00AM – 12:00AM

ICMP is quiet smooth , no significant huge traffic.

Notice there is no traffic between Feb 09 16:00PM – Feb 10 08:00 AM

Then have a look at TOP 30 source hosts, TOP 30 destination hosts, and TOP 30 frequently connected ports.

TOP 30 source hosts

```
select count(srcip) as mycount, srcip from tblAllTraffic group by srcip order by mycount desc limit 30;
```

```
+-----------+---------------------+
| mycount | srcip                 |
+-----------+---------------------+
|     22754 | 11.11.11.67         |
|     21829 | 66.60.166.84        |
|     10197 | 66.186.83.178       |
|       8121 | 63.13.135.27       |
|       6394 | 127.0.0.1          |
|       4018 | 63.123.70.166      |
|       3794 | 63.125.10.7        |
|       2801 | 63.126.133.117     |
|      2334 | 67.123.234.132      |
|      2087 | 63.126.133.8        |
|      1960 | 194.128.177.225     |
|      1809 | 69.55.143.53        |
|      1805 | 61.48.11.170        |
|      1602 | 64.156.39.12        |
|      1592 | 218.103.70.82       |
|      1584 | 68.237.49.113       |
|      1411 | 63.126.190.227      |
|      1296 | 83.30.20.8          |
|      1192 | 208.61.160.83       |
|      1025 | 63.126.133.131      |
|      1013 | 217.81.50.100       |
|       888 | 61.120.200.227      |
|       779 | 63.126.133.122      |
|       755 | 63.122.75.30        |
|       712 | 83.26.23.92         |
|       678 | 200.203.174.67      |
|       676 | 11.11.11.71         |
|       630 | 63.126.25.230       |
+-----------+---------------------+
```

TOP 30 destination hosts

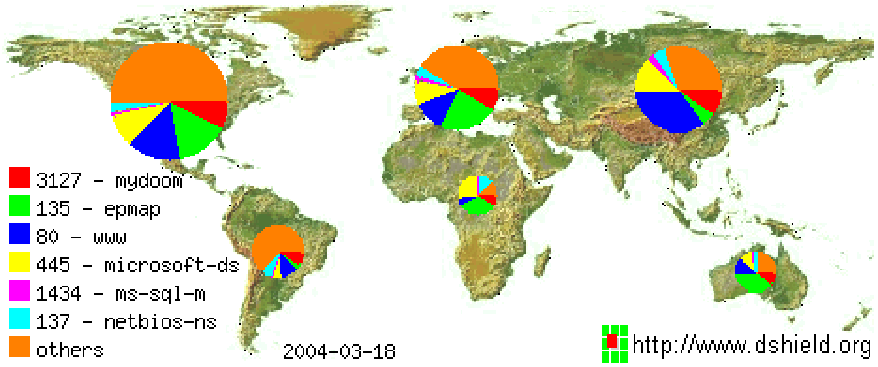
```
select count(dstip) as mycount, dstip from tblAllTraffic group by dstip order by mycount desc limit 30;
```

```
+-----------+---------------------+
| mycount   | dstip               |
+-----------+---------------------+
|     30130 | 11.11.11.75         |
|     13255 | 11.11.11.80         |
|     12381 | 11.11.11.67         |
|     11417 | 11.11.11.100        |
|     11359 | 11.11.11.90         |
|     11062 | 11.11.11.71         |
|     10994 | 11.11.11.87         |
|     10915 | 11.11.11.105        |
|     10842 | 11.11.11.115        |
|     10839 | 11.11.11.110        |
|     10689 | 11.11.11.70         |
|     10510 | 11.11.11.69         |
|     10322 | 11.11.11.73         |
|     10178 | 11.11.11.72         |
|     10062 | 11.11.11.82         |
|     10027 | 11.11.11.125        |
|      9805 | 11.11.11.81         |
|      9417 | 11.11.11.83         |
|      9324 | 11.11.11.89         |
|      9306 | 11.11.11.85         |
|      9109 | 23.23.23.60         |
|      8998 | 22.22.22.40         |
|      8939 | 11.11.11.95         |
|      8573 | 11.11.11.84         |
|      8122 | 11.11.11.120        |
|      5072 | 11.11.11.64         |
|      3792 | 11.11.11.255        |
|       135 | 11.11.11.65         |
|        61 | 195.36.244.104      |
+-----------+---------------------+
```

TOP 30 frequently connected ports

```
select count(*) as mycount , dport from tblAllTraffic group by dport order by mycount desc limit 30;
```

```
+-----------+---------+
| mycount   | dport   |
+-----------+---------+
|     88157 |     135 |
|     46439 |     445 |
|     26444 |     443 |
|     25781 |    3127 |
|     18156 |      53 |
|     15000 |     139 |
|     13309 |      80 |
|      8742 |     137 |
|      5909 |    1434 |
|      3773 |     138 |
|      3426 |    6129 |
|      3097 |     901 |
|      2791 |    1433 |
|      2401 |    1026 |
|      2147 |   17300 |
|      1932 |    1080 |
|      1539 |    3128 |
|      1412 |    4899 |
|      1107 |      21 |
|       557 |   10080 |
|       536 |   27374 |
|       421 |     113 |
|       413 |      23 |
|       381 |     111 |
|       378 |   20168 |
|        304 |    1038 |
|        294 |    1027 |
|        292 |      25 |
|        282 |     389 |
|        249 | 45295 |
+-----------+---------+
```



