Network Situational Awareness: Internet Traffic Analysis

46.168.0.0/16

Michael Scotto

Danial Ranjha

Analysis Outline

1. **Executive Summary**
2. **Identifying Key Resources**
    - 2.1 DNS and HTTP Servers
    - 2.2 Analysis of Protocols and Ports
    - 2.3 DNS Server Details
    - 2.4 Outgoing Traffic
    - 2.5 Incoming Traffic
    - 2.6 Routers and ICMP Messages
3. **Geographical Location**
4. **Deep Inspection**
5. **Architecture Recommendation**
6. **Internet-wide event and its effects on our network**

1.Executive Summary

The following report discusses analysis of anonymized class B network 46.168.0.0/16. This network was chosen randomly from a list of all class B addresses in the MAWI data source on a major routing point in the internet. Data from the analysis was performed using the SILK network flow data analysis tool. During our analysis we acquired some very interesting information. Perhaps one of the most interesting finds is the lack of identifiable client traffic detected on the MAWI data collection point. We believe this to be caused by client traffic routing through a different other than our data source. Key discoveries of the analysis report are as follows:

DNS and HTTP Servers: 46.168.x.x contains many web servers and several DNS servers. Specific IP address of DNS and HTTP servers are identified in the report. The largest web server comprised about 9% of all collected traffic while the largest DNS server comprised about 5% of all network traffic.

Ports and Protocols: Only 3 protocols are in use on this particular network, TCP, UDP and ICMP. There is a handful of ports that are used in our network. One of the interesting discoveries is the lack of ephemeral port traffic to our network. This is the leading cause that we believe that we cannot identify many clients on our network. Ports 53(DNS) and 80(Web Traffic) were the most prominently used ports on our network.

DNS Server Details: DNS traffic is mainly delegated to one DNS server with 43% of DNS traffic going to the main DNS server on our network. Most DNS traffic is the result of communication with another DNS server on the network. We did not witness many forwarded DNS packets from DNS servers on the network. We conclude that either traffic is being routed through a different route from the data source or that these DNS servers do not allow DNS forwarding.

Outgoing Traffic: It appears that there is no cache server on this particular network; if there was we would see a large amount of web traffic coming from and going to one address. Almost all outgoing traffic is to ephemeral ports from web servers on our networks. This is another reason why it was difficult to locate any client machines on the network.

Incoming Traffic: We found that 43% of all incoming traffic is coming from a machine on an external network. This is most likely a proxy server. Almost all incoming traffic are to ports 80 and 53 server as more evidence on the composition of our network being mostly web and DNS servers. There are very few incoming flows with a destination port not in the top 10 port list. Web surfing machines would usually have incoming flows with ephemeral ports. There was only 136/10000 flows that did not have a destination port in the top ten port list.

Routers and ICMP Messages: An attempt was made to discover routers on the network by searching for TTL time expired ICMP messages. However it was found that all ICMP messages coming from this network were echo or echo reply messages. This made it impossible to identify the location of any routers in the network .

Ephemeral port traffic: We carried out some analysis of traffic that we did not recognize. One such example was ephemeral port to ephemeral port communication. We found an interesting set of traffic destined towards port 20480 and offered three possible explanations for the existence of such traffic. That is, this was either a NetBUI acknowledgement packet, multiplayer gaming traffic, or a malformed packet destined for port 80.

Geographical analysis: We made an attempt to determine the geographical location of our servers based on the network traffic. We found that these servers were not geographically collocated and were probably somewhere in Russia or China (assuming we used a complete dataset and had complete day's traffic).

Architectural analysis: Given our data we assumed that our analysis set was part of traffic coming from an ISP. We identified some of the goals of an ISP as a service. We then talked about some problems that we found in the architecture during our analysis. That is, we found evidence of asymmetric routing and identified potential problems that can be a cause of asymmetric routing. Particularly, we talked about issues with Internet gateway devices such as firewalls and proxies. We offered architectural recommendations to remedy this situation.

Internet wide event: We chose the activities of Botnet infrastructure as an internet-wide event. In particular we looked at DNS fast-flux, and its use by the Storm network. Cybercriminals can use DNS fast-flux techniques to avoid detection and also to keep their malicious content providers running for a longer periods of time. We tried to look for evidence of such activities within our subnet using Snort IDS. Having found none we then investigated data from other networks. Having found some malware activity we concluded that it is quite likely that there was traffic from cybercriminals on our network. We offered some suggestions on how our ISP would have been affected if it was a victim of DNS fast-flux requests and suggested the use of intrusion detection systems and malware gateways for mitigation of this threat.

Note: Narrative text is in this analysis is Yellow while major headings are listed in Brown as to improve readability to the reader among large amounts of data output

2. Identifying Key Resources

2.1 DNS AND HTTP SERVERS

Class B Source: 46.168.x.x

Key Resources used by the network

```
[mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --saddress=46.168.x.x --prin
t-volume-stat --pass=stdout | bin/rwstats --sip --top --flow --count=5
     |           Recs|         Packets|           Bytes|     Files|
Total|         2609483|        28182301|       18752648387|         1|
 Pass|            9973|           83651|          35762293|          |
 Fail|         2599510|        28098650|       18716886094|          |
INPUT SIZE: 9973 records for 136 unique keys
SOURCE IP Key: Top 5 flow counts
          sIP|          Records|%_of_total|   cumul_%|
   46.168.96.5|              877|  8.793743|  8.793743|   Web Server
  46.168.96.27|              530|  5.314349| 14.108092|   Web Server
 46.168.96.143|              414|  4.151208| 18.259300|   DNS
  46.168.106.73|              397|  3.980748| 22.240048|  Web Server
  46.168.105.28|              367|  3.679936| 25.919984|  Web Server
```

The following section will identify and analyze these top 5 servers in our network

What kind of data is going to 46.168.96.5?

```
[mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.96.5 --print-volume-stat --
pass=stdout | bin/rwcut --fields=sip,dip,sport,dport | less
     |           Recs|         Packets|           Bytes|     Files|
Total|         2609483|        28182301|       18752648387|         1|
 Pass|             884|           6424|          2900163|          |
 Fail|         2608599|        28175877|       18749748224|          |
          sIP|          dIP|sPort|dPort|
  204.80.30.250|    46.168.96.5|65415|   80|
   204.80.31.60|    46.168.96.5|47683|   80|
   204.80.31.60|    46.168.96.5|50241|   80|
   204.80.31.60|    46.168.96.5|49991|   80|
   204.80.31.60|    46.168.96.5|36093|   80|
```

```
  204.80.31.60|    46.168.96.5|47592|   80|
  204.80.31.60|    46.168.96.5|38982|   80|
 136.90.16.213|    46.168.96.5| 6154|   80|
  204.80.31.60|    46.168.96.5|42026|   80|
  204.80.31.60|    46.168.96.5|37950|   80|
  204.80.31.60|    46.168.96.5|36519|   80|
  204.80.31.60|    46.168.96.5|58499|   80|
  204.80.31.60|    46.168.96.5|34083|   80|
  204.80.31.60|    46.168.96.5|60572|   80|
```

Looks like Source Ip 46.168.96.5 is a web server.

What kind of data is going to 46.168.96.27?

[mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.96.27 --print-volume-stat --pass=stdout | bin/rwcut --fields=sip,dip,sport,dport | less

```
         sIP|          dIP|sPort|dPort|
  136.90.45.47|   46.168.96.27|59730|   80|
 204.80.30.250|   46.168.96.27|49159|   80|
 136.90.71.132|   46.168.96.27| 2957|   80|
  204.80.31.60|   46.168.96.27|58775|   80|
  204.80.31.60|   46.168.96.27|41545|   80|
  204.80.31.60|   46.168.96.27|55588|   80|
  204.80.31.60|   46.168.96.27|37289|   80|
  204.80.31.60|   46.168.96.27|34631|   80|
  204.80.31.60|   46.168.96.27|37822|   80|
  204.80.31.60|   46.168.96.27|51258|   80|
```

46.168.96.7 is a web server

What kind of data is going to 46.168.96.143

mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.96.143 --print-volume-stat --pass=stdout | bin/rwcut --fields=sip,dip,sport,dport | less

```
         sIP|          dIP|sPort|dPort|
  204.80.31.74|  46.168.96.143|54731|   53|
205.100.248.230|  46.168.96.143|48330|   53|
```

```
  204.80.31.74|  46.168.96.143|55009|   53|
  204.80.31.74|  46.168.96.143|52239|   53|
 141.179.243.11|  46.168.96.143|65353|   53|
  204.80.30.230|  46.168.96.143|14986|   53|
  204.80.31.74|  46.168.96.143|55392|   53|
  136.90.167.65|  46.168.96.143|45869|   53|
```

46.168.96.143 is a DNS server

What kind of data is going to 46.168.106.73?

[mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.106.73 --print-volume-stat --pass=stdout | bin/rwcut --fields=sip,dip,sport,dport | less

```
          sIP|          dIP|sPort|dPort|
 169.185.154.41|  46.168.106.73|57044|   80|
  204.80.30.250|  46.168.106.73|51963|   80|
  204.80.30.250|  46.168.106.73|51977|   80|
  204.80.30.250|  46.168.106.73|51982|   80|
  204.80.30.250|  46.168.106.73|52004|   80|
  204.80.30.250|  46.168.106.73|52007|   80|
```

46.168.106.73 is a Web Server

What kind of data is going to0 46.168.105.28 ?

[mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.105.28 --print-volume-stat --pass=stdout | bin/rwcut --fields=sip,dip,sport,dport | less

```
          sIP|          dIP|sPort|dPort|
205.100.245.160|  46.168.105.28|49787|   80|
   192.168.2.45|  46.168.105.28|61922|   80|
   136.90.66.98|  46.168.105.28|50485|   80|
 169.185.152.47|  46.168.105.28|49823|   80|
  204.80.30.250|  46.168.105.28|53232|   80|
  204.80.30.250|  46.168.105.28|53327|   80|
```

46.168.105.28 is a Web Server

2.2 Analysis of Protocols and Ports

What Protocols are we hosting on our network?'

[mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.X.X --prin
t-volume-stat --pass=stdout | bin/rwstats --protocol --count=9

INPUT SIZE: 10946 records for 3 unique keys
PROTOCOL Key: Top 9 flow counts

protocol	Records	%_of_total	cumul_%
6	9680	88.434131	88.434131
17	1205	11.008588	99.442719
1	61	0.557281	100.000000

Protcol 6 is TCP, Protcol 17 is UDP and Protocol 1 is ICMP. We are not receiving any communication other than these protocols.

What Ports are we hosting on our network?

[mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.X.X --print-volume-stat
--pass=stdout | bin/rwstats --dport --count=9 INPUT SIZE: 10946 records for 121 unique keys
DESTINATION PORT Key: Top 9 flow counts

dPort	Records	%_of_total	cumul_%
80	9600	87.703271	87.703271
53	1083	9.894025	97.597296
443	35	0.319752	97.917047
20480	32	0.292344	98.209392
0	23	0.210122	98.419514
771	16	0.146172	98.565686
781	11	0.100493	98.666179
2816	5	0.045679	98.711858
769	5	0.045679	98.757537

The port use assignments are as follows:

Port 80 - HTTP
Port 53 – DNS
Port 443- SSL HTTP
Port 20480 – Ephemeral port
Port 0 - ICMP

Port 771 - rTip
Port 781 - hp-collector.hp performance data collector
Port 2816 - LBC Watchdog
Port 769 - Could not Identify

rTip description -" RTIP-32 is a TCP/IP stack for use in embedded systems. It provides deterministic, configurable memory usage, a table driven device driver interface and an easy-to-use API. The software can configure itself at runtime using standard RARP, BOOTP, and optionally DHCP protocols."

Hp collector description - **"**The Performance Data Collector for HP OpenVMS (TDC) gathers performance data for OpenVMS systems. No additional license or cost is required to use TDC. By default, TDC periodically collects and stores data in a file. Subsequently, user applications can retrieve and analyze data from the file."
LBC-Watchdog – Could not find any info regarding this application

2.3 DNS SERVER DETAILS

Let's find out more about our DNS server(s).

```
[mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.X.X --dpor
t=53 --print-volume-stat --pass=stdout | bin/rwstats --dip --count=99

INPUT SIZE: 1083 records for 7 unique keys
DESTINATION IP Key: Top 99 flow counts
          dIP|          Records|%_of_total|   cumul_%|
 46.168.96.143|              467| 43.120960| 43.120960|
46.168.104.139|              160| 14.773777| 57.894737|
 46.168.96.206|              148| 13.665743| 71.560480|
 46.168.96.207|              124| 11.449677| 83.010157|
46.168.109.183|               98|  9.048938| 92.059095|
 46.168.109.57|               85|  7.848569| 99.907664|
46.168.221.248|                1|  0.092336|100.000000|
```

We can see that we have one main DNS server along with 5 other alternate DNS servers.

Let's look at our main DNS server

```
[mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.96.143 --d
port=53 --print-volume-stat --pass=stdout | bin/rwstats --sip --count=99

INPUT SIZE: 467 records for 30 unique keys
SOURCE IP Key: Top 99 flow counts
          sIP|          Records|%_of_total|   cumul_%|
 204.80.30.230|              141| 30.192719| 30.192719|
  204.80.31.74|              136| 29.122056| 59.314775|
  141.179.243.8|               38|  8.137045| 67.451820|
169.185.96.122|               31|  6.638116| 74.089936|
141.179.243.11|               25|  5.353319| 79.443255|
136.90.179.179|               17|  3.640257| 83.083512|
 205.100.244.22|               17|  3.640257| 86.723769|
141.179.243.122|               13|  2.783726| 89.507495|
205.100.248.230|                8|  1.713062| 91.220557|
 204.80.30.250|                5|  1.070664| 92.291221|
 136.90.179.184|                5|  1.070664| 93.361884|
```

We can see that 59% of our DNS traffic comes from class B address 204.80.X.X.

Why do we have so much DNS traffic coming from 204.80.x.x?

[mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --saddress=204.80.30.230 --print-volume-stat --pass=stdout | bin/rwstats --dport --count=20

INPUT SIZE: 2915 records for 7 unique keys
DESTINATION PORT Key: Top 20 flow counts

dPort	Records	%_of_total	cumul_%
53	2877	98.696398	98.696398
0	30	1.029160	99.725557
771	4	0.137221	99.862779
45968	1	0.034305	99.897084
3630	1	0.034305	99.931389
3623	1	0.034305	99.965695
3627	1	0.034305	100.000000

So 204.80.30.230 is another DNS server communicating with our DNS server

[mjscotto@unix35 855]$ bin/rwfilter 200907071400.rw --saddress=204.80.31.74 --print-volume-stat --pass=stdout | bin/rwstats --dport --count=20
INPUT SIZE: 2270 records for 6 unique keys
DESTINATION PORT Key: Top 20 flow counts

dPort	Records	%_of_total	cumul_%
53	2199	96.872247	96.872247
0	65	2.863436	99.735683
771	3	0.132159	99.867841
32796	1	0.044053	99.911894
32971	1	0.044053	99.955947
60057	1	0.044053	100.000000

Same goes for 204.80.31.74

Are our DNS servers forwarding DNS requests from recursive queries to other DNS servers?

[mjscotto@unix38 855]$ bin/rwfilter 200907071400.rw --saddress 46.168.96.143 --dport=53 --pass=stdout | bin/rwstats --dport --count=10 --bottom
INPUT SIZE: 1 records for 1 unique keys
DESTINATION PORT Key: Bottom 10 flow counts

dPort	Records	%_of_total	cumul_%
53	1	100.000000	100.000000

DNS Server 46.168.96.143 has only one forwarded DNS request packet

[mjscotto@unix38 855]$ bin/rwfilter 200907071400.rw --saddress 46.168.96.206 --dport=53 --pass=stdout | bin/rwstats --dport --count=10 --bottom
INPUT SIZE: 1 records for 1 unique keys
DESTINATION PORT Key: Bottom 10 flow counts
        dPort| Records|%_of_total| cumul_%|
           53| 1|100.000000|100.000000|

46.168.96.206 has only one DNS record forwarded

[mjscotto@unix38 855]$ bin/rwfilter 200907071400.rw --saddress=46.168.104.139 --dport=53 --pass=stdout | bin/rwstats --dport --count=10 --bottom
INPUT SIZE: 0 records for 0 unique keys
DESTINATION PORT Key: Bottom 10 flow counts
        dPort| Records|%_of_total| cumul_%|

No DNS requests from DNS Server 46.168.104.139

[mjscotto@unix38 855]$ bin/rwfilter 200907071400.rw --saddress=46.168.96.207 --dport=53 --pass=stdout | bin/rwstats --dport --count=10 --bottom
INPUT SIZE: 0 records for 0 unique keys
DESTINATION PORT Key: Bottom 10 flow counts
        dPort| Records|%_of_total| cumul_%|
No DNS requests from DNS Server 46.168.96.207

[mjscotto@unix38 855]$ bin/rwfilter 200907071400.rw --saddress=46.168.109.183 --dport=53 --pass=stdout | bin/rwstats --dport --count=10 --bottom
INPUT SIZE: 0 records for 0 unique keys
DESTINATION PORT Key: Bottom 10 flow counts
        dPort| Records|%_of_total| cumul_%|
No DNS requests from DNS Server 46.168.109.183

[mjscotto@unix38 855]$ bin/rwfilter 200907071400.rw --saddress=46.168.109.57 --dport=53 --pass=stdout | bin/rwstats --dport --count=10 --bottom
INPUT SIZE: 0 records for 0 unique keys
DESTINATION PORT Key: Bottom 10 flow counts
        dPort| Records|%_of_total| cumul_%|\

No DNS requests from DNS Server 46.168.109.57

[mjscotto@unix38 855]$ bin/rwfilter 200907071400.rw --saddress=46.168.221.248 --dport=53 --pass=stdout | bin/rwstats --dport --count=10 --bottom
INPUT SIZE: 1 records for 1 unique keys
DESTINATION PORT Key: Bottom 10 flow counts
        dPort| Records|%_of_total| cumul_%|
           53| 1|100.000000|100.000000|

No DNS requests from DNS Server 46.168.221.248

For the DNS servers that do not have any outgoing packet flow with port 53 there can be two possibilities. Either these DNS servers are forwarding their requests away from our monitoring point or there is no DNS forwarding enabled on these DNS servers so they will not forward any DNS requests from clients.

