Network Situational Awareness, Spring 2009

Analysis of 193.52.0.0 and the MAWI set

Ron Bandes
Francis Fbgormittah
Robert Jackson Lee
Allison MacFarlan

Table of Contents

Table of Figures

Introduction, Executive Summary

Political, environmental and network unrest were prevalent in the world on March 30th and 31st 2009. The primary concern in the information technology world was a worm called "Conficker" that had morphed two previous times and whose purpose and source code was still being extensively analyzed. There was general fear that on April 1st Conficker would expand its influence and release a nefarious payload or a destructive attack on the world's networks.

Because of the time of our sample and the global focus on Conficker, we used this circumstance as the internet-wide "event" for our case study, and analyzed all the data in our sample to determine if Conficker had a significant effect on the large Class B network we chose for analysis (193.52.x.x). This was the problem we were trying to solve:

Was Conficker exhibited on our network, and if so, could we prove it?

We determined that there were two machines that were responsible for excessive service and port traffic, with over 77% of their available IP services listening for connections – these were likely compromised. We analyzed how much traffic our network sent to other locations, and what kind it was. We determined that our network seemed to be scanning other networks. But wasn't the biggest scanner by a wide margin. Our network was the biggest destination for web traffic. We also looked at the overall volume of traffic in our network with regard to the "whole" – the one-day sample we were provided from the Pacific listening post. Our subnet accounted for 21.31% of the total traffic collected by MAWI for this 24-hour period.

We then investigated other port traffic (email, web, ftp etc…) to determine the function of our Class B, what kind of traffic was predominant, and decided that our network is probably some kind of service provider. We found lots of ephemeral port traffic too, and that it exceeded all other kinds of traffic on our network. The reason for this last finding was not conclusive.

Political and Economic Context: March 31, 2009

On March 31, 2009, the major industrialized countries of the world were mired in a recession caused by bank exposure to high-risk debt, coupled with a sharp rise in oil prices that had driven up prices of most consumer goods. Stocks on the Dow Jones index had plunged 250 points on March 30th in reaction to the news that U.S. home prices had fallen by their greatest margin in January. President Obama and his advisors were considering the implications of a possible bailout of the U.S. auto industry, while General Motors and Chrysler warned of impending bankruptcy. The U. S. Congress was engaged in a federal budget debate and a Senate committee was holding confirmation hearings for HHS nominee Kathleen Sibelius. The U.S. House Subcommittee on Emerging Threats, Cybersecurity, Science and Technology was presiding over a 60-day review of the nation's cybersecurity efforts, amid calls for the appointment of a "Cybersecurity Czar." The FDA issued a warning about possible salmonella in pistachio nuts. A Senate committee was considering the elimination of the travel ban to Cuba.

Truck bombs[1] and continued attacks were making it less likely that Iraq would be able to stabilize itself without the prolonged (financial and military) assistance of the United States. On March 31st, Britain began to withdraw its own forces from Basra. Israel was pushing the United States for economic sanctions against Iran as the world's leaders went to London to gather for the G20 Summit. Seven Somali pirates opened fire on a German Naval supply ship in the Gulf of Aden, and were chased down and captured by an anti-piracy task force. A state of emergency was declared in the Philippine island of Jolo, where Abu Sayef rebels were threatening to behead one of their Red Cross hostages. A boat filled with African migrants capsized and sank off the Libyan coast, drowning more than 200 passengers. North Korea announced that it had detained two U.S. journalists, Euna Lee and Laura Ling, who had strayed across its borders. Albania and Croatia were getting ready to join NATO on April 1st.

[1] http://www.nytimes.com/2009/03/31/world/middleeast/01iraq.html?_r=1&scp=25&sq=3/31/09&st=cse

A fireball, likely a meteor, was reported across the Canadian provinces of Canada. A magnitude 4.3 earthquake struck the South Bay area of San Francisco. Sandstorms moved into northern China, causing officials to shut down highways and airports.

Internet Context: March 31, 2009

On the Internet, Red Hat revealed the details behind an August 2008 server compromise that had undermined the integrity of its update servers.[2] SecDev, a Canadian think tank, released findings from a ten-month study on a Chinese internet spy network that had infiltrated many international foreign affairs ministries and the computer systems of the Dalai Llama.[3] Media outlets were warning that the newest version of Conficker (Conficker.d, also known as the "Downadup" worm) was set to activate the next day on millions of machines, and security teams were preparing to handle the many infected hosts in their domains. This particular worm garnered attention because it was estimated that one in five Windows computers were still unpatched for MS08-067 (vulnerability in the Microsoft server service). The U. S. Department of Homeland Security released a Conficker detection tool for government agencies[4] that was developed by US-CERT, amidst concern that the size of the botnet could wreak havoc in those Federal agencies or hamper social services in local municipalities. The Honeynet Project also released a paper,[5] a proof-of-concept scanner and signature for nmap[6] based on the discovery that the worm installs a handler in its RPC calls that is set to a specific constant.