According to the data from Dshield, we can see a general match between our log analysis and the chart above : apart from the usual port traffic such as 53, 135, 137, 443, 445, some unusual ports also have huge traffic- port 3127, 6129, 901

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

Since all traffic related to Honeynet is suspicious, with the huge traffic towards 135,445,443,3127, 139, we can go to Symantec to find the answer:

1) **W32.Mydoom**

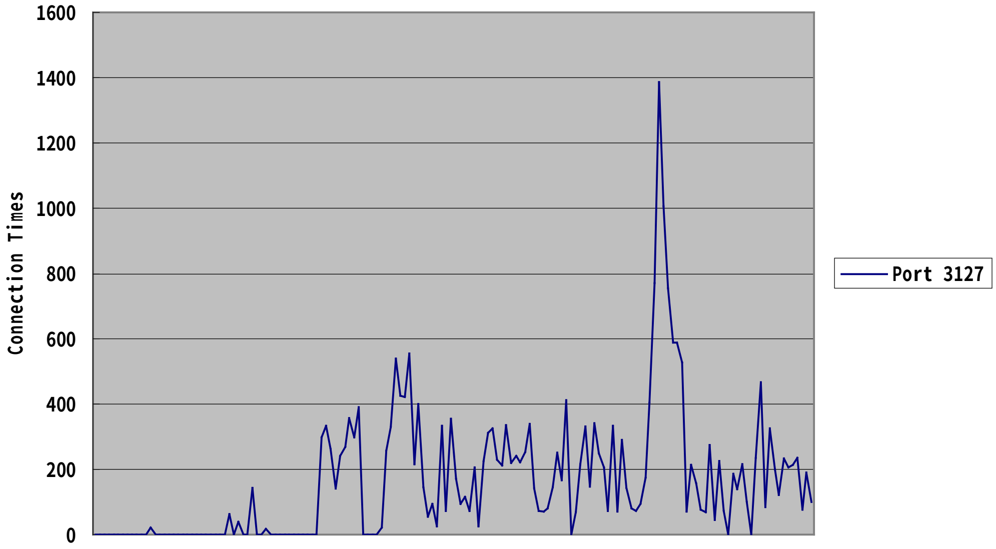
W32.Mydoom.A@mm (also known as W32.Novarg.A) is a mass-mailing worm that arrives as an attachment with the file extension .bat, .cmd, .exe, .pif, .scr, or .zip.

When a computer is infected, **the worm sets up a backdoor into the system by opening TCP ports 3127 through 3198**, which can potentially allow an attacker to connect to the computer and use it as a proxy to gain access to its network resources.

In addition, the backdoor can download and execute arbitrary files.

There is a 25% chance that a computer infected by the worm will perform a Denial of Service (DoS) on February 1, 2004 starting at 16:09:18 UTC, which is also the same as 08:09:18 PST, based on the machine's local system date/time. If the worm does start the DoS attack, it will not mass mail itself. It also has a trigger date to stop spreading/DoS-attacking on February 12, 2004. While the worm will stop on February 12, 2004, the backdoor component will continue to function after this date.

Using the same Perl script as in Question 1, we can see the trend of this worm detection