2.4 OUTGOING TRAFFIC

Do we have a proxy server on our network?

```
[mjscotto@unix39 855]$ bin/rwfilter 200907071400.rw --saddress=46.168.x.x --dport=80 --
pass=stdout | bin/rwstats --dport --count=90
INPUT SIZE: 3 records for 1 unique keys
DESTINATION PORT Key: Top 90 flow counts
        dPort|          Records|%_of_total|   cumul_%|
          80|                3|100.000000|100.000000|
```

It appears that there is not outgoing web traffic from our subnet. This is a little strange.

```
[mjscotto@unix39 855]$ bin/rwfilter 200907071400.rw --saddress=46.168.x.x --dpor
t=80 --pass=stdout | bin/rwcut --fields=sip,dport,dip,packets
          sIP|dPort|          dIP|   packets|
 46.168.101.71|   80| 205.100.251.56|         6|
 46.168.101.70|   80| 205.100.251.56|        10|
 46.168.101.70|   80| 205.100.251.57|         7|
```

We only have records of three flows, what ports are we using the most for outgoing?

```
[mjscotto@unix39 855]$ bin/rwfilter 200907071400.rw --saddress=46.168.x.x  --pass=stdout |
bin/rwstats --dport --count=90
INPUT SIZE: 9973 records for 8635 unique keys
[mjscotto@unix36 855]$ bin/rwfilter 200907071400.rw --saddress=46.168.x.x  --pass=stdout |
bin/rwstats --dport --count=90
INPUT SIZE: 9973 records for 8635 unique keys
DESTINATION PORT Key: Top 90 flow counts
        dPort|          Records|%_of_total|   cumul_%|
          53|               83|  0.832247|  0.832247|
        2048|               36|  0.360975|  1.193222|
         445|               11|  0.110298|  1.303520|
        1024|                7|  0.070190|  1.373709|
       32797|                7|  0.070190|  1.443899|
        1028|                5|  0.050135|  1.494034|
       57425|                5|  0.050135|  1.544169|
       50209|                5|  0.050135|  1.594305|
        3747|                5|  0.050135|  1.644440|
       40243|                4|  0.040108|  1.684548|
       59738|                4|  0.040108|  1.724657|
       43997|                4|  0.040108|  1.764765|
       37088|                4|  0.040108|  1.804873|
       62801|                4|  0.040108|  1.844981|
       37114|                4|  0.040108|  1.885090|
       45340|                4|  0.040108|  1.925198|
```

It looks like the majority of our outgoing traffic is to ephemeral ports. This is most likely web content being served back to clients.

It doesn't seem likely that there would be only servers on our class B network. The most likely situation is that web content that is being requested by clients on our network are going a different route away from our source monitoring point. We can try to narrow down the clients by finding out the machines that do are non listening on port 80 or 53. Clients will typically not listen on 80 and 53 but this is not a hard and fast rule.

```
[mjscotto@unix36 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.x.x --dpor
t=80 --pass=stdout | bin/rwsort --fields=dip | bin/rwcut --fields=packets,sip,dip,sport,dport | less
rwsort: Warning: Using default temporary directory /tmp
   packets|           sIP|           dIP|sPort|dPort|
         7|   204.80.31.60|    46.168.96.0|53912|   80|
         8|   204.80.31.60|    46.168.96.0|46741|   80|
         7|  136.90.110.23|    46.168.96.0| 2924|   80|
         8|  204.80.30.250|    46.168.96.0|56804|   80|
```

2.5. INCOMING TRAFFIC

[mjscotto@unix36 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.x.x --dport=80 --pass=stdout | bin/rwstats --sip --count=10
INPUT SIZE: 9600 records for 92 unique keys
SOURCE IP Key: Top 10 flow counts

sIP	Records	%_of_total	cumul_%
204.80.30.250	4163	43.364583	43.364583
204.80.31.60	3702	38.562500	81.927083
136.90.66.98	84	0.875000	82.802083
205.100.250.68	74	0.770833	83.572917
141.179.128.236	72	0.750000	84.322917

Why does 43% of inbound web traffic come from one machine? What is this machine?

[mjscotto@unix38 855]$ bin/rwfilter 200907071400.rw --saddress=204.80.30.250 --pass=stdout | bin/rwstats --dip --count=10
INPUT SIZE: 11021 records for 941 unique keys
DESTINATION IP Key: Top 10 flow counts

dIP	Records	%_of_total	cumul_%
71.34.154.81	1205	10.933672	10. 933672
46.219.2.32	471	4.273659	15.207331
210.139.107.23	436	3.956084	19.163415
46.168.105.28	382	3.466110	22.629525
46.168.96.64	277	2.513384	25.142909
46.168.96.9	252	2.286544	27.429453
46.168.96.30	243	2.204882	29.634334
46.168.105.163	240	2.177661	31.811995
46.168.105.162	238	2.159514	33.971509
46.168.107.81	225	2.041557	36.013066

This machine is sending out a lot of traffic to many different networks

[mjscotto@unix38 855]$ bin/rwfilter 200907071400.rw --daddress=204.80.x.x --pass=stdout | bin/rwstats --dip --count=10
INPUT SIZE: 65207 records for 3185 unique keys
DESTINATION IP Key: Top 10 flow counts

dIP	Records	%_of_total	cumul_%
204.80.30.2	15839	24.290337	24.290337

204.80.42.67| 9109| 13.969359| 38.259696|
204.80.30.250| 8635| 13.242443| 51.502139|
204.80.31.60| 6550| 10.044934| 61.547073|
204.80.31.3| 6234| 9.560323| 71.107396|

This machine is most likely a proxy server

This is the beginning of the list of a thousands of IP's that are listening on port 80…

| 204.80.31.60| 46.168.96.10|41730| 80|
9| 204.80.31.60| 46.168.96.10|36395| 80|
6| 169.185.0.157| 46.168.96.10| 4878| 80|
8| 204.80.31.60| 46.168.96.10|33324| 80|
7| 205.100.250.68| 46.168.96.10|62215| 80|
7| 205.100.250.68| 46.168.96.10|62221| 80|
8| 204.80.31.60| 46.168.96.10|38409|
Continues through .10

7| 204.80.30.250| 46.168.96.30|60429| 80|
7| 204.80.30.250| 46.168.96.30|51911| 80|
7| 204.80.30.250| 46.168.96.30|51921| 80|
7| 204.80.30.250| 46.168.96.30|51928| 80|
8| 204.80.30.250| 46.168.96.30|51937| 80|
7| 204.80.30.250| 46.168.96.30|60418| 80|
7| 204.80.30.250| 46.168.96.30|60400| 80|
And through .30

7| 136.90.204.41| 46.168.96.80| 4043| 80|
3| 136.90.204.41| 46.168.96.80| 3991| 80|
7| 136.90.66.98| 46.168.96.181|58104| 80|
1| 136.90.66.98| 46.168.96.181|58104| 80|
10| 136.90.66.98| 46.168.96.181|58111| 80|
1| 136.90.66.98| 46.168.96.181|58107| 80|
19| 136.90.66.98| 46.168.96.181|58107| 80|
1| 136.90.66.98| 46.168.96.181|58111| 80|
22| 204.80.31.60| 46.168.96.201|42687| 80|
1| 141.179.151.56| 46.168.96.201| 1595| 80|
10| 141.179.151.56| 46.168.96.201| 1595| 80|
7| 204.80.31.60| 46.168.96.201|52826| 80|
11| 204.80.31.60| 46.168.96.201|36149| 80|
22| 204.80.31.60| 46.168.96.201|40506| 80|

And then a break from .80 to .181 and .201

108|169.185.138.205| 46.168.97.37|53323| 80|
5| 204.80.30.250| 46.168.99.135|55672| 80|
5| 204.80.30.250| 46.168.99.135|58492| 80|
6| 204.80.31.60| 46.168.99.135|40395| 80|
440| 204.80.30.250| 46.168.99.166|53178| 80|
8| 136.90.67.24| 46.168.99.166| 4532| 80|
258| 204.80.31.60| 46.168.99.166|54702| 80|

46.168.97.37 , 46.168.99.135, 46.168.99.166

17| 204.80.31.60| 46.168.101.176|40812| 80|
17| 204.80.31.60| 46.168.101.176|53982| 80|

46.168.101.176

| 204.80.31.60| 46.168.101.178|32907| 80|
7| 204.80.30.250| 46.168.101.178|57854| 80|
2|169.185.200.206| 46.168.105.28|46274| 80|
2|169.185.200.206| 46.168.105.28|46272| 80|
8| 204.80.30.250| 46.168.105.28|55731| 80|

Then a break from 46.168.101.176 to 46.168.105.28

And it ends at 46.168.107.85|. Although the address space is quite broken up for now we will assume that the server starts at 46.168.96.0 and ends at 46.168.107.85. That's approximately 255* 11 = 2805 machines listening on port 80 in this address space.

