A Network Flow Analysis of One Anonymous Class B Network

Michael Hanley, Brent Kennedy, Devon Rollins

Graduate Students, Information Security Policy and Management

95-855 Network Situational Awareness – Fall 2009

Carnegie Mellon University

Professors: Tim Shimeall and Sid Faber

Introduction

On March 31, 2009 starting at approximately 1500 hours GMT, the MAWI Working Group cooperated with CAIDA, The Cooperative Association for Internet Data Analysis, to conduct a large-scale internet data collection project. The MAWI Working Group contributed by sampling a trans-Pacific link using tcpdump[1] to collect packet capture data and then annonimyzing and truncating the data using tcpdpriv[2]. This data appears to us to be in a network block-preserving anonymized state, provides an incredibly valuable tool for students (undergraduate and graduate, alike) to perform traffic analysis both at the granular packet level, and with the appropriate tools, at a net flow level. For the purposes of this paper, 24 hours worth of packet capture (pcap) data from sample point "F" was converted to net flow data and packed into files compatible with the SiLK suite, developed at the CERT Coordination Center at Carnegie Mellon University's Software Engineering Institute. We have been asked to analyze one /16 network from this dataset of our choice. The team has elected to analyze the 193.52.0.0/16 network, partially because of it's high volume, and partially because of

[1] tcpdump is a highly versatile command line interface for collecting packets on a computer network but using a network card in "promiscuous mode" in order to see all traffic passing by the interface, not just traffic destined for that interface. There are some graphical tools that have similar capabilities, such as Wireshark (wireshark.org) that users may be more framiliar with, but tcpdump's command line only approach makes it more suitable for this type of collection.

[2] Tcpdpriv (http://ita.ee.lbl.gov/html/contrib/tcpdpriv.html) was developed to eliminate confidential information from packet capture data and also to anonymize network addresses in the packet capture file such that relationships between real IPs are preserved, but the octets themselves are randomized to make geolocation or other identification strategies fruitless. The team that wrote this article made no effort to reverse-engineer this anonymization.

interesting traffic patterns which will be discussed later in this paper. Detailed analysis of other CIDR blocks is outside the scope of this paper.

It is worth noting that the team treated this exercise as if the 193.52.0.0/16 class-B network were a real network block. The fact that the data has been partially scrambled to provide some anonymity is irrelevant for the purposes of this exercise. This is still real data that traversed an undersea transit link several months ago, so we treated it as such. Thus, we made no efforts to reverse-engineer any of the scrambling done by tcpdpriv, nor did we attempt to extract or derive any personal or confidential data from the net flows or the packet capture from which the net flow data was derived. In the spirit of the exercise, and to show the educational benefits of this type of anonymous flow trace data, we treated this data is if it were live.

Tools

The purpose of this exercise was, in large part, to do a flow-based analysis of this 24 hour traffic window. That said, our toolset consisted largely of the SiLK tools (System of Internet Level Knowledge) engineered and maintained by the Network Situational Awareness group at CERT. These tools provide the means to create complex queries, filter sets, and analysis of flow data at high speed. The team was able to use the SiLK tools to parse approximately 3 GB of flow traffic on a moderately-powered machine with little delay. The key tool involved in our analysis was rwfilter for extracting information from the flow files related to our particular /16 network, as well as for filtering out service ports of interest, inbound vs. outbound traffic, and other characteristics. Data filtered with rwfilter was piped to a multitude of tools, including rwcount, rwstats, and rwscan. Information about these and the rest of the SiLK tools can be found at http://tools.netsa.cert.org/silk/silk_docs.html.

Aside from SiLK, we used two other primary tools for our analysis. First, as the reader will note throughout the remainder of this paper, we used gnuplot for some simple graphs of traffic volume over time. This is useful for demonstrating our /16's usage of the network at various points during the day. Second, we used Excel to aggregate and manipulate some of our output from SiLK. Graphs from Excel and some of the statistics we derived using our Excel spreadsheets also appear throughout the remainder of the paper.

Understanding Flow versus Packet Capture

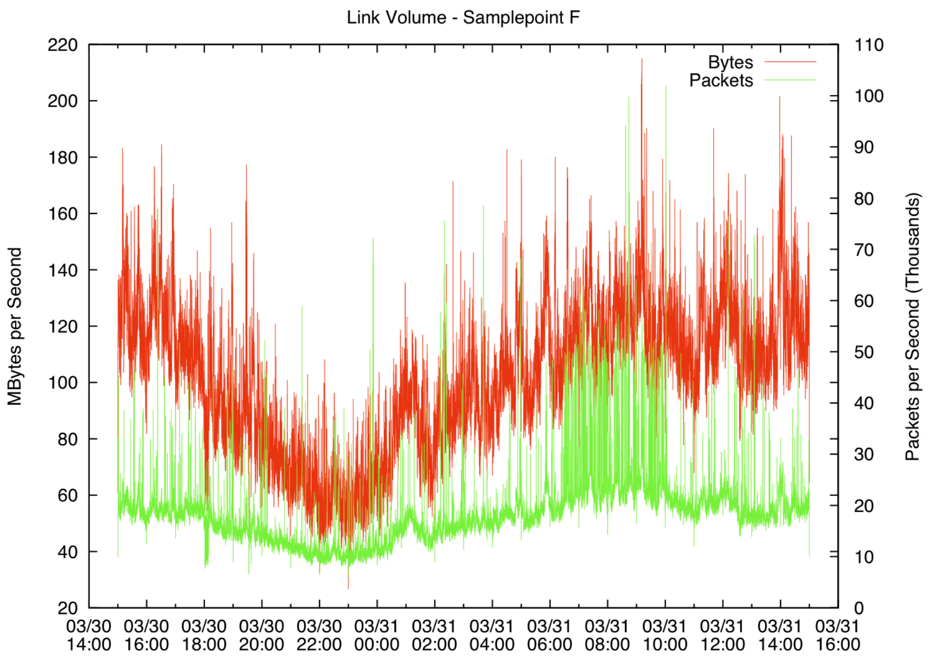
A key distinction to note is that our analysis is attempting to achieve a degree of situational awareness with respect to our /16 network's traffic on this transit link. We seek to understand issues like the following:

- What kind of services appear to be hosted on our network and where?
- Who are the clients outside of our network that are using our services?
- What threats does our network face?
- What external services are our clients accessing?
- How much of this link's utilization is associated with our network?

While this is not an all-inclusive list, these are just some of the questions we seek to answer. Further, we are not dealing with packet capture, we are dealing with flow. Thus, we are dealing with summary records of network transactions, not individual packets. Thus our records consist of transaction information such as source IP, destination IP, source port, destination port, protocol, the number of packets in the flow, the number of bytes transferred, flags, start time, end time, and flow duration. This data is immensely useful for tracking communication volume as a whole as well as volumes associated with specific services.

General Link Findings

Before we can talk about what we find in the 193.52.0.0/16 network, we need to provide some context. We would be remiss in our analysis if we did not first provide a short overview of the link on which the data was captured, and what sort of broad trends were observed on the day of the capture across all networks traversing the link. First, a look at the total link volume as extracted from the flows with SiLK and plotted with gnuplot.



There are a few obvious trends to observe here. First, the link as a whole is keeping a fairly steady rate of packets/second until a volatile period between 0600 and 1000 on the 31st. There is a corresponding volatile region in the MB/second graph, and we do find later in this article that our netblock had some role in this spike. The byte volume trend also follows a fairly smooth trend downward from 1400 to 2300 on the 30th, and increasing again from around 0000 hours on the 31st up through 1400hours on the 31st. There is a non-trivial spike at roughly 0100 hours in byte volume.

The composition of services hosted and accessed via this link is fairly unremarkable. In terms of byte volume, HTTP leads the way as almost 63.2% of our source traffic. The nearest service in terms of byte volume is SSH, with 2.3% of the volume by bytes, followed by DNS with close to 1.9% of traffic. If we count by packets, we find that ICMP traffic moves up to take the second place spot from SSH, but this is merely a different way of representing the data and still is not an alarming finding.