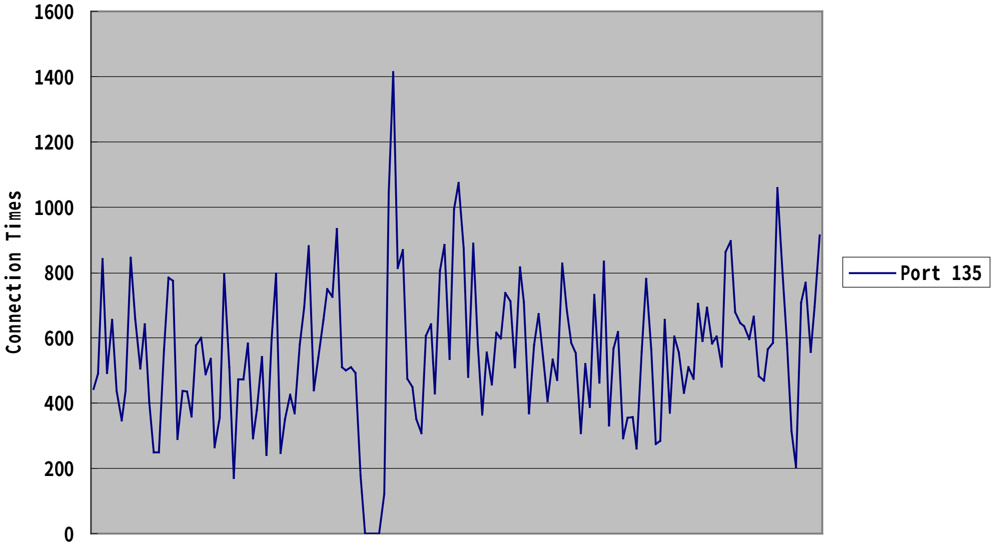
2) **W32.Welchia.B.Worm**

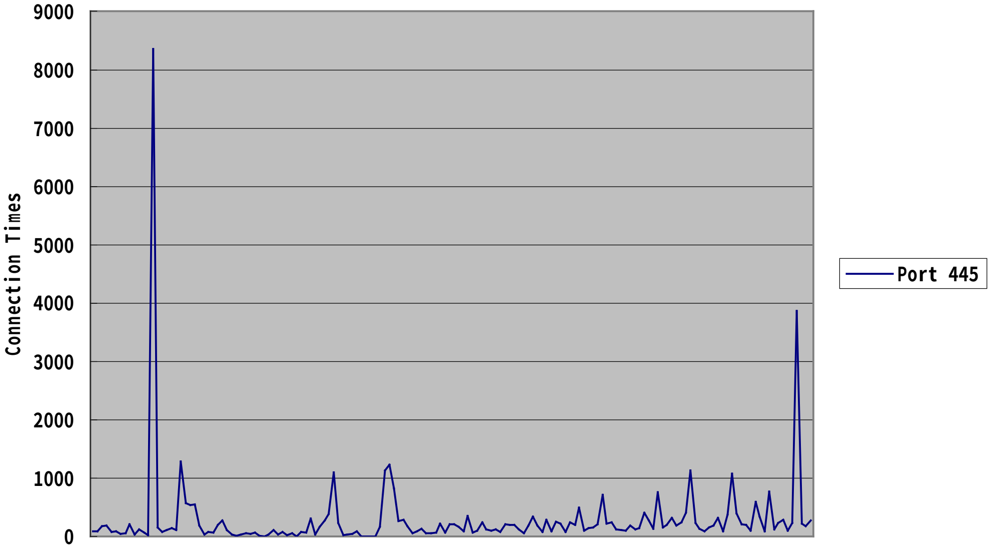
The DCOM RPC vulnerability (first described in Microsoft Security Bulletin MS03-026) **using TCP port 135.** The worm specifically targets Windows XP machines using this exploit. We recommend that you patch this vulnerability by applying Microsoft Security Bulletin MS03-03.

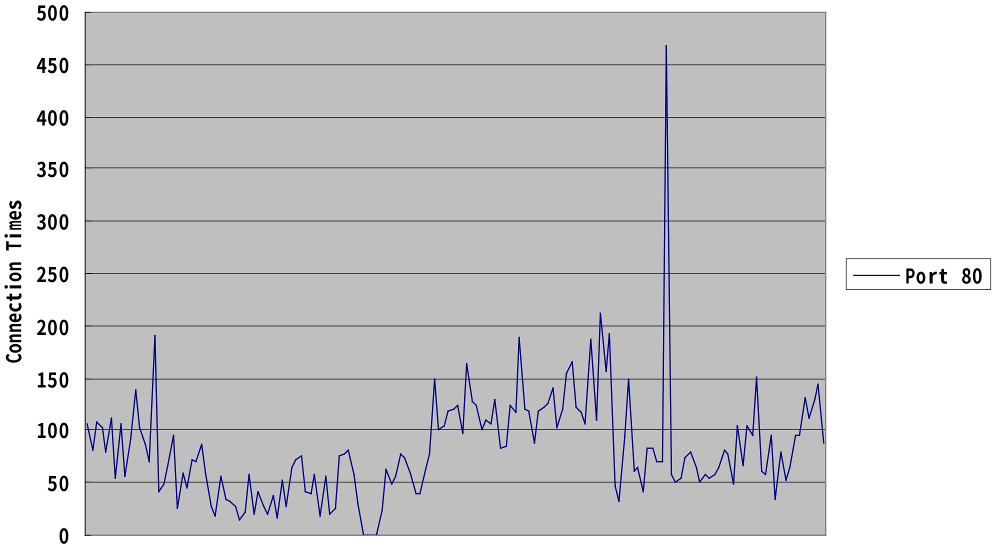
The WebDav vulnerability (described in Microsoft Security Bulletin MS03-007) **using TCP port 80.** The worm specifically targets machines running Microsoft IIS 5.0 using this exploit. The worm's use of this exploit will impact Windows 2000 systems and may impact Windows NT/XP systems.

The Workstation service buffer overrun vulnerability (described in Microsoft Security Bulletin MS03-049) **using TCP port 445**. Windows XP users are protected against this vulnerability if Microsoft Security Bulletin MS03-043 has been applied. Windows 2000 users must apply MS03-049.

The Locator service vulnerability **using TCP port 445** (described in Microsoft Security Bulletin MS03-001). The worm specifically targets Windows 2000 machines using this exploit.


1600
1400
1200
1000
800
600
400
200
0
Connection Times
Port 135


9000
8000
7000
6000
5000
4000
3000
2000
1000
0
Connection Times
Port 445



3) **W32.SQLExp.Worm**

W32.SQLExp.Worm is a worm that targets the systems running Microsoft SQL Server 2000, as well as Microsoft Desktop Engine (MSDE) 2000. The worm sends 376 bytes to **UDP port 1434**, the SQL Server Resolution Service Port.

The worm has the unintended payload of performing a Denial of Service attack due to the large number of packets it sends.

Since more and more viruses are spread like worms, there will be no exact discriminations between viruses and worms.