[mjscotto@unix36 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.x.x --pass=stdout | bin/rwfilter --input-pipe=stdin --dport=80 --fail=stdout | bin/rwfilter --input-pipe=stdin --dport=53 --fail=stdout | bin/rwcut --fields=dip,sip,dport,sport
dIP| sIP|dPort|sPort|
46.168.96.5| 136.90.16.213|20480| 6194|
46.168.101.38| 136.90.106.112| 443| 3157|
46.168.99.127|141.179.243.125|54188| 25|
46.168.101.36| 136.90.249.10| 443|49600|
46.168.96.5| 136.90.16.213|20480| 6306|
46.168.101.71| 205.100.251.56|49340| 80|
46.168.96.44| 136.90.167.42| 443| 1482|
46.168.101.70| 205.100.251.56|58805| 80|
46.168.96.5| 136.90.16.213|20480| 6378|
46.168.101.38| 136.90.167.42| 443| 1535|
46.168.101.38| 204.80.31.60| 443|40835|
46.168.169.110| 141.169.184.1| 4608| 445|
46.168.169.110| 141.169.184.1| 4608| 445|
46.168.169.110| 141.169.184.1| 4608| 445|

46.168.96.5| 136.90.16.213|20480| 6426|
46.168.96.44|141.179.128.236| 443|55804|
46.168.101.38| 169.185.134.35| 443| 2703|
46.168.96.5| 136.90.16.213|20480| 7546|
46.168.101.50| 136.90.107.216| 443|51693|
46.168.96.5| 136.90.16.213|20480| 7642|
46.168.163.57| 136.90.218.88|51408|64076|
46.168.163.57| 136.90.218.88| 771| 0|
46.168.101.36| 141.179.242.90| 443|57493|
46.168.105.87| 136.90.177.96| 781| 0|
46.168.105.87| 136.90.166.53| 6260| 53|
46.168.169.110| 205.100.245.83| 2816| 0|
46.168.106.162| 136.90.177.98| 781| 0|
46.168.98.55| 193.137.56.229|39928| 53|
46.168.96.80| 136.90.66.98| 769| 0|
46.168.96.145| 136.90.177.98| 781| 0|
46.168.96.96| 136.90.179.179| 0| 0|
46.168.99.120|141.179.243.125|41229| 25|
46.168.98.56|141.179.243.125| 25|15817|
46.168.104.144| 193.137.56.229|35805| 53|
46.168.96.5| 136.90.16.213|20480| 7754|
46.168.99.12| 193.137.56.229| 5863| 53|
46.168.99.3| 193.137.56.229|64991| 53|

This is is very interesting, there are only about 1500 records coming outside of our class B that do NOT have a destination of port 53 or port 80. And most of these records have destination ports in our top 10 destination port lists. Records that pertain to destination ports in the top 10 are highlighted in red

The top ten port list is

Port 80 - HTTP
Port 53 – DNS
Port 443- SSL HTTP
Port 20480 – Ephemeral port
Port 0 - ICMP
Port 771 - rTip
Port 781 - hp-collector.hp performance data collector
Port 2816 - LBC Watchdog
Port 769 - Could not Identify

Let's see what happens when we query for records that do not have destination ports in the top 10 list

The following is the complete list from the entire list of records with a destination for our class B that do NOT have a destination port in our top ten ports. There are only 136 records that match this query.

```
bin/rwfilter 200907071400.rw --daddress=46.168.x.x --pass=stdout | bin/rwfilter --input-pipe=stdin --
dport=80 --fail=stdout | bin/rwfilter --input-pipe=stdin --dport=53 --fail=stdout | bin/rwfilter --input-
pipe=stdin --dport=443 --fail=stdout |  bin/rwfilter --input-pipe=stdin --dport=20480 --fail=stdout |
bin/rwfilter --input-pipe=stdin --dport=0 --fail=stdout | bin/rwfilter --input-pipe=stdin --dport=771 --
fail=stdout | bin/rwfilter --input-pipe=stdin --dport=781 --fail=stdout | bin/rwfilter --input-pipe=stdin --
dport=2816 --fail=stdout |bin/rwfilter --input-pipe=stdin --dport=769 --fail=stdout |    bin/rwcut --
fields=dip,sip,dport,sport
          dIP|           sIP|dPort|sPort|
 46.168.99.127|141.179.243.125|54188|   25|
 46.168.101.71| 205.100.251.56|49340|   80|
 46.168.101.70| 205.100.251.56|58805|   80|
46.168.169.110|  141.169.184.1| 4608|  445|
46.168.169.110|  141.169.184.1| 4608|  445|
46.168.169.110|  141.169.184.1| 4608|  445|
 46.168.163.57|  136.90.218.88|51408|64076|
 46.168.105.87|  136.90.166.53| 6260|   53|
  46.168.98.55| 193.137.56.229|39928|   53|
 46.168.99.120|141.179.243.125|41229|   25|
  46.168.98.56|141.179.243.125|   25|15817|
46.168.104.144| 193.137.56.229|35805|   53|
  46.168.99.12| 193.137.56.229| 5863|   53|
   46.168.99.3| 193.137.56.229|64991|   53|
   46.168.99.2| 193.137.56.227| 8962|   53|
  46.168.99.12| 193.137.56.227|43711|   53|
   46.168.99.3| 205.100.241.52|36442|   53|
46.168.105.107|141.179.243.125|47967|   25|
 46.168.101.84|  136.90.204.49|33340|   53|
  46.168.99.14| 193.137.56.227|44408|   53|
 46.168.96.175| 205.100.241.52|47074|   53|
46.168.128.166| 205.100.250.69|16443|63463|
  46.168.99.15| 205.100.241.55|55785|   53|
  46.168.99.12| 193.137.56.229|59698|   53|
46.168.227.101| 141.179.16.242| 6345|36735|
  46.168.98.54| 205.100.241.52|39928|   53|
46.168.104.144| 193.137.56.227|35805|   53|
46.168.182.125| 193.137.56.227| 1905|   53|
46.168.182.125| 193.137.56.227|61615|   53|
46.168.182.125| 193.137.56.227| 4213|   53|
46.168.182.125| 193.137.56.227|26223|   53|
 46.168.96.148| 136.90.242.109|34077|   53|
46.168.107.178| 136.90.242.109|33519|   53|
```

46.168.182.19| 193.137.56.229|48973| 53|
46.168.169.9| 205.100.250.68|17332|62654|
46.168.99.15|141.179.118.154| 6351| 53|
46.168.104.145| 193.137.56.227|39037| 53|
46.168.182.21| 205.100.241.55| 5457| 53|
46.168.98.54| 193.137.56.227|33972| 53|
46.168.104.145|141.179.118.154|39037| 53|
46.168.104.145|141.179.118.154|34684| 53|
46.168.105.87| 136.90.166.53|42471| 53|
46.168.104.144| 193.137.56.227|34684| 53|
46.168.104.140| 136.90.204.49|32787| 53|
46.168.98.55| 193.137.56.227|33483| 53|
46.168.105.87| 136.90.242.109|54729| 53|
46.168.104.144| 193.137.56.227|33136| 53|
46.168.105.206| 136.90.166.53|34258| 53|
46.168.96.161| 136.90.166.53|32856| 53|
46.168.104.142| 204.91.2.32|32937| 53|
46.168.99.14| 205.100.241.55|34956| 53|
46.168.29.115| 141.169.28.114| 3218| 445|
46.168.29.115| 141.169.28.114| 3218| 445|
46.168.29.115| 141.169.28.114| 3218| 445|
46.168.98.54| 193.137.56.227|54878| 53|
46.168.101.70| 205.100.251.57|51004| 80|
46.168.99.15| 193.137.56.227|10881| 53|
46.168.99.13| 205.100.241.52|51259| 53|
46.168.96.161| 204.91.2.32|32856| 53|
46.168.98.54| 205.100.241.52|33483| 53|
46.168.96.168| 205.100.241.55|47074| 53|
46.168.96.172| 169.185.37.4|47074| 53|
46.168.99.13| 193.137.56.227|24429| 53|
46.168.99.2| 205.100.241.55| 7517| 53|
46.168.98.54| 193.137.56.229|33483| 53|
46.168.98.55| 193.137.56.227|32977| 53|
46.168.0.201| 193.137.56.229| 7581| 53|
46.168.96.100| 193.137.56.227| 2154| 53|
46.168.169.115| 205.100.250.68| 8148|63357|
46.168.99.2| 193.137.56.227|62793| 53|
46.168.99.16| 136.90.204.49|54371| 53|
46.168.107.148| 136.90.204.49|29553| 53|
46.168.176.191| 205.100.250.69|18667|61646|
46.168.99.12|141.179.118.154|34104| 53|
46.168.164.144| 205.100.250.68| 7525|62835|