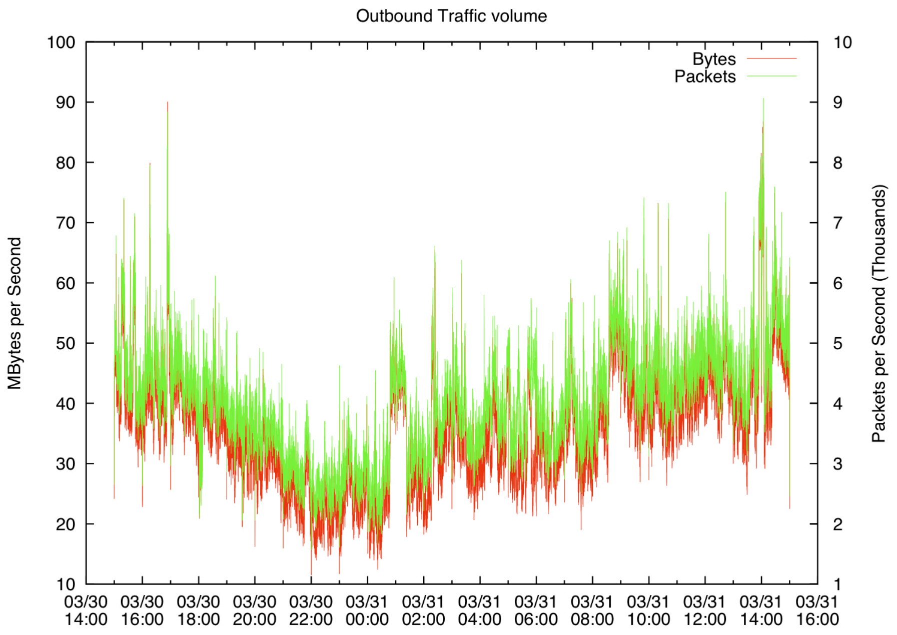
General Findings – 193.52/16

There are some interesting general findings to be mentioned about our netblock. In terms of overall volume, we find that our netblock accounts for a significant amount of the traffic moving on this link. Specifically, we find that over the 24 hour period, our /16 accounts for over 35% of the traffic volume with 381 gigabytes of traffic volume served. In terms of inbound traffic to our network, the number is not as impressive, but is still significant at about 4% of the total traffic by byte count. This puts our network at close to 40% of all traffic running on the link during this time frame. Complete statistics are shown below:

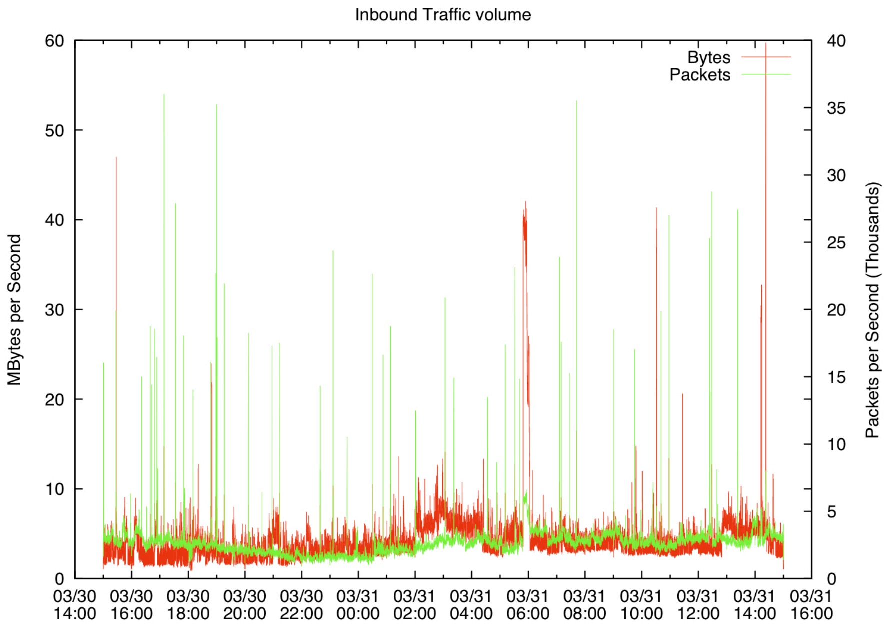
```
$ rwfilter ext*/* --saddr=193.52.0.0/16 --print-vol     |              Recs|
Packets|               Bytes|     Files|
Total|         168947242|        1543854447|       1083477010728|        24|
 Pass|          16661024|         344913772|        381322829661|          |
 Fail|         152286218|        1198940675|        702154181067|          |
$ rwfilter ext*/* --daddr=193.52.0.0/16 --print-vol
     |              Recs|           Packets|               Bytes|     Files|
Total|         168947242|        1543854447|       1083477010728|        24|
 Pass|          19347793|         224218579|         43942603806|          |
 Fail|         149599449|        1319635868|       1039534406922|          |
```

If we map out this traffic in granular detail with gnuplot, we find that our block's graph holds the general shape of the overall link volume graph in terms of our served traffic (where the source address is within our netblock), which seems intuitive given that a large portion of the link's traffic is from our address space. For traffic that was inbound to our address space, the graph is actually relatively flat. Comparing these two suggests that out network space might have some very large content providers residing on it, and that the user population on our netblock is probably small.

First the outbound traffic:



Then the inbound traffic:



Another finding of interest is that SiLK returns no results when we query for traffic where our block is both the source and destination of the traffic, which might represent internal traffic in that address space. Because this is a transit link, and the scrubbed data is network-preserving, this tells us that we are really only seeing what is coming in and out of the 193.52/16 enclave, not what is happening within it. Thus, there's little we can say about the actual contents of our address space, and it's inner workings, however the data still lends well to an analysis of what is hosted on the network and what external users are coming to us for, and that discussion is continued in the following sections.

A final general finding, which will be analyzed later in detail is that our network's large portion of the link traffic seems to stem from some very large web servers resident on our network. Some simple commands in SiLK immediately reveal this finding:

```
$ rwfilter <SiLK volume> --sport=80 --print-vol
     |              Recs|           Packets|               Bytes|     Files|
Total|         168947242|        1543854447|       1083477010728|        24|
 Pass|           7143132|         507528106|        684461175484|          |
 Fail|         161804110|        1036326341|        399015835244|          |
$ rwfilter <SiLK volume> --saddr=193.52.0.0/16 --sport=80 --print-vol
     |              Recs|           Packets|               Bytes|     Files|
Total|         168947242|        1543854447|       1083477010728|        24|
 Pass|           2282677|         217711963|        298022441713|          |
 Fail|         166664565|        1326142484|        785454569015|          |
```

Right away we see that in terms of records, packets, and bytes, we are serving 31.9%, 42.9%, and 43.5% respectively. If we dive deeper to see who's hosting this traffic, and if it is isolated to a few hosts, we find the following:

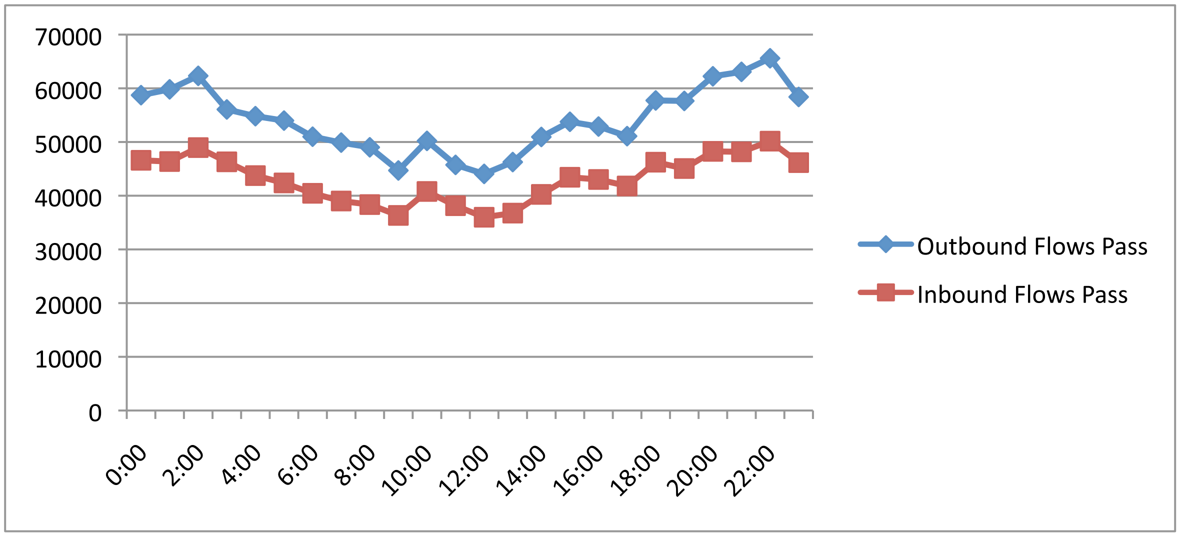
Services, spikes

To achieve true situational awareness, it is important to get a full view of your network and understand the "norm" in order to diagnose anomalies. To do this, common services will be looked at on an individual basis. Being that the netblock is so large, the services that are looked at have been narrowed down based on those that are classically popular and the ones being utilized the most on the netblock. DNS, HTTP, HTTPS, SMTP, and FTP will be looked at in a more granular fashion. Also, port 1063 will be analyzed since it is showing up as the 4th most used port in the netblock.