4) **Others**

From the logs, there are a lot of connections indicating backdoors and Trojans, e.g. the port 901 traffic. Although 901 is the swat service of Samba Web Administration Tool, maybe it’s actually a scan for the trojan "Net-Devil", not sure about it. Also port 17300, is being used by the Milkit Trojan.

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?

1) As mentioned in Question 2, the MyDoom worm opens port 3127, which can potentially allow an attacker to connect to the computer and use it as a proxy to gain access to its network resources. So the 3127 connections indicates the hunting for worm infected hosts.

```
SELECT COUNT(*) AS mycount, srcip FROM tblAllTraffic WHERE srcip NOT
LIKE "11.11.11.%" AND dport=3127 GROUP BY srcip ORDER BY mycount;
+-----------+---------------------+
| mycount | srcip                 |
+-----------+---------------------+
|           1 | 80.144.160.247    |
|           1 | 217.56.225.147    |
…………
```

```
…………
|        86 | 216.207.219.226 |
|        88 | 81.128.73.136   |
|        88 | 203.116.130.28  |
|        88 | 68.196.177.236  |
|        92 | 80.11.132.160   |
|        95 | 200.54.192.190  |
|        96 | 209.71.67.100   |
|       136 | 200.93.66.80    |
|       142 | 67.73.34.191    |
+-----------+-----------------------+
484 rows in set (0.63 sec)
```

We can see there are 484 hosts trying to hunt for the victims of MyDoom. Among them, 67.73.34.191, 200.93.66.80, 209.71.67.100, 200.54.192.190, 80.11.132.160 are the TOP 5.

2) Port 135, 445, 139 are the most open ports on WinBox, so they became the targets of many hackers. Analyzed with the same process as above, we can see

```
select count(*) as mycount, srcip from tblAllTraffic where srcip not like "11.11.11.%"
and dport=135 group by srcip order by mycount;
+-----------+----------------------+
| mycount | srcip                |
+-----------+----------------------+
|         1 | 63.149.20.153   |
|         1 | 203.164.84.117  |
……………
……………
|       513 | 63.126.133.131  |
|       536 | 63.124.18.172   |
|       591 | 63.124.18.175   |
|       755 | 63.122.75.30    |
|       809 | 64.156.39.12    |
|      1065 | 63.126.133.8    |
|      1411 | 63.126.190.227  |
|      1418 | 63.126.133.117  |
|      3794 | 63.125.10.7     |
|      4018 | 63.123.70.166   |
+-----------+-----------------------+
3644 rows in set (0.78 sec)
```

Among the 3644 hosts, maybe most of them are affected by worms and are forced to find other victims, but there are some intended hosts who make the active detection. 63.123.70.166, 63.125.10.7, 63.126.133.117, 63.126.190.227, 63.126.133.8 are TOP 5, all in the subnet of 63.122-126.X.Y, from UUNET,

```
OrgName:      UUNET Technologies, Inc.
OrgID:        UU
```

```
Address:       22001 Loudoun County Parkway
City:          Ashburn
StateProv:     VA
PostalCode:    20147
Country:       US

NetRange:      63.64.0.0 - 63.127.255.255
CIDR:          63.64.0.0/10
NetName:       UUNET63
NetHandle:     NET-63-64-0-0-1
Parent:        NET-63-0-0-0-0
NetType:       Direct Allocation
NameServer:    AUTH03.NS.UU.NET
NameServer:    AUTH00.NS.UU.NET
Comment:       ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:       1999-01-22
Updated:       2003-01-23
```

3) Port 443 traffic is the third, during the time of Feb, because of the ASN.1 vulnerability. Since only firewall log is available, we can't make further analysis.

```
select count(*) as mycount, srcip from tblAllTraffic where srcip not like "11.11.11.%"
and dport=443 group by srcip order by mycount;
+-----------+---------------------+
| mycount | srcip                 |
+-----------+---------------------+
|         1 | 203.194.147.88      |
|         1 | 216.65.31.201       |
|         5 | 211.90.223.186      |
……………
……………
|       602 | 200.203.174.125     |
|       678 | 200.203.174.67      |
|       872 | 61.120.200.227      |
|      1569 | 218.103.70.82       |
|     21829 | 66.60.166.84        |
+-----------+---------------------+
28 rows in set (0.71 sec)
```

Among the 28 hosts, 66.60.166.84 takes the most part. Terrible! Is this host mad? :-)

4) The next is port 6129, used by the Dameware remote administration software. See http://www.linklogger.com/TCP6129.htm for details. A remote unauthenticated attacker can execute arbitrary code on the system

```
select count(*) as mycount, srcip from tblAllTraffic where srcip not like
"11.11.11.%" and dport=6129 group by srcip order by mycount;
+-----------+---------------------+
| mycount | srcip                 |
+-----------+---------------------+
|         1 | 65.174.216.92       |
|         3 | 211.218.212.172 |
……………
……………
|        55 | 24.17.237.70        |
|        72 | 68.59.101.217       |
|        87 | 195.167.19.135      |
|        94 | 210.105.40.67       |
+-----------+---------------------+
115 rows in set (0.61 sec)
```

5) Other reconnaissances do exist, such as port 17300 for the Milkit Trojan, and so on.

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?