46.168.99.3| 205.100.241.52|19901| 53|
46.168.182.126|141.179.118.154|50998| 53|
46.168.161.70| 205.100.250.68|27491|63001|
46.168.98.54| 193.137.56.227|39928| 53|
46.168.166.247|141.179.213.184|60038|58489|
46.168.227.101| 141.179.200.6| 6345|65013|
46.168.227.101| 141.179.199.68| 6345|49763|
46.168.168.70| 205.100.250.68|12995|61850|
46.168.182.21| 205.100.241.55|21694| 53|
46.168.182.21| 205.100.246.69|15393| 53|
46.168.182.21| 205.100.246.69|29264| 53|
46.168.182.21| 205.100.246.69|15667| 53|
46.168.182.21| 205.100.241.48|26361| 53|
46.168.182.21| 205.100.241.48| 4551| 53|
46.168.182.21| 205.100.241.48|34836| 53|
46.168.182.21| 205.100.241.48|59311| 53|
46.168.182.21| 205.100.250.83|25406| 53|
46.168.182.21| 205.100.250.83|41044| 53|
46.168.182.21| 205.100.250.83|62564| 53|
46.168.182.21| 205.100.250.83|49070| 53|
46.168.182.21| 205.100.250.85|50549| 53|
46.168.182.21| 205.100.250.83|17105| 53|
46.168.182.21| 205.100.250.83| 8065| 53|
46.168.182.21| 205.100.250.85|64347| 53|
46.168.105.99| 136.90.204.49|34095| 53|
46.168.96.150| 136.90.204.49|34044| 53|
46.168.99.14| 193.137.56.227|31652| 53|
46.168.98.55| 205.100.241.55|54878| 53|
46.168.99.13| 205.100.241.52|51499| 53|
46.168.104.145| 193.137.56.227|35805| 53|
46.168.185.80| 193.137.56.227| 1702| 53|
46.168.182.125| 193.137.56.229|17813| 53|
46.168.182.125| 193.137.56.229|52282| 53|
46.168.98.54| 204.80.31.96|33483| 53|
46.168.99.13| 205.100.241.55|40468| 53|
46.168.176.4|141.179.215.149| 4476| 7508|
46.168.104.145| 205.100.241.52|35805| 53|
46.168.186.218| 205.100.250.69|45247|63398|
46.168.98.54| 205.100.241.52|54878| 53|
46.168.99.25| 136.90.166.53|35664| 53|
46.168.104.140| 136.90.166.53|32787| 53|
46.168.161.100| 205.100.250.68|60356|63225|

46.168.105.193| 136.90.242.109|34478| 53|
  46.168.99.3| 193.137.56.227| 5573| 53|
 46.168.96.117| 136.90.166.53|33954| 53|
 46.168.96.123| 136.90.242.109|53261| 53|
 46.168.173.26| 205.100.250.68|48791|63373|
  46.168.98.54| 205.100.241.55|33486| 53|
46.168.107.179| 136.90.242.109|33544| 53|
46.168.217.216| 169.185.74.24| 2048| 0|
 46.168.163.57| 136.90.218.88|51408|64076|
  46.168.98.54| 204.80.31.96|54878| 53|
 46.168.102.56| 136.90.242.109|35258| 53|
 46.168.182.19| 193.137.56.227|21782| 53|
  46.168.99.13| 193.137.56.227|55537| 53|
 46.168.182.20| 205.100.241.52|43040| 53|
 46.168.182.20| 193.137.56.227|39021| 53|
 46.168.182.20| 193.137.56.227|23080| 53|
 46.168.165.76| 141.179.150.77|50886| 2128|
 46.168.201.87| 205.100.241.55|11325| 53|
  46.168.99.12| 205.100.241.52|43524| 53|

Almost all of these are DNS query response messages.

Let's subtract DNS response messages.

[mjscotto@unix36 855]$ bin/rwfilter 200907071400.rw --daddress=46.168.x.x --pass=stdout | bin/rwfilter --input-pipe=stdin --dport=80 --fail=stdout | bin/rwfilter --input-pipe=stdin --dport=53 --fail=stdout | bin/rwfilter --input-pipe=stdin --dport=443 --fail=stdout | bin/rwfilter --input-pipe=stdin --dport=20480 --fail=stdout | bin/rwfilter --input-pipe=stdin --dport=0 --fail=stdout | bin/rwfilter --input-pipe=stdin --dport=771 --fail=stdout | bin/rwfilter --input-pipe=stdin --dport=781 --fail=stdout | bin/rwfilter --input-pipe=stdin --dport=2816 --fail=stdout | bin/rwfilter --input-pipe=stdin --dport=769 --fail=stdout | bin/rwfilter -input-pipe=stdin -sport=53 --fail=stdout | bin/rwsort --fields=dip | bin/rwcut --fields=dip,sip,dport,sport
rwsort: Warning: Using default temporary directory /tmp
          dIP| sIP| dPort|sPort|
 46.168.29.115| 141.169.28.114| 3218| 445|
 46.168.29.115| 141.169.28.114| 3218| 445|
 46.168.29.115| 141.169.28.114| 3218| 445|
  46.168.98.56|141.179.243.125| 25|15817|
 46.168.99.120|141.179.243.125|41229| 25|
 46.168.99.127|141.179.243.125|54188| 25|
 46.168.101.70| 205.100.251.57|51004| 80|

46.168.101.70| 205.100.251.56|58805| 80|
46.168.101.71| 205.100.251.56|49340| 80|
46.168.105.107|141.179.243.125|47967| 25|
46.168.128.166| 205.100.250.69|16443|63463|
46.168.161.70| 205.100.250.68|27491|63001|
46.168.161.100| 205.100.250.68|60356|63225|
46.168.163.57| 136.90.218.88|51408|64076|
46.168.163.57| 136.90.218.88|51408|64076|
46.168.164.144| 205.100.250.68| 7525|62835|
46.168.165.76| 141.179.150.77|50886| 2128|
46.168.166.247|141.179.213.184|60038|58489|
46.168.168.70| 205.100.250.68|12995|61850|
46.168.169.9| 205.100.250.68|17332|62654|
46.168.169.110| 141.169.184.1| 4608| 445|
46.168.169.110| 141.169.184.1| 4608| 445|
46.168.169.110| 141.169.184.1| 4608| 445|
46.168.169.115| 205.100.250.68| 8148|63357|
46.168.173.26| 205.100.250.68|48791|63373|
46.168.176.4|141.179.215.149| 4476| 7508|
46.168.176.191| 205.100.250.69|18667|61646|
46.168.186.218| 205.100.250.69|45247|63398|
46.168.217.216| 169.185.74.24| 2048| 0|
46.168.227.101| 141.179.199.68| 6345|49763|
46.168.227.101| 141.179.16.242| 6345|36735|
46.168.227.101| 141.179.200.6| 6345|65013|

This seems to fall in line with our analysis of web serving clients. The potential clients 46.168.29.115 is below the suspected address space for servers and 46.168.128.x thourgh 46.168.227.x are above the suspected server space. At this point we can be pretty sure that we really do not have enough information to clearly identify the list of clients on our address space.

2.6. ROUTERS AND ICMP MESSAGES

If we want to find out where the routers are on our network we can look for where ICMP messages are coming from. The following command lists all the IP address that sent out an ICMP message

```
[mjscotto@unix11 855]$ bin/rwfilter 200907071400.rw --saddres=46.168.x.x --protocol=1 -
pass=stdout | bin/rwsort --fields=sip | bin/rwcut --fields=sip,dip,dport,sport
rwsort: Warning: Using default temporary directory /tmp
            sIP|            dIP|            dPort|sPort|
   46.168.96.96| 136.90.179.179| 2048|    0|
  46.168.96.117|  136.90.166.53| 2048|    0|
  46.168.96.123| 136.90.242.109| 2048|    0|
  46.168.96.137|  136.90.218.69| 2048|    0|
  46.168.96.142|  141.179.17.61| 2048|    0|
  46.168.96.143|    136.90.5.74| 2048|    0|
  46.168.96.144|141.179.243.122| 2048|    0|
  46.168.96.144|  136.90.129.66| 2048|    0|
  46.168.96.148| 136.90.242.109| 2048|    0|
  46.168.96.150|  136.90.204.49| 2048|    0|
  46.168.96.152|205.100.248.230| 2048|    0|
  46.168.96.153| 136.90.144.144| 2048|    0|
  46.168.96.156|  136.90.245.96| 2048|    0|
  46.168.96.158|  136.90.148.54| 2048|    0|
  46.168.96.161|  136.90.166.53| 2048|    0|
  46.168.96.161|    204.91.2.32| 2048|    0|
  46.168.96.162|  204.80.30.230| 2048|    0|
   46.168.98.60|141.179.243.122| 2048|    0|
    46.168.99.5|  136.90.133.12| 2048|    0|
    46.168.99.6|  136.90.148.54| 2048|    0|
   46.168.99.16|  136.90.204.49| 2048|    0|
   46.168.99.16|  136.90.245.96| 2048|    0|
   46.168.99.25|  136.90.166.53| 2048|    0|
  46.168.101.80|  136.90.108.29| 2048|    0|
  46.168.101.84| 141.179.243.11| 2048|    0|
  46.168.101.84|  136.90.204.49| 2048|    0|
 46.168.101.100|  136.90.133.12| 2048|    0|
  46.168.102.56| 136.90.242.109| 2048|    0|
  46.168.102.56|  141.179.243.8| 2048|    0|
 46.168.104.142|    204.91.2.32| 2048|    0|
  46.168.105.86|  141.5.224.175| 2048|    0|
 46.168.105.119| 169.185.97.183| 2048|    0|
 46.168.105.205| 205.100.241.37| 2048|    0|
 46.168.107.129|  141.5.224.174| 2048|    0|
 46.168.107.161|   204.80.42.67| 2048|    0|
```

46.168.107.178| 169.185.97.183| 2048| 0|
46.168.217.216| 169.185.74.24| 0| 0|

Anyone of these IP address could potentially be a router sending out ICMP messages, but it can just as likely be a client sending out ping messages. Let's subtract echo reply messages.