Services

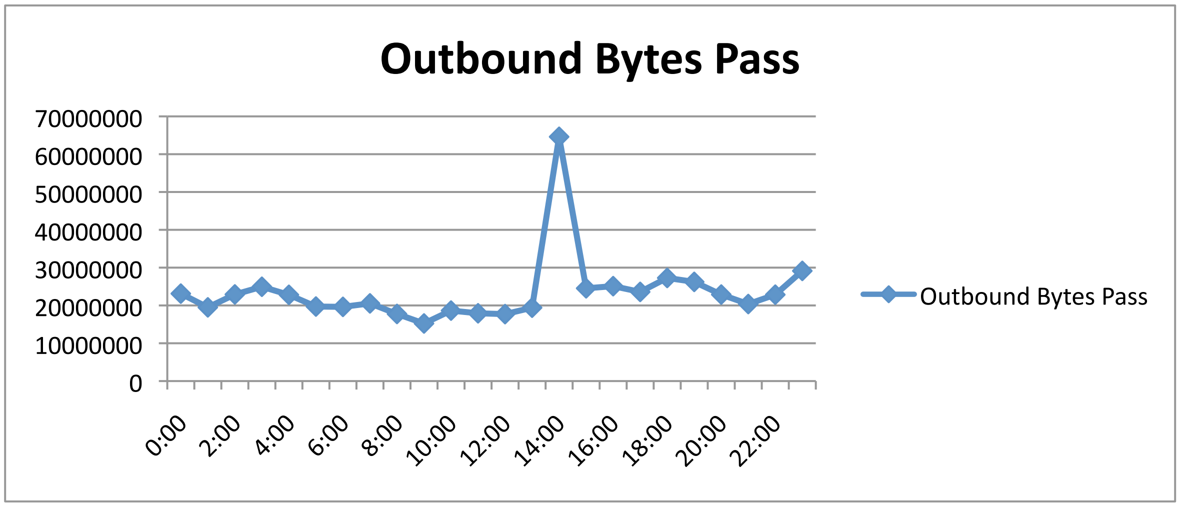
DNS

As expected, there is a great deal of DNS traffic originating and going to the netblock. This services appears to be acting normally though. Looking at the overall traffic graph shown previously, you can get a general idea of the traffic shape throughout the day. The graph below shows the DNS inbound and outbound flows throughout the day and it can easily be seen that it matches up will with the overall traffic trend.

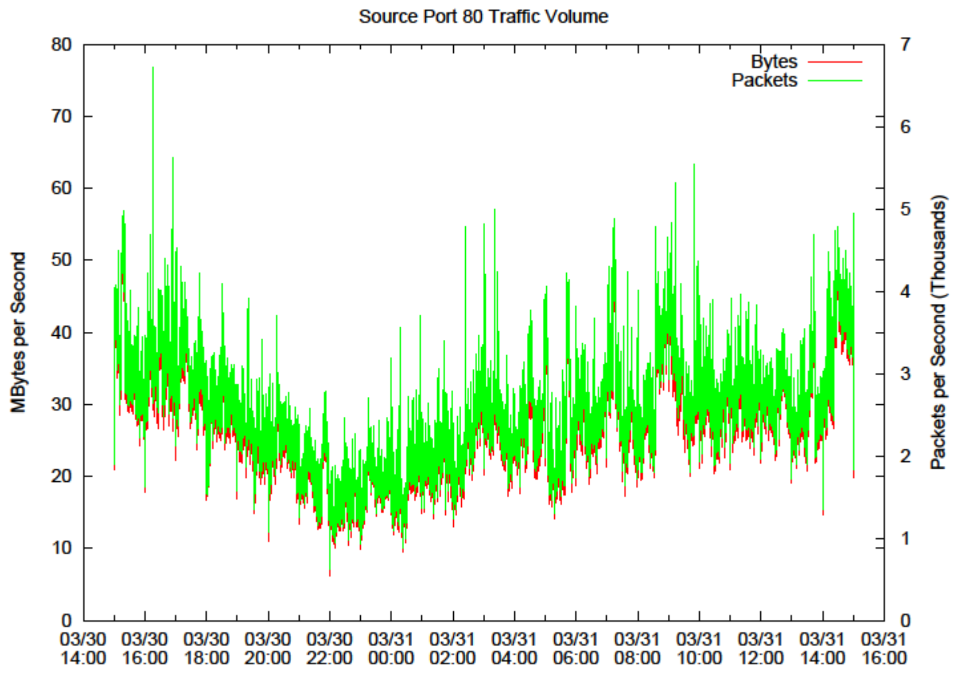


HTTP

Web traffic is obviously going to be a much utilized service in almost any network. This traffic can take many forms, but we will be looking specifically at port 80 in regards to HTTP. The graph below shows the outbound packets across the day.

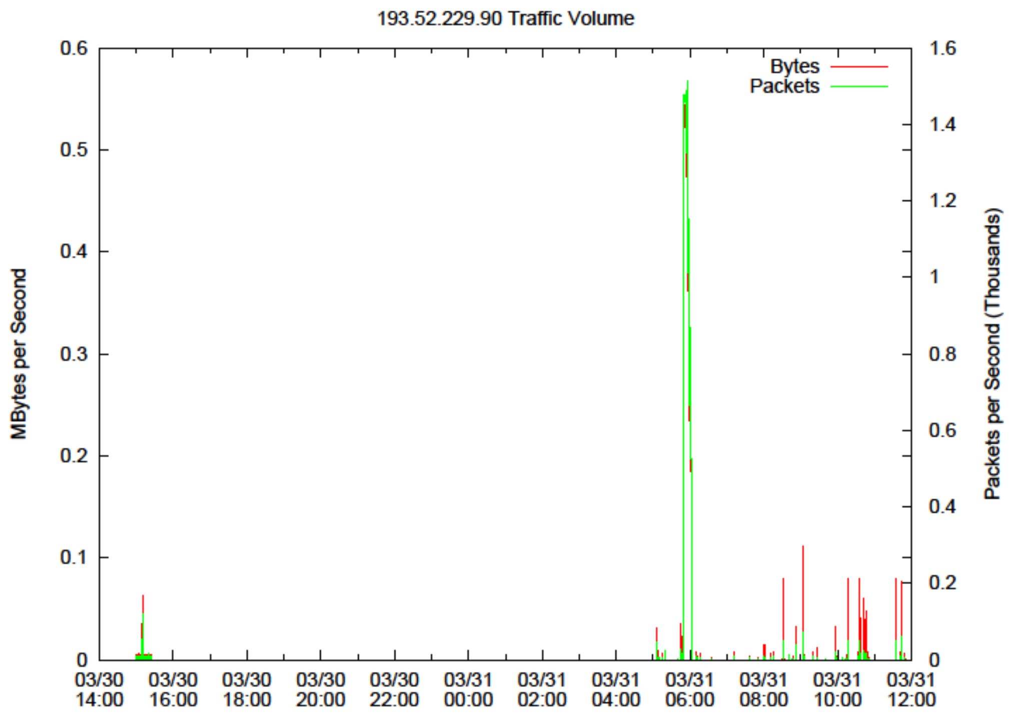


The traffic is consistent throughout the day asides from a very noticeable spike around 14:00 mark. This is seen again by the GNUplot graph below.



Since this spike is not normal, it is worthwhile to dive a little deeper to figure out its cause. There is two ways that HTTP traffic can occur with our netblock acting as the source addresses. The first is traffic with a source port of 80 which means that web servers exist within our netblock and are responding to requests. A great deal of the HTTP seems to be due to web servers and specifically two web servers (193.52.237.154 and 193.52.237.155) account for 55% of the total outbound traffic. Analysis shows though that traffic from these web servers is consistent throughout the day. So, the spike must be due to outbound traffic that is initiating web requests and thus has a destination port of 80.