Sequential scan:

```
mysql> select time, srcip, dstip, dport from tblAllTraffic where dport=21 order by srcip;
+-------------------------+---------------------+---------------------+---------+
| time                    | srcip               | dstip               | dport   |
+-------------------------+---------------------+---------------------+---------+
……………
| 2004-02-18 21:28:41 | 193.253.34.98     | 11.11.11.64       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.67       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.69       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.70       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.71       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.72       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.73       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.75       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.80       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.81       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.82       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.83       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.84       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.85       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.87       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.89       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.90       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.95       |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.100      |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.105      |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.110      |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.115      |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.120      |      21 |
| 2004-02-18 21:28:42 | 193.253.34.98     | 11.11.11.125      |      21 |
……………
+---------------------+-------------------+-------------------+---------+
```

Random Scan

```
+---------------------+-------------------+-------------------+---------+
| time                | srcip             | dstip             | dport   |
+---------------------+-------------------+-------------------+---------+
……………
| 2004-02-22 19:27:34 | 128.164.136.46    | 11.11.11.67       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.75       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.70       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.82       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.84       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.87       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.64       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.95       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.100      |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.69       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.72       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.80       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.71       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.73       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.81       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.83       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.90       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.85       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.120      |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.89       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.125      |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.105      |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.110      |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46    | 11.11.11.115      |      21 |
……………
+---------------------+-------------------+-------------------+---------+
```

Normal tools such as nmap use sequential scan, although nmap can use parallelism, it seems to handle simultaneous port connections not host connections. There are also other tools, e.g. Nessus, can perform parallel scans towards multiple hosts. There is not enough information about the tools they used.

Long term scan

This question seems a little challenging, so I make a Perl script to handle it.

1) First make a temporary table containing the source IPs whose scan count is between 20 and 100

```
Mysql>CREATE TABLE tblsrcip SELECT COUNT(*) AS mycount, srcip FROM tblAllTraffic group by srcip order by mycount;
```

2) Run the following Perl script

```
#!/usr/bin/perl

use strict;
use DBI;

my $dbh = DBI->connect("DBI:mysql:sotm", "root", "");
my $sth;
my $sth2;

$sth = $dbh->prepare("select srcip from tblsrcip where mycount < 100 "
            . " and mycount > 20");
$sth->execute();
while(my $srcip = $sth->fetchrow_array()){
    $sth2 = $dbh->prepare("select time from tblAllTraffic where srcip = "
                . " '$srcip' order by time limit 1 ");
    $sth2->execute();
    my $starttime = $sth2->fetchrow_array();
    $sth2 = $dbh->prepare("select count(*) from tblAllTraffic where srcip="
                . " '$srcip' and time > "
                . " date_add('$starttime', interval 1 minute) "
                . " order by time ");
    $sth2->execute();
    if(my $count = $sth2->fetchrow_array()){
        print "$srcip seems interesting .\n";
        print "The following are the details.\n";
        $sth2 = $dbh->prepare("select * from tblAllTraffic where srcip="
                . " '$srcip'");
        $sth2->execute();
        while(my @array = $sth2->fetchrow_array()){
            print join("\t", @array), "\n";
        }
        <STDIN>;
    }else{
        print "$srcip scans too fast .\n";
    }
}
$sth2->finish();
$sth->finish();
$dbh->disconnect();
```

From the output, we got:

……

……

63.208.193.241 seems interesting .

The following are the details.

```
2004-02-03 02:20:24    63.208.193.241    11.11.11.64    3127    3127    1
2004-02-03 02:21:46    63.208.193.241    11.11.11.67    3127    3127    1
2004-02-03 02:22:41    63.208.193.241    11.11.11.69    3127    3127    1
2004-02-03 02:23:09    63.208.193.241    11.11.11.70    3127    3127    1
2004-02-03 02:23:36    63.208.193.241    11.11.11.71    3127    3127    1
2004-02-03 02:24:03    63.208.193.241    11.11.11.72    3127    3127    1
2004-02-03 02:24:31    63.208.193.241    11.11.11.73    3127    3127    1
2004-02-03 02:25:25    63.208.193.241    11.11.11.75    3127    3127    1
2004-02-03 02:27:42    63.208.193.241    11.11.11.80    3127    3127    1
2004-02-03 02:28:10    63.208.193.241    11.11.11.81    3127    3127    1
2004-02-03 02:28:37    63.208.193.241    11.11.11.82    3127    3127    1
2004-02-03 02:29:05    63.208.193.241    11.11.11.83    3127    3127    1
2004-02-03 02:29:32    63.208.193.241    11.11.11.84    3127    3127    1
2004-02-03 02:30:00    63.208.193.241    11.11.11.85    3127    3127    1
2004-02-03 02:34:34    63.208.193.241    11.11.11.95    3127    3127    1
2004-02-03 02:36:51    63.208.193.241    11.11.11.100   3127    3127    1
2004-02-03 02:39:08    63.208.193.241    11.11.11.105   3127    3127    1
2004-02-03 02:41:25    63.208.193.241    11.11.11.110   3127    3127    1
2004-02-03 02:43:42    63.208.193.241    11.11.11.115   3127    3127    1
2004-02-03 02:45:59    63.208.193.241    11.11.11.120   3127    3127    1
2004-02-03 02:48:16    63.208.193.241    11.11.11.125   3127    3127    1
……
```

See? 63.208.193.241 spent almost 30 mins to scan the honeynet trying to find the victims of MyDoom. Got YOU!! :-)

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

Besides worms, virus, scans, anything left?   Maybe exploits?