[mjscotto@unix11 855]$ bin/rwfilter 200907071400.rw --saddres=46.168.x.x --proto
col=1 -pass=stdout | bin/rwfilter --input-pipe=stdin --icmp-code=0 --fail=stdout | bin/rwsort --
fields=sip | bin/rwcut --fields=sip,dip,dport,sport
rwsort: Warning: Using default temporary directory /tmp
            sIP| dIP|dPort|sPort|

Looks like all ICMP messages were echo reply codes

If we had time to live expired codes coming from our subnet then we could tell where the routers were. We probably won't find any…

[mjscotto@unix11 855]$ bin/rwfilter 200907071400.rw --saddress=46.168.x.x --proto=1 --icmp-
type=11 --icmp-code=0 --pass=stdout | bin/rwcut --fields=sip,sport,dip,dport
                        sIP|sPort| dIP|dPort|

And we didn't. We have a list of POTENTIAL routers but we really cannot confirm that any of these are routers.

3. Geographical Location

As stated earlier we did not have a large enough dataset to derive any conclusions on the data over a period of 24 hours. The above graph depicts traffic over a 15 minute period. However, the underlying principle is the same. That is, we try to ascertain the location of this subnet by looking for a rise and fall of traffic. We would then align this rise and fall with the 9 to 5 workday, if the day being examined was a weekday. If it were a weekend, then we would have to employ traffic pattern matching for an average weekend. In all our graphs, the Y axis-to the left represents Megabytes per second, while Y-axis-to the right represents thousands of packets per second. The X-axis represents time in minutes.

Figure 1 is a graph of all incoming and outgoing web traffic on our subnet with the subnet as destination as well as source.

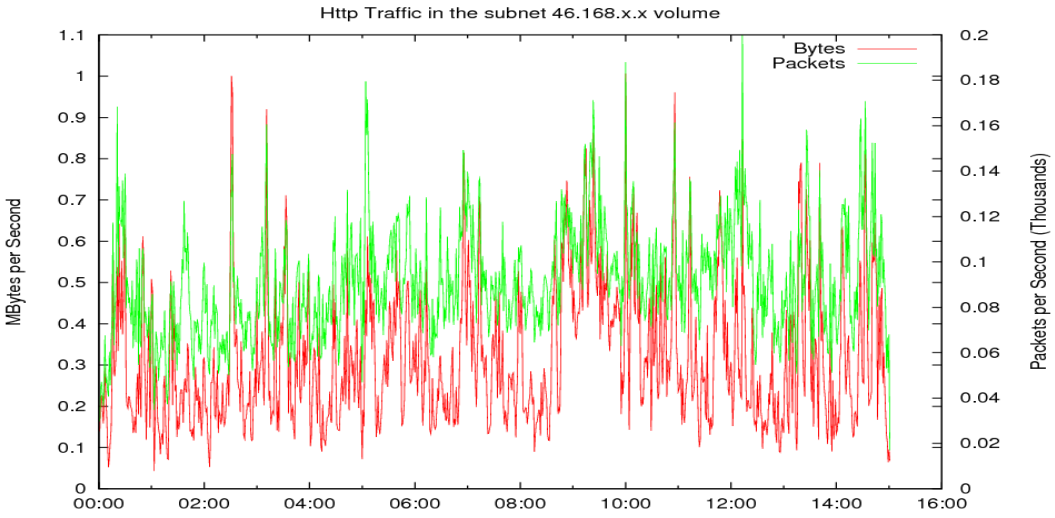

Figure 1

If this had been a while workday, we would look for a pattern. In this case, not seeing one, we would have to dig a little deeper.

In Figure 2, we plotted incoming and outgoing traffic for our subnet only as destination. This gives us a better picture of traffic our servers are receiving, and excludes the responses from our subnet. Again we are unable to see any particular pattern for us to judge what the location of these servers is.

Figure 3 is a plot of just one of our DNS servers (the top talker 46.168.96.143) as the destination server. It again looks like our DNS server is busy throughout the time period queried and no particular pattern is visible.