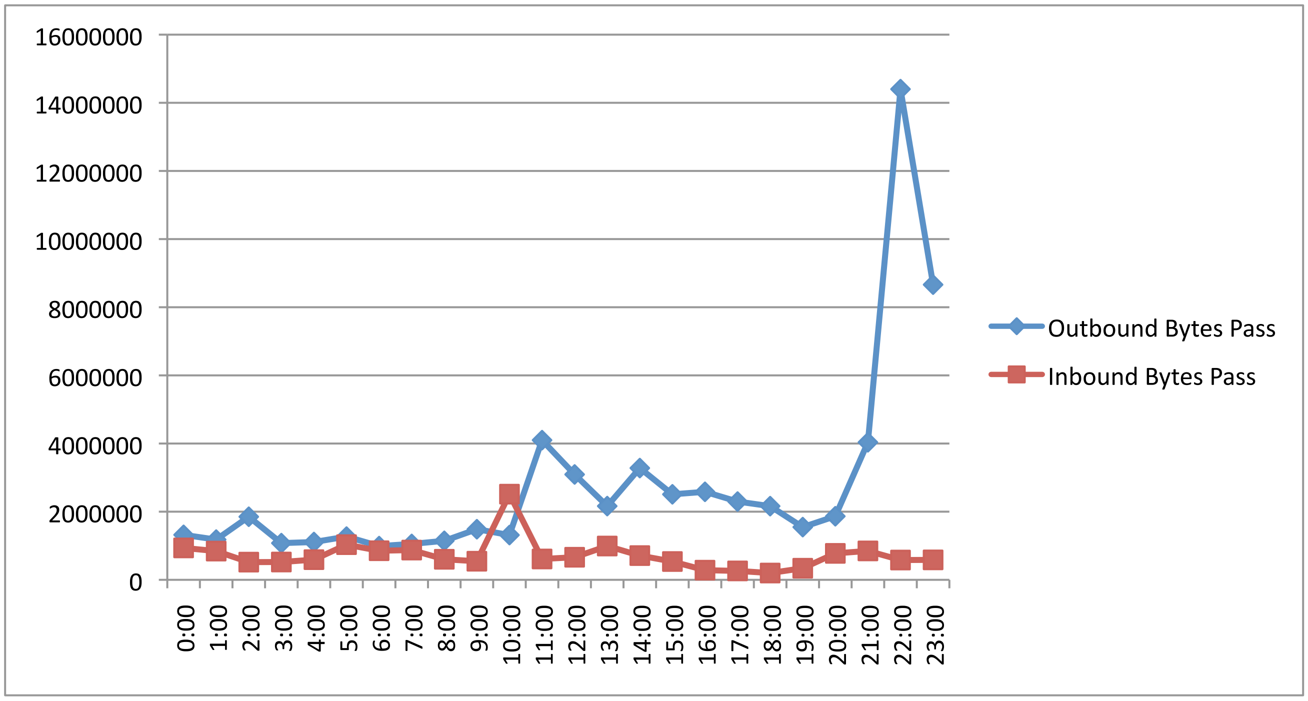
It turns out that this is exactly the case. SiLK was used to show the top IP addresses based on byte count on a per hour basis (see Appendix A). There is a very obvious jump in bytes for 2 hosts, 193.52.229.90 and 193.53.232.189, that occur at the point in question. The former host is responsible for over 35 million bytes which is about 30 million more than normal. Its traffic plot can be seen below. What's more interesting is that this host did not show up in the top 10 talkers in any other data file (hour). It is still listed in the following data file as the peak decreases, but it then disappears for the remainder of the day. The latter host listed above had around 10 million packets but was a consistent top talker throughout the day. It returned to its normal byte count in the following hours.



Taking the top talker (193.52.229.90) it is beneficial to see who that host is communicating with to cause the increase in traffic. The results (see Appendix B) show that the host made some connections early in the day and then was dormant for several hours. It then talks a great deal towards the middle of the day which is where the spike occurs. Afterwards, it was mentioned above that the host goes out of the top 10 list which is true, but further analysis revealed that it still continues to communicate after the spike. The output shows that it actually communicated with a specific set of hosts and sent the same fixed amount of bytes to each of them. It is hard to explain the spike and this fixed byte behavior without seeing the payload, but it seems as though the host spent a great deal of resources searching for other hosts during the spike and then continued to talk to specific ones after the fact. One explanation of this could be a botnet in that the fixed traffic is instructions or responses that are being sent. This would be consistent with the fact that many present botnets are using HTTP to communicate as opposed to the outdated IRC channels. Once again though, without the payload information it is very hard to tell what is specifically causing the increase in traffic.

HTTPS

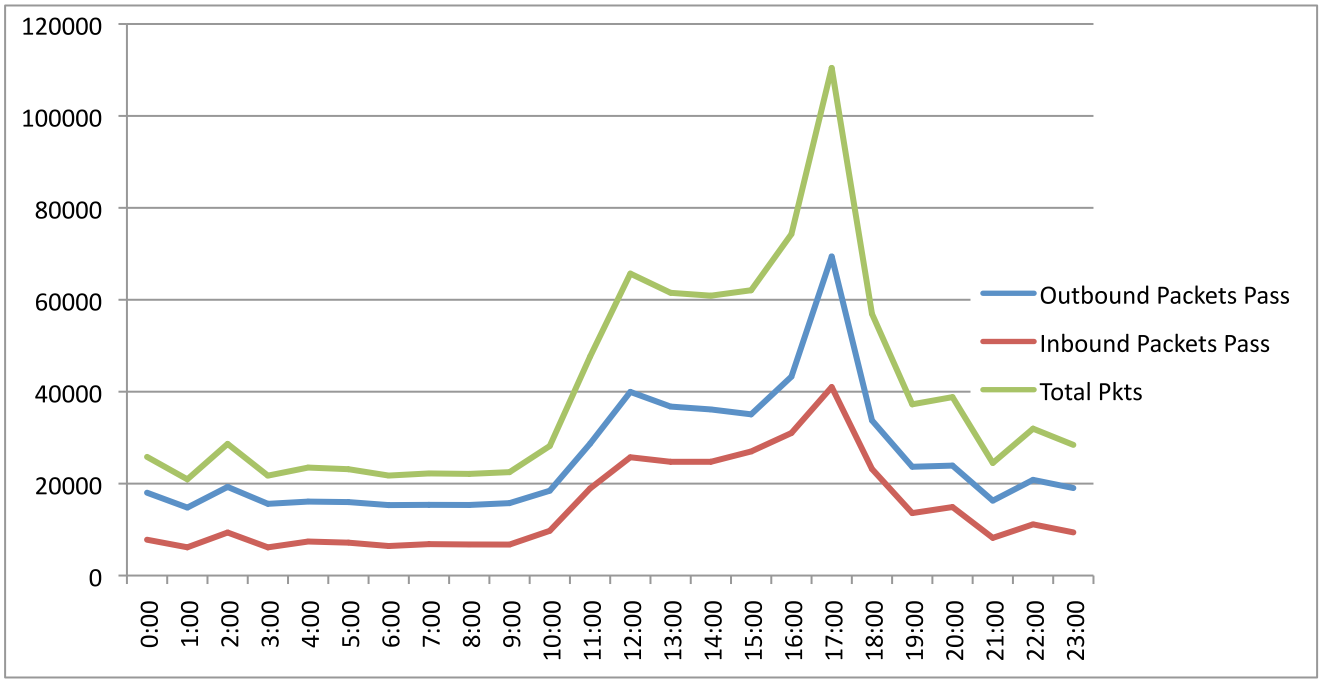
Like HTTP traffic, HTTPS traffic is equally important to analyze. The initial interesting observation is that on average there is more HTTPS traffic than HTTP traffic. This may be due to the fact that many websites are switching to HTTPS for the added security. Looking at the bytes over time graph (see below) it is seen that the traffic is fairly consistent over the day until a large spike at the end. It is also apparent that there is more outbound traffic than inbound and only the outbound traffic had a spike.



The same methodology to determine the HTTP spike was used here. Also like the HTTP spike, this spike proved to be due to outgoing traffic except with a destination port of 443. Traffic with a source port of 443 shows consistency during the time of the spike, but it was the destination port traffic that shows where the spike is coming from. The results of the flow analysis (see Appendix C) shows that the second to last data file contains a host with around 10 million more bytes than normal. Diving deeper it is seen (see Appendix D) that this host is talking to only 3 other hosts during the hour of the spike and during the hour before and after. The communication may continue longer but there is no data to find this out for certain. During all three hours, the top destination IP (47.89.25.106) stays the same and during the peak and the hour before this communication accounted for 99% of the traffic from that host. This is an awful lot of HTTPS traffic that is going on. One potential explanation is the secure transfer of a very large file. Even with the payload information though, the file would not be known since it is done securely.

SMTP

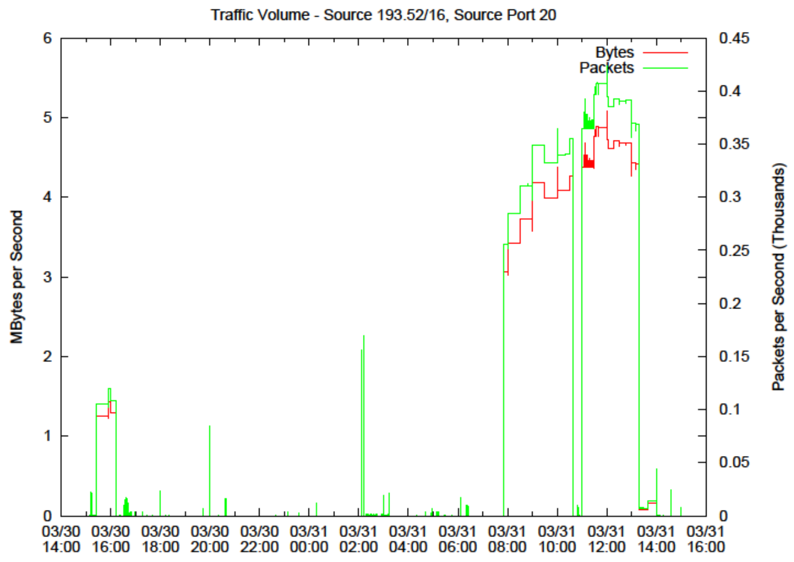
A great deal of situational awareness is composed of web traffic, but email traffic is also very important as it can be a common tool for attacks. As a service, SMTP traffic (port 25) was seen as one of the most popular on our netblock. The graph below shows the inbound and outbound traffic across the day.