There is only firewall log traffic available, no network application payload, but we can infer if a hacker found something vulnerable, he would try some exploit codes. It could be a success or a failure depends on the exploit codes.

Using the example mentioned in Question 4

```
| 2004-02-22 19:27:36 | 128.164.136.46   | 11.11.11.90       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46   | 11.11.11.85       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46   | 11.11.11.120      |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46   | 11.11.11.89       |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46   | 11.11.11.125      |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46   | 11.11.11.105      |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46   | 11.11.11.110      |      21 |
| 2004-02-22 19:27:36 | 128.164.136.46   | 11.11.11.115      |      21 |
```

After the random scanning,

```
| 2004-02-22 19:30:05 | 128.164.136.46   | 11.11.11.71       |      21 |
```

Maybe 128.164.136.46 found a FTP vulnerability in 11.11.11.71 and want to try some exploit code ?

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

Generally speaking, since the honeynet is 11.11.11.X, there shouldn’t be any reserved subnet traffic such as 192.168.X.Y and local loop - 127.X.Y.Z , but

```
mysql> select count(*) from tblAllTraffic where srcip like "127.%";
    +------------+
    | count(*)   |
    +------------+
    |       6394 |
    +------------+
    1 row in set (1.25 sec)

    mysql> select count(*) from tblAllTraffic where srcip like "192.168.%";
    +------------+
    | count(*)   |
    +------------+
    |       1274 |
    +------------+
    1 row in set (0.50 sec)

    mysql> select count(*) from tblAllTraffic where srcip like "172.16.%";
    +------------+
    | count(*)   |
    +------------+
    |          5 |
    +------------+
    1 row in set (0.53 sec)

    mysql> select count(*) from tblAllTraffic where srcip like "255.255.255.255";
    +------------+
    | count(*)   |
    +------------+
    |          0 |
    +------------+
    1 row in set (0.49 sec)

    mysql> select count(*) from tblAllTraffic where srcip like "10.%";
    +------------+
    | count(*)   |
    +------------+
    |         71 |
    +------------+
    1 row in set (0.54 sec)
```

```
Feb  3 13:43:51 bridge kernel: OUTG CONN OTHER: IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth0 SRC=11.11.11.67 DST=224.0.0.2 LEN=32 TOS=0x00 PREC=0x00 TTL=1 ID=0 DF PROTO=2
```

Because most routers and perimeter firewalls block such unidentified/anomalous traffic, so the traffic above is most likely originated from the LAN. One reasonable answer : some honeynet host is hacked and used to gather information. Although using spoofed packet won't get correct response from the target host, hackers can conceal their detection by making a lot of trash logs. Also idle scan can use spoofed packet to gather information.

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

All outbound traffic should be suspicious since a normal host in the honeynet shouldn't make any active connections to the Internet. But there are exceptions, e.g. a FTP server can use Ident to authenticate ftp clients. But hackers can use this exception, make a ftp server listening on port 113, and the firewall think you're connecting the ident service outside .

1) See the following example:

```
Feb  1 09:05:50 bridge kernel: Drop TCP after 13 attempts IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth0 SRC=11.11.11.67 DST=80.116.93.36 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=55344 DF PROTO=TCP SPT=1167 DPT=113 WINDOW=5840 RES=0x00 SYN URGP=0
```

```
mysql> select * from tblAllTraffic where srcip = "80.116.93.36" and time <= "2004-02-01 09:05:50" order by time desc limit 10;
+-------------------------+---------------+----------------+---------+---------+---------+
| time                    | srcip         | dstip          | sport   | dport   | proto   |
+------------------------+----------------+----------------+---------+---------+---------+
| 2004-02-01 09:05:47 | 80.116.93.36 | 11.11.11.72    |    3349 |      21 |       1 |
| 2004-02-01 09:05:47 | 80.116.93.36 | 11.11.11.125 |    3400 |      21 |       1 |
| 2004-02-01 09:05:47 | 80.116.93.36 | 11.11.11.87    |    3406 |      21 |       1 |
| 2004-02-01 09:05:47 | 80.116.93.36 | 11.11.11.89    |    3470 |      21 |       1 |
| 2004-02-01 09:05:47 | 80.116.93.36 | 11.11.11.90    |    3486 |      21 |       1 |
| 2004-02-01 09:05:46 | 80.116.93.36 | 11.11.11.69    |    3203 |      21 |       1 |
| 2004-02-01 09:05:46 | 80.116.93.36 | 11.11.11.70    |    3290 |      21 |       1 |
| 2004-02-01 09:05:46 | 80.116.93.36 | 11.11.11.71    |    3301 |      21 |       1 |
| 2004-02-01 09:05:45 | 80.116.93.36 | 11.11.11.67    |    3168 |      21 |       1 |
| 2004-02-01 09:05:36 | 80.116.93.36 | 11.11.11.70    | 25780 |      21 |       1 |
+------------------------+-----------------+----------------+---------+---------+--------+
10 rows in set (0.55 sec)
```

So this time the outbound connection doesn't mean the honeynet host is compromised