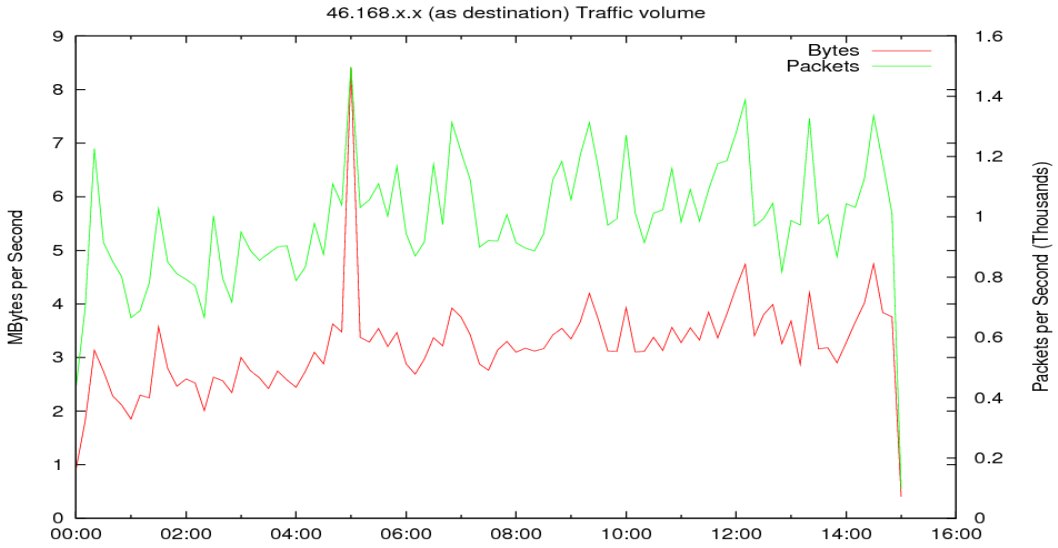

Figure 2

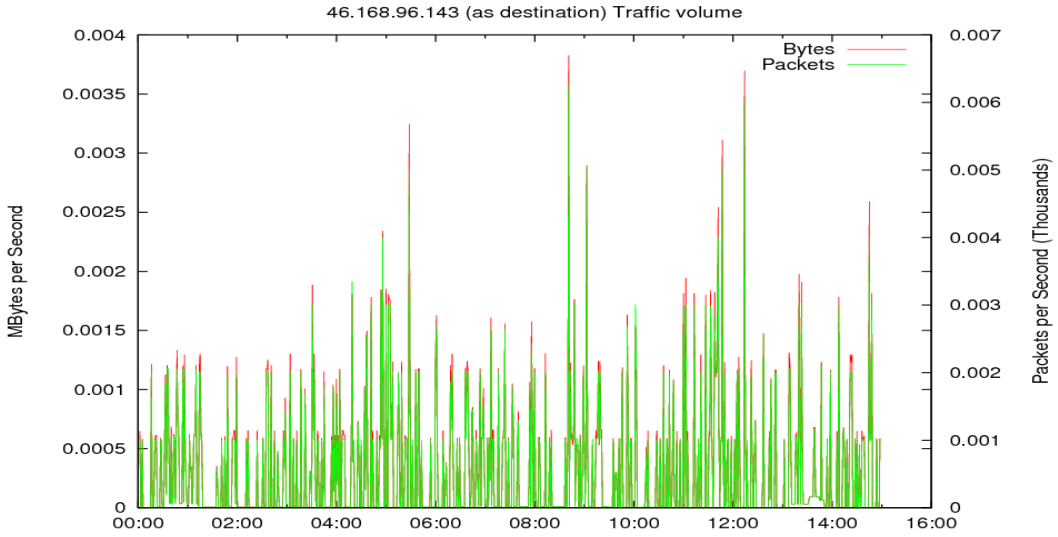

Figure 3

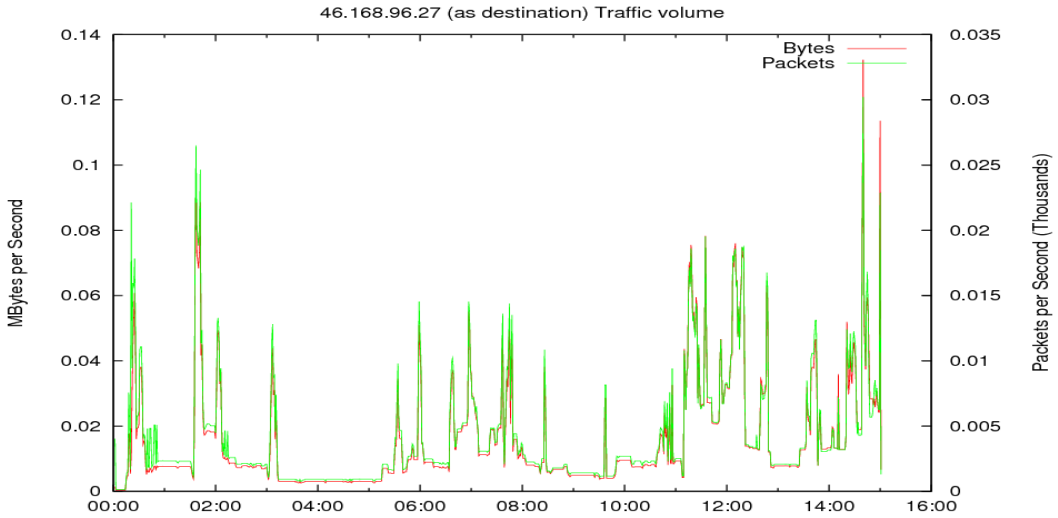

Figure 4

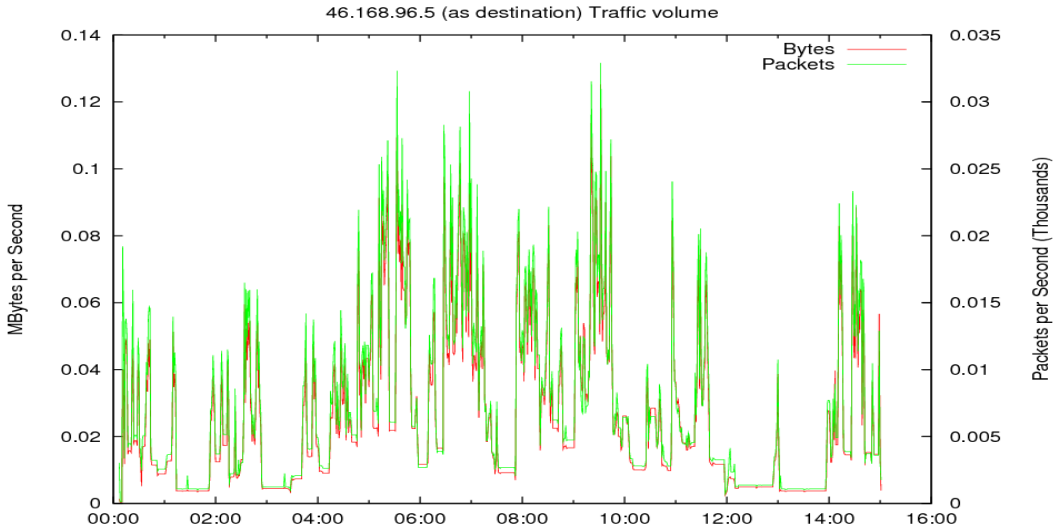

Figure 5

On querying 46.168.96.27, a better idea of the pattern starts to emerge (assuming this is a whole day's dataset instead of just one day). If this had been a 24 hour plot, we could have had been able to determine how far away this server is from GMT, and then would have made guesses to its location. Similarly, we are also able to see such a patter on 46.168.96.5, another top web talker on our subnet—again we can see somewhat of a pattern (assuming this is a whole day's dataset instead of just one day),

and would have been able to determine the geographical location if we had a dataset consisting of 24 hours.

4. Deep Inspection

Ephemeral to ephemeral communication

Earlier analysis showed allowed us to identify communication between ephemeral ports to ephemeral ports. The type of traffic that was particularly interested was on destination port 20480. The source port was in the range of 6000-6500, and the top recipient of this traffic was one of our web server, 46.168.96.5.

Investigating Ephemeral ports ephemeral communication

```
rwfilter 200907071400.rw --daddress=46.168.x.x --sport=1024-65535 --
dport=1024-65535 --print-volume-stat --pass=stdout | rwcut --
fields=sip,dip,sport,dport
```

```
            sIP|            dIP|sPort|dPort|
     |              Recs|           Packets|               Bytes|     Files|
Total|           2609483|          28182301|         18752648387|         1|
 Pass|                50|                54|                6641|          |
 Fail|           2609433|          28182247|         18752641746|          |
  136.90.16.213|    46.168.96.5| 6194|20480|
  136.90.16.213|    46.168.96.5| 6306|20480|
  136.90.16.213|    46.168.96.5| 6378|20480|
```

We attempted to figure out where this traffic was coming from. So we ordered the traffic by bytes and found that this traffic particular traffic was always the same size, 40 bytes.

```
rwfilter 200907071400.rw --daddress=46.168.x.x --dport=20480 --print-volume-
stat --pass=stdout | rwcut --fields=sip,dip,sport,dport,bytes
```

```
            sIP|            dIP|sPort|dPort|     bytes|
     |              Recs|           Packets|               Bytes|     Files|
Total|           2609483|          28182301|         18752648387|         1|
 Pass|                32|                32|                1280|          |
 Fail|           2609451|          28182269|         18752647107|          |
```

```
136.90.16.213|    46.168.96.5| 6194|20480|        40|

136.90.16.213|    46.168.96.5| 6306|20480|        40|

136.90.16.213|    46.168.96.5| 6378|20480|        40|
```

In trying to analyze what sort of traffic this could have been we came up three explanations:

1. NetBUI – was a protocol that allowed older windows machine to speak to windows networks over NETBIOS. It turns out that the NetBUI acknowledgement packets ran on destination ports 20480 and had a total packet size of 40 bytes (20 bytes of which were the TCP payload).

```
TCP: Source port = 1022
TCP: Destination port = 20480
TCP: Sequence number = 137227594
TCP: Acknowledgement number = 1293706515
TCP: Data offset = 20 bytes
TCP: Flags = 0x10
TCP: ..0. .... = No urgent pointer
TCP: ...1 .... = Acknowledgement
TCP: .... 0... = No push
TCP: .... .0.. = No reset
TCP: .... ..0. = No Syn
TCP: .... ...0 = No Fin
```

2. Gaming – multiplayer gaming protocols sometimes use ephemeral ports, and some use port 20480.
3. Malformed packet[1] – potentially this could have been a malformed packet that read in a little endian port number 80 incorrectly as 20480. To illustrate this further:

   Port 80 in hex and binary is 50, and 01010000 respectively.

   Port 20480 in hex and binary is 5000 and 0101000000000000 respectively.

[1] Credit for this suggestion is due to Ron Bandes

5. Architectural Recommendations

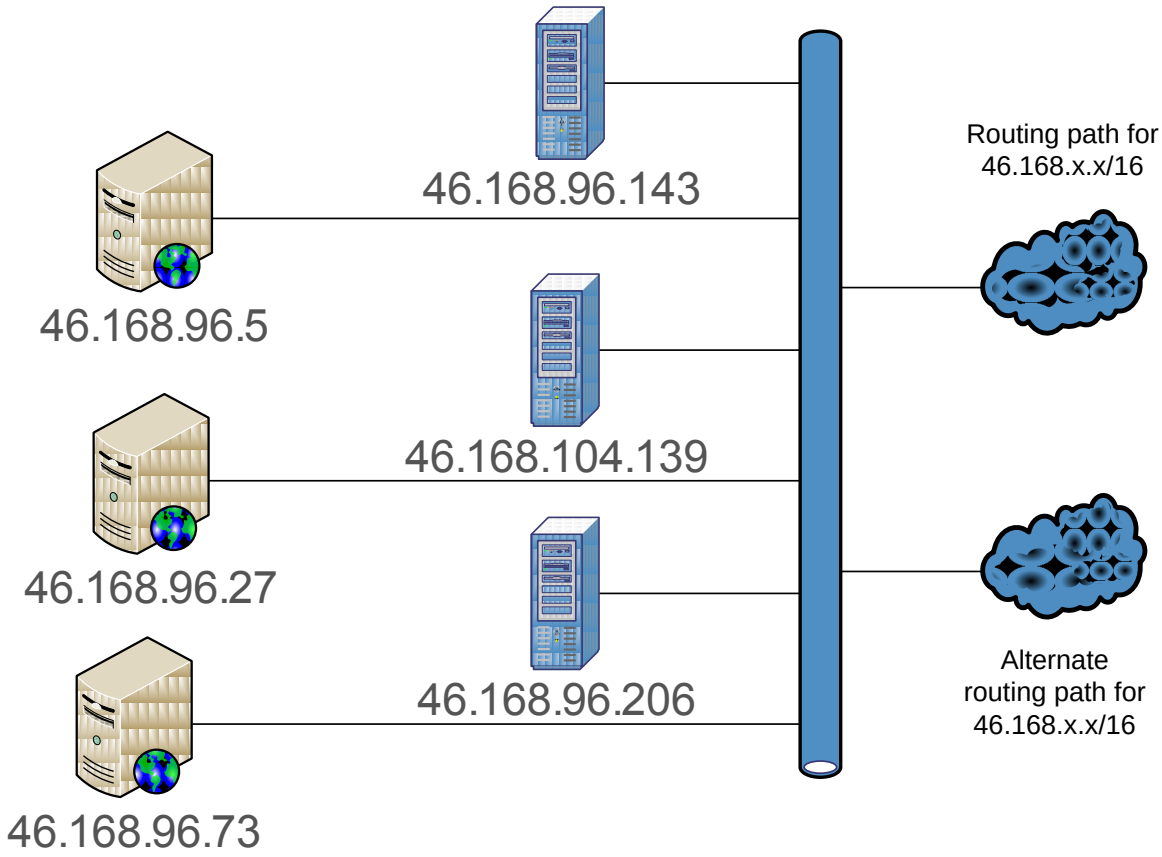

Figure 6

The top talkers on our network are represented in the network diagram shown above. As stated earlier our subnet consists primarily of web and DNS services. 46.168.96.5, 27, and 73, are the top web talkers on our network. Additionally we found that 46.168.96.143, 46.168.96.206, and 46.168.96.206 are the top DNS speakers on the subnet in question.