Per the graph, you can see a gradual incline with a spike and then a gradual decline. This is actually very normal traffic and is explained by the high use of email to conduct business. The start of the incline and the end of the decline (approx. 10:00 – 19:00) signifies the beginning and end of business hours. With this normal trend of traffic, no further analysis was needed to detect any sort of malicious behavior.

FTP

The File Transfer Protocol can be a popular service on networks and the netblock being analyzed is no exception as it was the second most popular port (20) being used. Looking at some initial flow analysis, it seemed as though that FTP had a minor presence since the number of flows were minimal. The small amount of flows though was made up by the amount of data each one represented. The GNUplot seen below shows these facts.

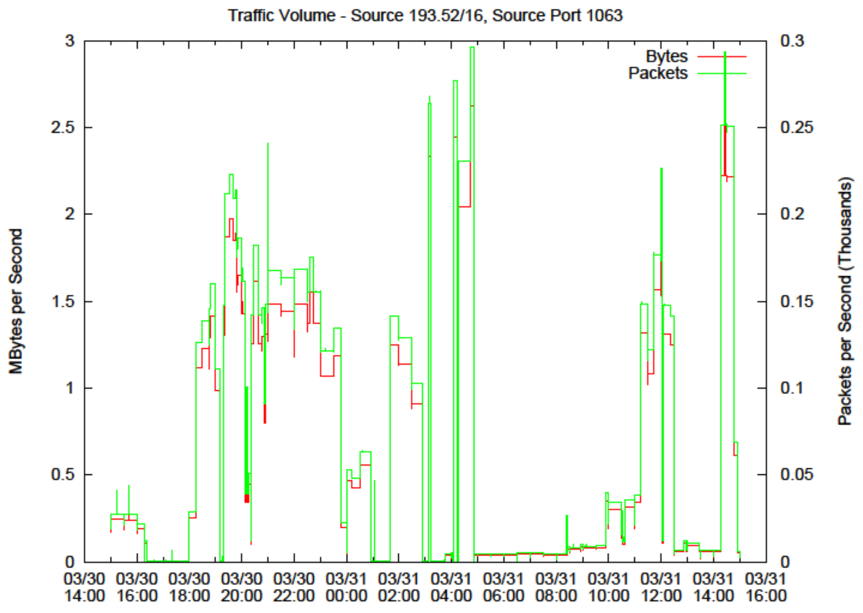


Looking a little bit closer at the data, there are only a select amount of hosts that are doing the FTP talking. The common hosts are 193.52.237.155, 193.52.229.233, 193.52.224.3, 193.52.230.15, and 193.52.237.154. On the other side of the connection, the destination addresses do not seem to have any trend and are somewhat random. It seems as though the

FTP traffic is typical. It is possible that malicious files are being transported back in forth but that cannot be determined with this analysis. The traffic pattern is a bit odd with its “step” pattern, but this can be explained several ways. The actual step is due to the fact that many of the file transfers span several hours. This is scene by seeing the same source IP and destination IP across several files (hours) and then never seen again. The graph also shows that the “steps” all occur in a several hour window. This could just be due to the fact that it was normal business hours and FTP is a useful tool used in many daily operations.

Port 1063

Interestingly enough, a port showed up on the top 10 list that was not identifiable. After doing some research on port 1063, the most common finding was that it is common to the KyoceraNetDev service. However there seems to be very little documentation on this service so it is still unknown what it is used for. It is also possible that this service is outdated and the port is being used for another purpose. A plot of the inbound traffic using the port can be seen below.



The plot shows how erratic the port use is. Like the FTP traffic, the flows are "table" like which shows that each use is occurring over a longer period of time. Unlike the FTP port though, this service is being used all throughout the day with no real trend. It could be possible that this port is being used as another proxy port, but it is not common to that type of service. Many different services can be configured to use whatever port they like but without seeing the payloads it's hard to tell what it's being used for. Using more flow analysis, it shows that only a couple hosts are using the port with the most common being 193.52.252.40. There are also a few destination addresses that are don't seem to follow any pattern. Overall, the flow analysis matches up with the plot in that there are only a few connections that span over a couple hours each with a large amount of data being transferred.

Threats

The collection of network flow data is optimal for discovering threats to large-scale networks as attacks follow certain behavioral patterns that can be detected through known methods and a deep understanding of normal network activity. The sheer volume of data can be partitioned to dig deeper into anomalies that may signal malicious events. Any findings can be correlated with system log files, firewall logs, and intrusion detection alerts to determine how effective the security layers of an organization are against active and passive adversaries. Furthermore, an organization can enumerate the type of attacks that are targeted to their network and develop strategy to respond accordingly.

Our analysis of the MAWI netblock 193.52.0.0/16 disclosed varying techniques used to perform reconnaissance on the netblock. Scanning is a technique highly regarded in the security community as a means to perform vulnerability assessments against a target of choice. It is the equivalent of going to a house and checking the doors and windows to see if the locks aren't properly attached or if there are any known ways to finagle the lock to open. Communication on the Internet follows a standard of communication vetted by an Internet governing body. These protocols determine how end-to-end communication is initiated and how that communication is sustained until one party decides to terminate the connection. It is important to note where the scanning activity originates. This is integral to our analysis as we pinpoint the infected hosts on the netblock and external hosts that account for large volumes of traffic. The SiLK Toolkit provides the *rwscan* tool, which allows us to detect scans targeting our netblock or originating from it.

To detect scans directed at netblock 193.52.0.0/16 run the following:

```
rwfilter ext*/* --daddr=193.52.0.0/16 --all-dest=stdout | rwsort --fields=sip,proto,dip | rwscan --scan-model=2 --
output-path=<somepath>
```

To detect scans originating from netblock 193.52.0.0/16 run the following:

```
rwfilter ext*/* --saddr=193.52.0.0/16 --all-dest=stdout | rwsort --fields=sip,proto,dip | rwscan --scan-model=2 --
output-path=<somepath>
```

The data is first partitioned and sorted based on the source ip, protocol, and destination ip.

Then the *rwscan* tool is used with the BLR algorithm, which is said to have less false positives than its TRW counterpart. [1] A series of metrics for determining if a host is a scanner is discussed thoroughly in the documentation.

Results of these findings are below in a graphical treemap representation.