2) The next example is

```
Feb  8 11:49:57 bridge kernel: Drop TCP after 13 attempts IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth0 SRC=11.11.11.67 DST=207.66.155.21 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=24147 DF PROTO=TCP SPT=1765 DPT=80 WINDOW=5840 RES=0x00 SYN URGP=0
```

```
mysql> select * from tblAllTraffic where srcip = "207.66.155.21" and time <= "2004-02-08 11:49:57" order by time desc limit 10;
```

```
Empty set (0.56 sec)
```

No previous Conection !!!

```
mysql> select * from tblAllTraffic where dstip="207.66.155.21" and srcip like "11.11.11.%";
+---------------------+-------------+---------------+-------+-------+-------+
| time                | srcip       | dstip         | sport | dport | proto |
+---------------------+-------------+---------------+-------+-------+-------+
| 2004-02-08 07:34:30 | 11.11.11.67 | 207.66.155.21 |  1742 |    80 |     1 |
| 2004-02-08 10:56:35 | 11.11.11.67 | 207.66.155.21 |  1759 |    80 |     1 |
| 2004-02-08 10:56:38 | 11.11.11.67 | 207.66.155.21 |  1759 |    80 |     1 |
| 2004-02-08 10:56:44 | 11.11.11.67 | 207.66.155.21 |  1759 |    80 |     1 |
| 2004-02-08 10:56:56 | 11.11.11.67 | 207.66.155.21 |  1759 |    80 |     1 |
| 2004-02-08 10:57:20 | 11.11.11.67 | 207.66.155.21 |  1759 |    80 |     1 |
| 2004-02-08 10:58:08 | 11.11.11.67 | 207.66.155.21 |  1759 |    80 |     1 |
| 2004-02-08 11:01:20 | 11.11.11.67 | 207.66.155.21 |  1762 |    80 |     1 |
| 2004-02-08 11:46:51 | 11.11.11.67 | 207.66.155.21 |  1764 |    80 |     1 |
| 2004-02-08 11:46:54 | 11.11.11.67 | 207.66.155.21 |  1764 |    80 |     1 |
| 2004-02-08 11:47:00 | 11.11.11.67 | 207.66.155.21 |  1764 |    80 |     1 |
| 2004-02-08 11:47:12 | 11.11.11.67 | 207.66.155.21 |  1764 |    80 |     1 |
| 2004-02-08 11:47:36 | 11.11.11.67 | 207.66.155.21 |  1764 |    80 |     1 |
| 2004-02-08 11:48:24 | 11.11.11.67 | 207.66.155.21 |  1765 |    80 |     1 |
| 2004-02-08 11:48:27 | 11.11.11.67 | 207.66.155.21 |  1765 |    80 |     1 |
| 2004-02-08 11:48:33 | 11.11.11.67 | 207.66.155.21 |  1765 |    80 |     1 |
| 2004-02-08 11:48:45 | 11.11.11.67 | 207.66.155.21 |  1765 |    80 |     1 |
| 2004-02-08 11:49:09 | 11.11.11.67 | 207.66.155.21 |  1765 |    80 |     1 |
| 2004-02-08 11:51:45 | 11.11.11.67 | 207.66.155.21 |  1767 |    80 |     1 |
| 2004-02-08 11:56:19 | 11.11.11.67 | 207.66.155.21 |  1769 |    80 |     1 |
| 2004-02-08 12:10:27 | 11.11.11.67 | 207.66.155.21 |  1779 |    80 |     1 |
| 2004-02-08 12:36:41 | 11.11.11.67 | 207.66.155.21 |  1794 |    80 |     1 |
| 2004-02-08 12:36:44 | 11.11.11.67 | 207.66.155.21 |  1794 |    80 |     1 |
+---------------------+-------------+---------------+-------+-------+-------+
23 rows in set (0.51 sec)
```

Now we can see 11.11.11.67 is hacked and maybe try to get some tarballs via HTTP.

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?

First, the top 3 src should be considered a high threat, i.e. 66.60.166.84, 66.186.83.178, 63.13.135.27

```
mysql> select distinct dport from tblAllTraffic where srcip="66.60.166.84";
+-------+
| dport |
+-------+
|   443 |
+-------+
```

```
mysql> select count(*) as mycount, dstip from tblAllTraffic where srcip="66.60.166.84"
group by dstip;
```

```
+---------+--------------+
| mycount | dstip        |
+---------+--------------+
|     311 | 11.11.11.105 |
|     519 | 11.11.11.69  |
|       2 | 11.11.11.73  |
|   19584 | 11.11.11.75  |
|     478 | 11.11.11.82  |
|     439 | 11.11.11.87  |
|     496 | 11.11.11.89  |
+---------+--------------+
```