We assume for the purpose of this exercise that this was an ISP with the web and DNS servers belonging to the clients on this ISP. This is a suitable assumption since we observed numerous web and DNS service providers. Furthermore, we also assume that these servers are not geographically collocated and the subnet is spread over a large geographical area, within a single country (such as Russia or China).

We know from or analysis that there were multiple incoming/outgoing paths from this network. This conclusion was based on our observation that we were seeing web traffic coming into this presumed ISP, but did not see any going out. It would then be reasonable to assume that there were multiple paths in and out of the network and the routing infrastructure going out for web traffic was different from the routing architecture coming in.

An ISP needs to be Scalable to provide for a large number of requests, and needs to have enough capacity for future growth. It needs to be reliable and have redundancy built into the network. It needs to have high availability. And, finally, it needs to meet customer expectations in that a completely locked down service provider would be great for security but would not be profitable if the customers are not able to meet their needs.

Our recommendation takes into account the above factors and makes recommendation on the asymmetric routing we observed in the ISP architecture.

What is asymmetric routing? Asymmetric routing occurs if the outgoing traffic leaves the network from a different route than the incoming traffic. So looking at our potential network diagram, traffic from web clients leave the top routing path but return on the alternate routing path. This can cause issues with some sensitive server applications, perimeter NAT devices that require state information, etc.

A network like ours would most likely be using some sort of firewall or proxy infrastructure. This can be problematic on redundant internet links. Let us assume that one proxy server for a client on the ISP's network receives traffic from one internet link, and the redundant proxy/firewall receives traffic from the other link. This asymmetric routing was cause the latter to drop the packets because it received an out-of-state traffic.

We would therefore recommend that this ISP ensure that any given client does not experience dropped packets because of asymmetric routing. This would ensure the earlier stated goals of an ISP are fulfilled.

6. An Internet-wide event and its effects on our network

DNS fast-flux

In July of 2007 reports came to surface of cybercriminals increasingly using fast-flux DNS to hide and sustain infrastructures. The creators of the malware, Storm have also moved their infrastructures to fast-flux service networks, according to the Honeynet Project & Research Alliance[2].

Fast-flux is basically used in the form of load-balancing for hosting malware infrastructure. It's a round-robin method where infected bot machines work as web servers for malicious content. The hosts that are part of this infrastructure is constantly rotated, changing their DNS records to prevent discovery. That is, they are able to bypass black-listing based schemes and are eventually able to serve their malicious content for longer.

Looking for evidence of criminal activity

Also in July 2007, the same time period we were analyzing our traffic, researchers uncovered a turf war between criminal rings. The creators of Storm were one of the group active in this period. We wanted to see if this cybercriminal ring was active in our sample traffic files. If we did find evidence, we could more aptly answer this question.

[2] http://www.darkreading.com/security/perimeter/showArticle.jhtml?articleID=208804630

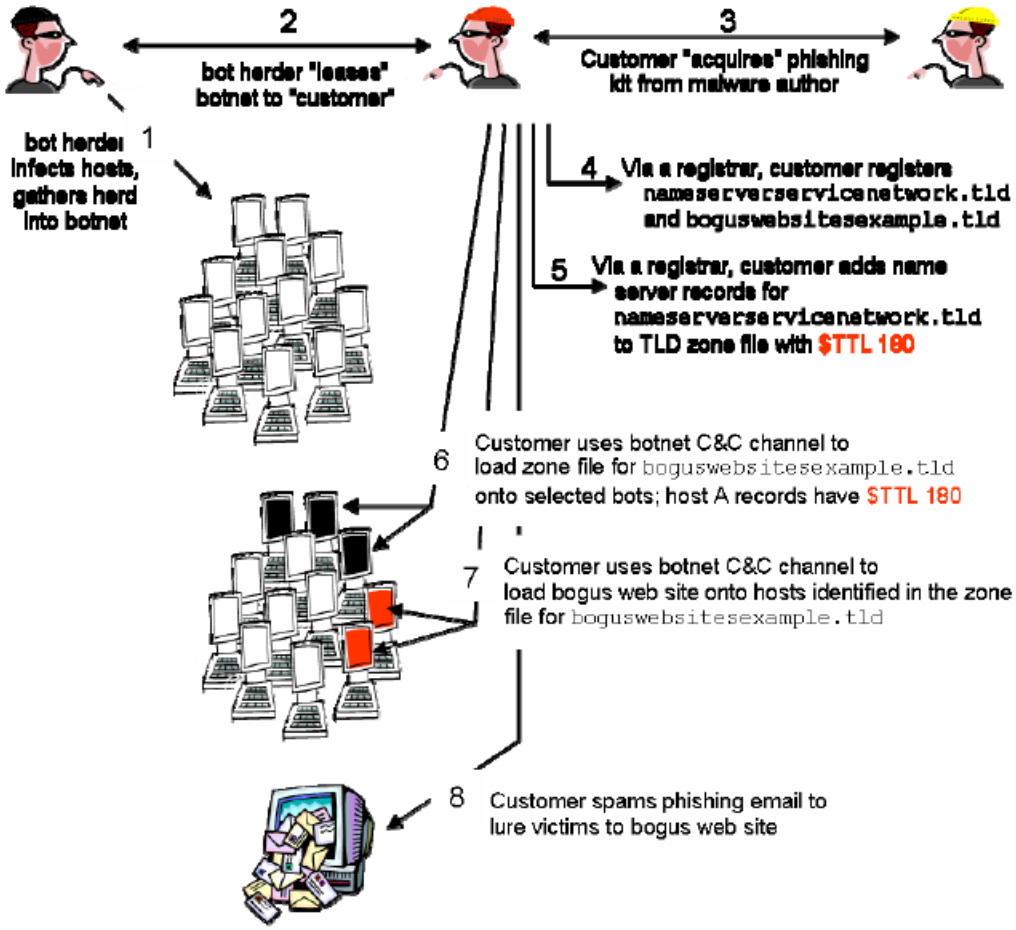

Figure 7 - Fast and double flux attacks[3]

We decided to use Snort IDS to see if there was any evidence of malware being delivered to machines in our network.

We did not see any evidence of traffic from the creators of Storm and related malware. We then checked the whole dataset to see if there was any interesting alert. We discovered the following priority-1 Snort alerts:

```
[**] [1:2182:8] BACKDOOR typot trojan traffic [**]

[Classification: A Network Trojan was detected] [Priority: 1]

07/07-00:00:04.434072 132.198.169.98:59272 -> 204.80.14.47:15338

TCP TTL:117 TOS:0x0 ID:10823 IpLen:20 DgmLen:52
```

[3] Taken from http://www.icann.org/en/committees/security/sac025.pdf

```
******S* Seq: 0x3923C3FE  Ack: 0x0  Win: 0xDA00  TcpLen: 32

TCP Options (6) => MSS: 1460 NOP WS: 2 NOP NOP SackOK

[Xref => http://vil.nai.com/vil/content/v_100406]
```

> Stumbler (also known as Typot and 55808) is a trojan horse that uses a stealth scanning technique to scan and receive network mapping data. Stumbler begins network mapping by sending TCP SYN packets with a TCP window size of 55808. The source IP address and the source MAC address for each packet are spoofed, making it impossible to identify the source by examining the packet. Stumbler causes each infected system to enter promiscuous mode, using a promiscuous mode packet sniffing tool to listen to all responses of spoofed SYN packets from all other Stumbler agent peers.
>
> Stumbler seems to harvest and send network mapping information to a specific hardcoded IP address. If Stumbler detects that network connectivity has been interrupted, it will exit and attempt to delete itself from the file system.
>
> This trojan has currently only been observed on Linux systems, but its variants could be easily ported to other Unix or Windows-based platforms.
>
> Stumbler's current scanning capabilities are used for information gathering. The trojan horse program has no means to propagate automatically, or to scan for and infect other hosts[4].

We could not find any more evidence of whether this was related to the cybercriminal ring in question. However, it is possible that such criminal activity was being carried out on this network and that these groups were employing fast-flux DNS to keep their malicious content providers alive.

Potential impact on our networks

DNS servers were one of the major pieces of infrastructures found on our network. This means that our DNS service providers could have easily been a victim of fast-fluxing to keep malicious content pages alive.

Keeping this in mind our ISP should carry out measures to block DNS fast-flux. Such DNS requests should be probed by employing intrusion detection systems and anti-malware gateways. We should then block TCP port 80 and UDP port 53 so that from well-known and/or black-listed networks.

Our ISP should then employ similar mechanisms on hosts in its network to allow detection of infected hosts. One such hosts are detected, they should be immediately taken offline until resolution.

[4] Taken verbatim from http://xforce.iss.net/xforce/xfdb/12376