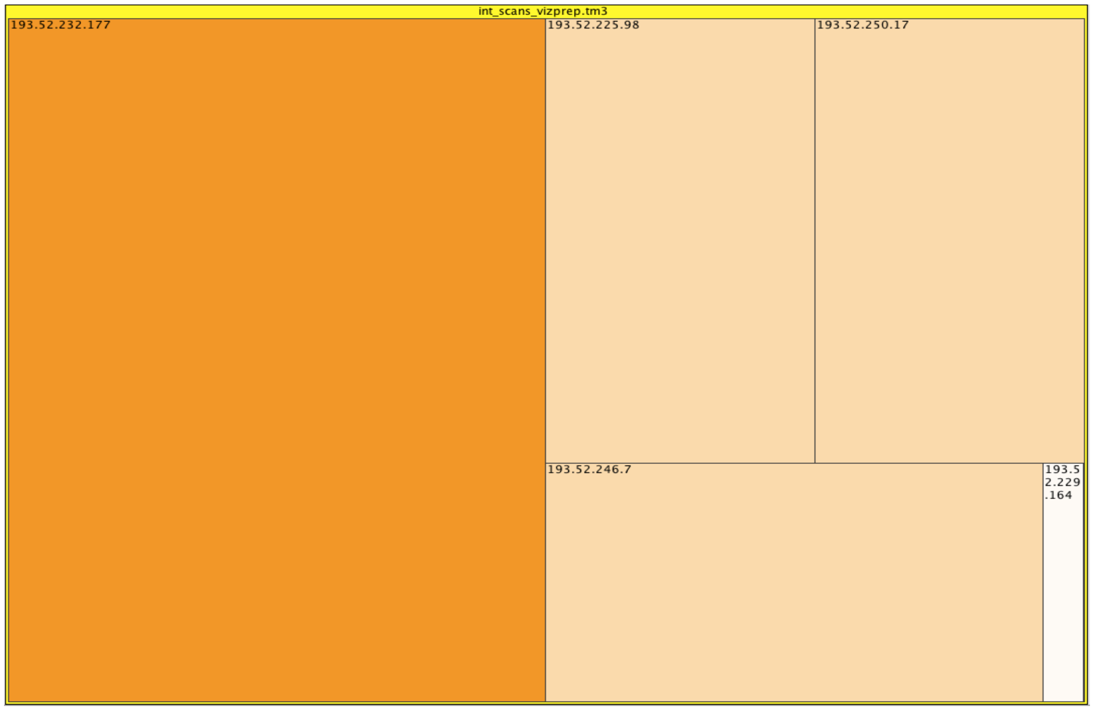


Figure 1: Internal Scanners from netblock 193.52.0.0/16

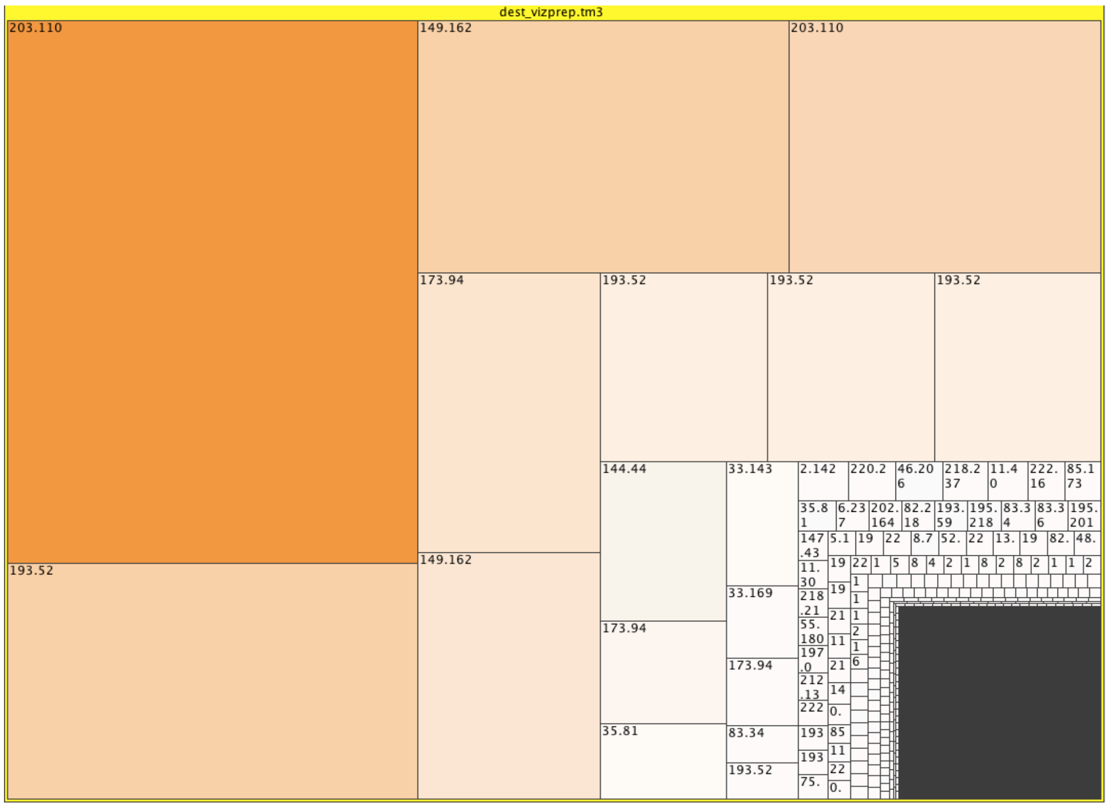


Figure 2: External Scanners targeting netblock 193.52.0.0/16

In using visualization software, specifically Treemap 4.0, it is easy to note which netblocks are harboring scanners that degrade the service of our netblock. With this knowledge we recommend the use of load balancing techniques and outreach to the responsible netblock administrators. The disproportionate amount of activity on 203.110 could be partitioned further to conclude which hosts are the culprits of such malicious behavior.

Other activity of interest is high port to high port communication, which is typical of worm traffic. Using SiLk tools we can partition based on communication originating from netblock 193.52.0.0/16 where the source and destination ports are both ephemeral.

To see high port to high port communication originating from netblock 193.52.0.0/16 do the following:

```
rwfilter ext*/* --saddr=193.52.0.0/16 --sport=1024- --dport=1024- --pass=stdout | rwstats --
dip --top --count=10
INPUT SIZE: 7724795 records for 966216 unique keys
DESTINATION IP Key: Top 10 flow counts
            dIP|            Records|%_of_total|   cumul_%|
   83.49.177.67|             123728|  1.601699|  1.601699|
   85.30.80.159|              93955|  1.216278|  2.817978|
   83.34.64.143|              91393|  1.183112|  4.001090|
 111.85.154.207|              79249|  1.025904|  5.026994|
  55.213.74.242|              59809|  0.774247|  5.801241|
  61.51.171.205|              50217|  0.650076|  6.451317|
   43.59.192.17|              45555|  0.589724|  7.041041|
  51.53.187.233|              45497|  0.588974|  7.630015|
   2.22.197.175|              43068|  0.557529|  8.187544|
    11.41.97.77|              38737|  0.501463|  8.689007|
```

To see high port to high port communication targeted to netblock 193.52.0.0/16 do the following:

```
rwfilter ext*/* --daddr=193.52.0.0/16 --sport=1024- --dport=1024- --pass=stdout | rwstats --
sip --top --count=10
INPUT SIZE: 9377814 records for 1010840 unique keys
SOURCE IP Key: Top 10 flow counts
            sIP|            Records|%_of_total|   cumul_%|
 222.16.253.237|             256954|  2.740020|  2.740020|
 218.216.95.179|             163861|  1.747326|  4.487346|
 55.180.183.229|             135923|  1.449410|  5.936757|
   83.49.177.67|             123956|  1.321801|  7.258557|
    83.34.73.41|              98310|  1.048325|  8.306883|
   85.30.80.159|               94158|  1.004051|  9.310933|
   83.34.64.143|               91600|  0.976773| 10.287707|
 111.85.154.207|               79251|  0.845090| 11.132797|
   2.22.197.175|               67895|  0.723996| 11.856793|
   82.188.86.17|               62131|  0.662532| 12.519325|
```

Furthermore, we are concerned with incomplete TCP handshakes directed at specific hosts within the netblock. Popular scans of this type are packets with the SYN flag set. However, when these packets increase in volume it serves as evidence of a SYN flooding denial of service (DOS) attack.

To see flows where the SYN flag is set and directed at netblock 193.52.0.0/16 do the following:

```
rwfilter ext*/* --daddr=193.52.0.0/16 --syn=1 --ack=0 --fin=0 --proto=6 --dport=80 --pass=stdout | rwuniq --fields=sip --
all-counts --flows=500 --dip-distinct
rwuniq: Warning: Using default temporary directory /tmp
            sIP|               Bytes|   Packets|   Records|          min_sTime|          max_eTime|Unique_DIP|
   218.160.72.5|              180288|      2817|      2776|2009/03/31T00:43:46|2009/03/31T00:49:39|       470|
   215.187.89.6|             2020580|     33680|     33669|2009/03/30T15:00:03|2009/03/31T14:59:57|         2|
    39.47.60.10|              275200|      6880|      3440|2009/03/30T15:02:30|2009/03/31T14:57:57|         1|
  46.249.113.11|              149640|      2494|      2494|2009/03/30T15:01:13|2009/03/31T14:54:42|         1|
  222.48.204.15|              198112|      4169|      1585|2009/03/31T01:46:20|2009/03/31T14:59:57|         2|
   84.231.57.52|              834480|     13933|     13858|2009/03/30T15:00:02|2009/03/31T14:59:47|         2|
   53.78.199.66|              262560|      5470|      1849|2009/03/30T15:03:12|2009/03/31T14:58:06|         1|
  60.165.163.70|               44680|      1117|      1105|2009/03/30T15:02:51|2009/03/31T14:59:39|      1105|
    85.21.46.82|               96672|      2014|       938|2009/03/30T17:37:55|2009/03/31T13:45:46|         2|
   11.34.138.82|               56216|      1177|       512|2009/03/31T02:52:37|2009/03/31T14:56:37|         2|
199.251.147.101|              369540|      6159|      1251|2009/03/30T15:00:11|2009/03/31T14:59:56|         1|
 195.109.25.118|               70060|      1173|       751|2009/03/30T17:52:29|2009/03/30T17:54:01|        90|
 222.148.54.132|              144592|      3018|      1586|2009/03/30T18:05:18|2009/03/31T08:18:51|         2|
 219.69.194.155|               36720|       765|       698|2009/03/31T10:57:24|2009/03/31T14:58:21|         1|
  84.65.250.191|               54176|      1178|       882|2009/03/31T03:26:33|2009/03/31T05:44:16|         2|
  11.35.148.206|               71856|      1497|      1109|2009/03/30T18:54:22|2009/03/30T23:41:50|         2|
 11.161.191.249|              154408|      3488|      1786|2009/03/30T15:00:12|2009/03/31T14:59:30|         1|
```