See? Different hosts got different connection times.

```
mysql> select distinct dport from tblAllTraffic where srcip="66.186.83.178";
+-------+
| dport |
+-------+
|   445 |
|   139 |
+-------+
mysql> select count(*) as mycount, dstip from tblAllTraffic where srcip="66.186.83.178" and dport=445 group by dstip;
+-----------+----------------+
| mycount | dstip          |
+-----------+----------------+
|     238 | 11.11.11.100 |
|     261 | 11.11.11.105 |
|     471 | 11.11.11.110 |
|     478 | 11.11.11.115 |
|     480 | 11.11.11.120 |
|     486 | 11.11.11.125 |
|     392 | 11.11.11.67  |
|     397 | 11.11.11.69  |
|     393 | 11.11.11.70  |
|     385 | 11.11.11.71  |
|     388 | 11.11.11.72  |
|     391 | 11.11.11.73  |
|     388 | 11.11.11.75  |
|     387 | 11.11.11.80  |
|     384 | 11.11.11.81  |
|     217 | 11.11.11.82  |
|     224 | 11.11.11.83  |
|     210 | 11.11.11.84  |
|     220 | 11.11.11.85  |
|     212 | 11.11.11.87  |
|         213 | 11.11.11.89    |
|         218 | 11.11.11.90    |
|         224 | 11.11.11.95    |
+-----------+-----------------+
```

The connections among hosts are even. It’s also the same with port 139

```
mysql> select distinct dport from tblAllTraffic where srcip="63.13.135.27";
+-------+
| dport |
+-------+
|     113 |
|     445 |
|     139 |
|     137 |
+-------+
mysql> select count(*) as mycount, dstip from tblAllTraffic where srcip="63.13.135.27"
and dport=445 group by dstip;
+-----------+----------------+
| mycount | dstip               |
+-----------+----------------+
|         151 | 11.11.11.100 |
|         141 | 11.11.11.105 |
|         132 | 11.11.11.110 |
|         141 | 11.11.11.115 |
|         143 | 11.11.11.120 |
|         129 | 11.11.11.125 |
|         150 | 11.11.11.67    |
|         150 | 11.11.11.69    |
|         159 | 11.11.11.70    |
|         156 | 11.11.11.71    |
|         147 | 11.11.11.72    |
|         144 | 11.11.11.73    |
|         146 | 11.11.11.75    |
|         138 | 11.11.11.80    |
|         152 | 11.11.11.81    |
|         141 | 11.11.11.82    |
|         143 | 11.11.11.83    |
|         139 | 11.11.11.84    |
|         150 | 11.11.11.85    |
|         143 | 11.11.11.87    |
|         142 | 11.11.11.89    |
|         144 | 11.11.11.90    |
|         149 | 11.11.11.95    |
+-----------+-----------------+
```

The connections among hosts are even. It’s also the same with port 113,137,139

Next, the source who hacked the honeynet hosts should be a high threat, but the firewall log can't supply enough information. Since most of the time when a hacker gets in the system, he will make outbound connection immediately. So we can make some inspect about the connection before the outbound connection.

There are 7 hosts who made outbound connections, 11.11.11.67, 11.11.11.69, 11.11.11.71, 11.11.11.72, 11.11.11.73, 11.11.11.75, 11.11.11.80.

It seems that 11.11.11.67 is hacked by ???.???.???.???
It seems that 11.11.11.69 is hacked by 80.116.93.36?
It seems that 11.11.11.71 is hacked by 80.116.93.36?
It seems that 11.11.11.72 is hacked by 80.116.93.36?
It seems that 11.11.11.73 is hacked by 211.182.117.130?
It seems that 11.11.11.75 is hacked by 195.36.244.104?
It seems that 11.11.11.80 is hacked by 211.182.117.130?

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?

From the huge scans on port 135, 443, 445, 3127, and the worms running on the Internet, Windows seems to be the most attacked system.. Among them,

11.11.11.75 with open ports: 21, 443
11.11.11.80 with ports: 21, 139, 1434?
11.11.11.67 with ports: 21 137 138 139 443

I'm not sure about the ports opened on the hosts, the difficulty is that the firewall is stateful, so the respond packet to the open port request is not logged, we can't tell whether the respond packet is a RST or a ACK+SYN. Also, if a host is hacked and makes an outbound connection, he can use Internal NET Privileged port -> External Net Unprivileged port to escape from the stateless firewall.

If the attack is from the worms, the worms will treat every host equally, but if the attack is from man, he usually detects the OS of the target first, and choose the specified tool to hack.. So the difference exists towards hosts with close IPs.

Bonus Question:

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

The analysis has been give in question 1, such as:

TOP 30 frequently connected ports

```
select count(*) as mycount , dport from tblAllTraffic group by dport order by mycount desc limit 30;
```

```
+-----------+---------+
| mycount | dport    |
+-----------+---------+
|     88157 |      135 |
|     46439 |      445 |
|     26444 |      443 |
|     25781 |    3127 |
|     18156 |        53 |
|     15000 |      139 |
|     13309 |      80 |
|      8742 |     137 |
|      5909 |    1434 |
|      3773 |     138 |
|      3426 |    6129 |
|      3097 |     901 |
|      2791 |    1433 |
|      2401 |    1026 |
|      2147 |   17300 |
|      1932 |    1080 |
|      1539 |    3128 |
|      1412 |    4899 |
|      1107 |      21 |
|       557 |   10080 |
|       536 |   27374 |
|       421 |     113 |
|       413 |      23 |
|       381 |     111 |
|       378 |   20168 |
|       304 |    1038 |
|       294 |    1027 |
|       292 |      25 |
|       282 |     389 |
|       249 |   45295 |
+-----------+---------+
```

From the data above we can see that most traffic are caused by worms, the victims are mostly Windows system. The scans go on and on and the net is extremely unsafe.