One potential occurrence of worm traffic has been since refuted in our analysis due to mere traffic volume, but will be discussed here as an alternative to that hypothesis to showcase our methodology. Issuing the following command results in a listing of internal hosts that communicate from some ephemeral source port to destination port 3127, which is a characteristic of one popular worm outbreak.

```
rwfilter ext*/* --daddr=193.52.0.0/16 --sport=1024- --dport=3127 --pass=stdout | rwstats --dip
--top --count=50
INPUT SIZE: 1405246 records for 272 unique keys
DESTINATION IP Key: Top 50 flow counts
            dIP|             Records|%_of_total|   cumul_%|
 193.52.232.189|              476745| 33.926088| 33.926088|
 193.52.232.188|              433862| 30.874452| 64.800540|
 193.52.232.177|              430368| 30.625812| 95.426352|
 193.52.232.176|               62464|  4.445058| 99.871410|
```

These alleged hosts are infected with the myDoom worm, which has been polarized in the security community due to its widespread propagation through email channels and P2P file sharing networks. It has acquired the status as one of the largest mass mailers to date; hence its approaches to infection are well documented. The worm was discovered in January 2004 and allegedly caused uproar concerning the practices of how antivirus vendors disseminate information. The worm took advantage of these alerting mechanisms as propagation technique. It commonly spreads as an infected attachment and creates a backdoor proxy listening on port 3127 TCP.[3]

DDoS Vulnerability

The threats to our netblock are modest considering its size; however, it should be dually noted that an influx of traffic targeted at servers 193.52.237.154 and 193.52.237.155 could be detrimental their ability to serve requests. We have identified 5 servers on our netblock and 3 proxies that account for huge volumes of the total traffic. These hosts can become bottlenecks and discontinue the flow of traffic. Another cascading effect can be the fail mechanism of these servers. We cannot determine their configurations, but in the event they fail open our netblock will become exposed to a series of attacks that can prove harmful. For example, if a DDoS attack analogous to the one that occurred the summer of 09 were to stampede our netblock we could see a degradation of services on port 80 and 443. It is imaginable that failures would

[3] Stewart, Joe. "MyDoom Worm Advisory." SecureWorks. 2004 January 27. <http://www.secureworks.com/research/threats/mydoomadvisory/>

occur and potentially open the network to scathing attacks from adversaries. The integrity of the communication stream would be compromised and require a massive overhaul of the infected hosts to be considered operational. Even worse, the attack could go without notice and serve as a hotbed for malicious software and compromised drone computers who do the bidding of a remote controller.   We recommend DNS round robin techniques to assist in the load balancing of such requests. Furthermore, firewall rulesets should govern that a series of packets with only the SYN flag set should be dropped.   This requires the firewall to be stateful and most importantly, this information to be cross-referenced with our flow analysis. Any overlaps should punctuate the root cause of the problem and the business continuity plan should be enforced.

Conclusion

Our analysis on the netblock proved fruitful in developing a methodology for properly conducting flow analysis. The attestation process is much more challenging since full content captures aren't inclusive in the set of data; however, it does give a macro-level view of network. The SiLK Toolkit is crafted for such analysis but is not a panacea for situational awareness. Through this exercise, layering the defense architecture to include flow analysis will benefit the network engineers and magnify performance issues and attacks on the network.

Appendix A

```
INPUT SIZE: 12724 records for 111 unique keys
SOURCE IP Key: Top 10 byte counts
            sIP|              Bytes|%_of_total|   cumul_%|
 193.52.232.189|            4050269| 20.870596| 20.870596|
 193.52.232.188|            3050463| 15.718705| 36.589301|
 193.52.232.177|            2684688| 13.833906| 50.423207|
 193.52.230.218|            1497380|  7.715837| 58.139044|
 193.52.229.232|             966711|  4.981357| 63.120400|
 193.52.229.240|             954749|  4.919718| 68.040118|
 193.52.231.144|             845795|  4.358290| 72.398408|
 193.52.228.123|             825354|  4.252960| 76.651368|
 193.52.234.207|             550332|  2.835801| 79.487169|
 193.52.228.231|             427674|  2.203758| 81.690926|
INPUT SIZE: 12530 records for 122 unique keys
SOURCE IP Key: Top 10 byte counts
            sIP|              Bytes|%_of_total|   cumul_%|
  193.52.229.90|           37740813| 58.422320| 58.422320|
 193.52.232.189|           10932386| 16.923201| 75.345521|
 193.52.232.177|            2851192|  4.413611| 79.759132|
 193.52.232.188|            2358181|  3.650436| 83.409567|
 193.52.230.218|            1764350|  2.731192| 86.140759|
 193.52.229.240|            1041405|  1.612082| 87.752842|
 193.52.229.232|            1006730|  1.558406| 89.311248|
 193.52.234.207|             691509|  1.070448| 90.381695|
 193.52.228.123|             672416|  1.040892| 91.422587|
 193.52.229.130|             528041|  0.817401| 92.239988|
INPUT SIZE: 13385 records for 173 unique keys
SOURCE IP Key: Top 10 byte counts
            sIP|              Bytes|%_of_total|   cumul_%|
  193.52.229.90|            5231714| 21.326688| 21.326688|
 193.52.232.177|            2769552| 11.289870| 32.616558|
 193.52.232.189|            2650038| 10.802680| 43.419238|
 193.52.232.188|            2644264| 10.779143| 54.198381|
 193.52.230.218|            1485236|  6.054453| 60.252834|
 193.52.234.207|            1325148|  5.401866| 65.654700|
 193.52.229.232|            1052271|  4.289503| 69.944203|
 193.52.229.240|            1040557|  4.241752| 74.185955|
  193.52.238.39|             854473|  3.483195| 77.669150|
 193.52.228.123|             815907|  3.325983| 80.995134|
```

Appendix B

```
[bkennedy@unix37 ext1]$ rwfilter 2009033100.rw  --saddr=193.52.229.90
--dport=80  --pass=stdout | rwstats --dip --bytes --top --count=10 &&
rwfilter 2009033101.rw  --saddr=193.52.229.90  --dport=80  --
pass=stdout | rwstats --dip --bytes --top --count=10 && rwfilter
2009033102.rw  --saddr=193.52.229.90  --dport=80  --pass=stdout |
rwstats --dip --bytes --top --count=10 && rwfilter 2009033103.rw  --
saddr=193.52.229.90  --dport=80  --pass=stdout | rwstats --dip --bytes
--top --count=10 && rwfilter 2009033104.rw  --saddr=193.52.229.90  --
dport=80  --pass=stdout | rwstats --dip --bytes --top --count=10 &&
rwfilter 2009033105.rw  --saddr=193.52.229.90  --dport=80  --
pass=stdout | rwstats --dip --bytes --top --count=10 && rwfilter
2009033106.rw  --saddr=193.52.229.90  --dport=80  --pass=stdout |
rwstats --dip --bytes --top --count=10 && rwfilter 2009033107.rw  --
saddr=193.52.229.90  --dport=80  --pass=stdout | rwstats --dip --bytes
--top --count=10 && rwfilter 2009033108.rw  --saddr=193.52.229.90  --
dport=80  --pass=stdout | rwstats --dip --bytes --top --count=10 &&
rwfilter 2009033109.rw  --saddr=193.52.229.90  --dport=80  --
pass=stdout | rwstats --dip --bytes --top --count=10 && rwfilter
2009033110.rw  --saddr=193.52.229.90  --dport=80  --pass=stdout |
rwstats --dip --bytes --top --count=10 && rwfilter 2009033111.rw  --
saddr=193.52.229.90  --dport=80  --pass=stdout | rwstats --dip --bytes
--top --count=10 && rwfilter 2009033112.rw  --saddr=193.52.229.90  --
dport=80  --pass=stdout | rwstats --dip --bytes --top --count=10 &&
rwfilter 2009033113.rw  --saddr=193.52.229.90  --dport=80  --
pass=stdout | rwstats --dip --bytes --top --count=10 && rwfilter
2009033114.rw  --saddr=193.52.229.90  --dport=80  --pass=stdout |
rwstats --dip --bytes --top --count=10 && rwfilter 2009033115.rw  --
saddr=193.52.229.90  --dport=80  --pass=stdout | rwstats --dip --bytes
--top --count=10
INPUT SIZE: 129 records for 78 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
  147.138.17.24|               32377| 36.301155| 36.301155|
 160.137.153.25|                9867| 11.062899| 47.364054|
 45.128.130.180|                4038|  4.527413| 51.891468|
   36.36.176.75|                2065|  2.315282| 54.206750|
  96.116.72.176|                1605|  1.799529| 56.006279|
   36.22.239.85|                1451|  1.626864| 57.633143|
   46.207.29.98|                1301|  1.458684| 59.091826|
  37.126.135.91|                1207|  1.353291| 60.445117|
  196.111.22.37|                1184|  1.327503| 61.772620|
 199.202.185.20|                1168|  1.309564| 63.082184|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 237 records for 67 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
  80.35.136.164|             3318661|  8.793295|  8.793295|
 217.217.186.22|             3287535|  8.710822| 17.504117|
    80.35.136.5|             3248077|  8.606272| 26.110389|
  201.225.98.12|             3247657|  8.605159| 34.715548|
   80.35.136.75|             3243613|  8.594444| 43.309992|
    54.27.49.16|             3231097|  8.561281| 51.871273|
    54.27.4.193|             3213445|  8.514509| 60.385782|
   54.27.49.182|             2752780|  7.293908| 67.679689|
  80.35.137.215|             2706564|  7.171451| 74.851141|
 80.245.153.180|             2280661|  6.042957| 80.894097|
INPUT SIZE: 18 records for 9 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
   80.35.136.85|             1756664| 33.577218| 33.577218|
  80.35.138.115|             1517384| 29.003573| 62.580791|
    54.27.23.40|             1012324| 19.349758| 81.930549|
 80.245.153.180|              922172| 17.626575| 99.557124|
 46.138.135.136|               16598|  0.317257| 99.874382|
   208.68.5.116|                4256|  0.081350| 99.955732|
193.172.235.134|                 981|  0.018751| 99.974483|
    34.163.1.44|                 867|  0.016572| 99.991055|
  37.116.22.113|                 468|  0.008945|100.000000|
[bkennedy@unix37 ext1]$ cd ../ext2
[bkennedy@unix37 ext2]$ rwfilter 2009033116.rw  --saddr=193.52.229.90
--dport=80  --pass=stdout | rwstats --dip --bytes --top --count=10 &&
rwfilter 2009033117.rw  --saddr=193.52.229.90  --dport=80  --
pass=stdout | rwstats --dip --bytes --top --count=10 && rwfilter
2009033118.rw  --saddr=193.52.229.90  --dport=80  --pass=stdout |
rwstats --dip --bytes --top --count=10 && rwfilter 2009033119.rw  --
saddr=193.52.229.90  --dport=80  --pass=stdout | rwstats --dip --bytes
--top --count=10 && rwfilter 2009033120.rw  --saddr=193.52.229.90  --
dport=80  --pass=stdout | rwstats --dip --bytes --top --count=10 &&
rwfilter 2009033121.rw  --saddr=193.52.229.90  --dport=80  --
pass=stdout | rwstats --dip --bytes --top --count=10 && rwfilter
2009033122.rw  --saddr=193.52.229.90  --dport=80  --pass=stdout |
rwstats --dip --bytes --top --count=10 && rwfilter 2009033123.rw  --
saddr=193.52.229.90  --dport=80  --pass=stdout | rwstats --dip --bytes
--top --count=10
INPUT SIZE: 9 records for 4 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
 46.138.135.136|               15867| 71.861413| 71.861413|
   208.68.5.115|                4576| 20.724638| 92.586051|
193.172.235.139|                 981|  4.442935| 97.028986|
  198.20.160.41|                 656|  2.971014|100.000000|
INPUT SIZE: 38 records for 19 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
 46.138.135.136|               16770| 15.270164| 15.270164|
  200.77.67.229|               10000|  9.105644| 24.375808|
     54.22.10.99|              10000|  9.105644| 33.481452|
   80.139.214.89|              10000|  9.105644| 42.587095|
    212.90.16.23|              10000|  9.105644| 51.692739|
    43.14.65.129|              10000|  9.105644| 60.798383|
  200.205.198.47|              10000|  9.105644| 69.904027|
   201.31.222.56|              10000|  9.105644| 79.009670|
     213.117.4.2|               5669|  5.161989| 84.171660|
   200.77.67.195|               4874|  4.438091| 88.609750|
INPUT SIZE: 64 records for 13 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
192.122.209.231|               31756| 43.244274| 43.244274|
 46.138.135.136|               10658| 14.513713| 57.757987|
 37.116.177.207|                8461| 11.521911| 69.279898|
    208.68.5.114|               4656|  6.340387| 75.620285|
    42.86.41.150|               4252|  5.790233| 81.410518|
   200.205.198.1|               3278|  4.463872| 85.874391|
213.152.136.125|                3189|  4.342675| 90.217066|
   34.143.144.68|               2493|  3.394885| 93.611951|
   16.68.189.205|               1896|  2.581910| 96.193861|
193.172.235.138|                 981|  1.335893| 97.529755|
INPUT SIZE: 73 records for 25 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
   213.70.71.102|              23868| 11.620706| 11.620706|
   80.139.214.89|              20000|  9.737478| 21.358183|
    43.14.65.129|              20000|  9.737478| 31.095661|
  200.205.198.47|              20000|  9.737478| 40.833139|
    212.90.16.23|              20000|  9.737478| 50.570616|
   200.77.67.229|              20000|  9.737478| 60.308094|
   201.31.222.56|              20000|  9.737478| 70.045571|
     54.22.10.99|              20000|  9.737478| 79.783049|
     213.117.4.2|              12376|  6.025551| 85.808600|
   200.77.67.195|               6845|  3.332652| 89.141252|
INPUT SIZE: 51 records for 12 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
192.122.209.231|               27093| 25.058732| 25.058732|
     54.22.10.99|              10000|  9.249154| 34.307886|
    43.14.65.129|              10000|  9.249154| 43.557040|
   200.77.67.229|              10000|  9.249154| 52.806193|
   201.31.222.56|              10000|  9.249154| 62.055347|
   80.139.214.89|              10000|  9.249154| 71.304501|
    212.90.16.23|              10000|  9.249154| 80.553654|
  200.205.198.47|              10000|  9.249154| 89.802808|
   200.77.67.195|               4529|  4.188942| 93.991750|
  206.202.68.50|                3052|  2.822842| 96.814591|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
INPUT SIZE: 0 records for 0 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
```

Appendix C

```
INPUT SIZE: 686 records for 31 unique keys
SOURCE IP Key: Top 10 byte counts
            sIP|               Bytes|%_of_total|   cumul_%|
 193.52.227.125|             2610657| 64.680452| 64.680452|
  193.52.238.39|              319788|  7.922922| 72.603375|
 193.52.233.113|              174742|  4.329328| 76.932703|
  193.52.227.99|              153912|  3.813254| 80.745957|
 193.52.230.218|              122420|  3.033022| 83.778979|
  193.52.228.85|              120402|  2.983025| 86.762005|
  193.52.228.64|              112828|  2.795375| 89.557380|
  193.52.252.82|               87312|  2.163202| 91.720582|
  193.52.228.37|               78835|  1.953180| 93.673763|
 193.52.232.189|               36380|  0.901334| 94.575097|
INPUT SIZE: 494 records for 24 unique keys
SOURCE IP Key: Top 10 byte counts
            sIP|               Bytes|%_of_total|   cumul_%|
 193.52.227.125|            13188787| 91.582642| 91.582642|
  193.52.238.39|              280947|  1.950890| 93.533532|
 193.52.230.218|              178109|  1.236785| 94.770317|
  193.52.227.99|              153487|  1.065810| 95.836127|
  193.52.228.64|              117684|  0.817195| 96.653322|
  193.52.228.85|              110596|  0.767976| 97.421298|
  193.52.252.82|              108436|  0.752977| 98.174276|
  193.52.228.37|               50615|  0.351469| 98.525745|
 193.52.232.189|               34998|  0.243025| 98.768770|
 193.52.232.188|               34752|  0.241317| 99.010087|
INPUT SIZE: 511 records for 24 unique keys
SOURCE IP Key: Top 10 byte counts
            sIP|               Bytes|%_of_total|   cumul_%|
 193.52.227.125|             7474243| 86.280158| 86.280158|
  193.52.238.39|              250091|  2.886967| 89.167125|
  193.52.227.99|              153800|  1.775416| 90.942540|
 193.52.230.218|              136343|  1.573898| 92.516438|
  193.52.228.85|              111156|  1.283148| 93.799586|
  193.52.252.82|              108503|  1.252522| 95.052108|
  193.52.228.64|              104943|  1.211427| 96.263535|
  193.52.228.37|               85190|  0.983405| 97.246940|
 193.52.233.113|               47488|  0.548186| 97.795125|
 193.52.232.189|               35264|  0.407076| 98.202201|
```

Appendix D

```
INPUT SIZE: 5 records for 3 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
   47.89.25.106|             2603358| 99.720415| 99.720415|
 39.149.249.181|                5348|  0.204853| 99.925268|
 39.149.249.186|                1951|  0.074732|100.000000|
INPUT SIZE: 6 records for 3 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
   47.89.25.106|            13183107| 99.956933| 99.956933|
 39.149.249.181|                3701|  0.028062| 99.984995|
  35.44.213.154|                1979|  0.015005|100.000000|
INPUT SIZE: 7 records for 3 unique keys
DESTINATION IP Key: Top 10 byte counts
            dIP|               Bytes|%_of_total|   cumul_%|
   47.89.25.106|             5296799| 70.867364| 70.867364|
212.214.241.222|             1640096| 21.943306| 92.810670|
     47.89.2.12|              537348|  7.189330|100.000000|
```