[2] http://blog.internetnews.com/skerner/2009/03/red-hat-fedora-reveals-details.html#more
[3] http://www.infowar-monitor.net/2009/03/tracking-ghostnet-investigating-a-cyber-espionage-network-3/
[4] http://news.cnet.com/conficker-flaw-reveals-which-computers-are-infected/?tag=mncol;txt
[5] "Know Your Enemy: Containing Conficker", Felix Leder, Tillmann Werner (The Honeynet Project, March, 2009)
[6] http://insecure.org/

The Conficker Threat, in Detail

On October 23, 2008, Microsoft released an out-of-band patch (MS08-067) to plug a "privately reported"[7] security vulnerability in the NetpwPathCanonicalize() service that caused a buffer overflow in response to a specially-crafted RPC call. This vulnerability existed in all extant versions of Windows since W2000 SP4, including patched Vista, Server and client systems, although Server and Vista systems required the user to be authenticated. The patch had been released in response to targeted attacks[8] against the RPC service. By November 2008, Conficker.a (Downadup) had appeared, and it was the first worm to exploit this vulnerability on a large scale. The novelty of this worm was that it could also use the universal plug and play (UnPnP) and AutoPlay features of Windows to propagate, so it wasn't entirely deflected by network mitigations. Common USB fobs were driving up infection rates inside organizations with very restrictive firewall policies and NATted networks. The basic mechanism of the worm was to target the Microsoft-DS listening service at port 445 or 139, infect the host, and then set up a web server at a random port to receive the rest of its payload. Each version of the worm did this differently,[9] and as it mutated, it became less reliant on the presence of fixed exterior update sources. The "D" version employed advanced features like encryption, self-removal and self-update to new versions, dictionary attacks, peer-to-peer push/pull updating, UDP scanning, blacklisting, and opportunistic incorporation of other malware, like fake virus software.

Another interesting feature of this worm was that it computed its own domain names for the web server using a random seed generator[10]. It then did a lookup for 500 of them to check if the domains were valid, and if they were not, slept for 24 hours, and generated more random domains. These 500 name lookups created dns collisions and excess udp 53 traffic to name servers, and could effectively cause a denial of service for them. The domain traffic was

[7] http://www.microsoft.com/technet/security/Bulletin/MS08-067.mspx - hacker unknown
[8] Nahorney, Ben; Park, John (2009-04-21), "Connecting The Dots: Downadup/Conficker Variants", *The Downadup Codex* (2.0 ed.), p. 1 – the targets were not identified
[9] *Conficker* http://en.wikipedia.org/wiki/Conficker#cite_note-0
[10] Conficker.c generates 50,000 domains (Leder, Werner, *ibid.,* p. 9

preceded by a valid query to one of the large search sites (msn, yahoo, google, even facebook) to obtain time synchronization, and to fill out the http headers for the advertised web server that was going to distribute the malware. This random <various character name> was appended with a random top-level domain (to avoid blackholing). Since all Conficker-infected machines were being seeded with the same pseudo-random number generator (PRNG), some of the infected machines were able to contact other infected machines, despite these collisions, and obtain the rest of the files they needed to fully join the botnet.

Conficker.d, released on March 3, 2009, was the most complex and mature version of the worm. It computed 50,000 domains, and instead of using the netbios "pull" of the previous three versions, developed its own push/pull, custom peer-to-peer network that scanned for new hosts on UDP, then transferred the malware via TCP[11]:

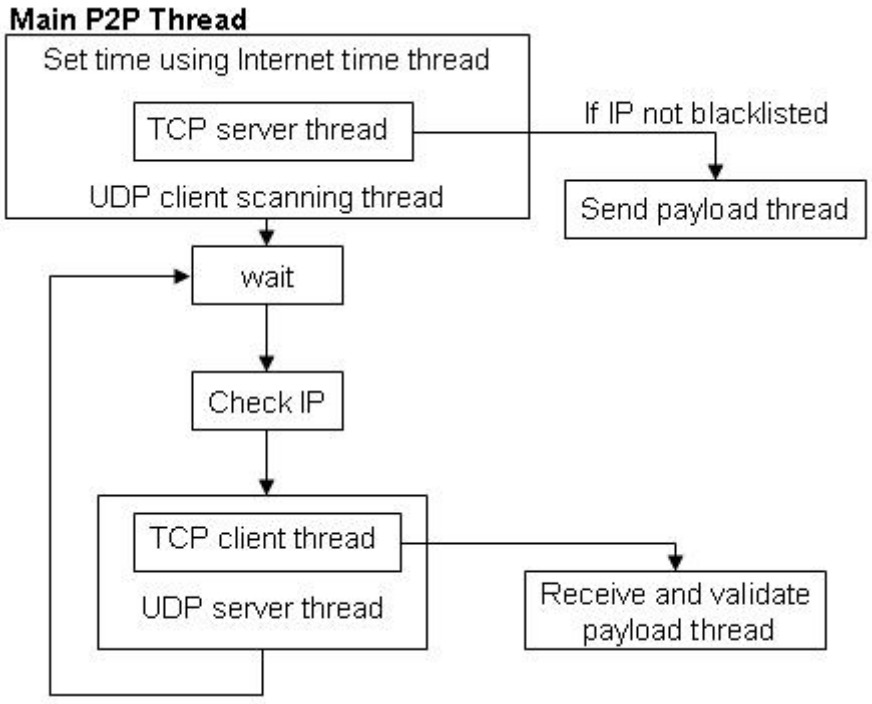

Figure 1: Diagram of Conficker D's P2P Traffic

Each version of the worm updated previous versions. In press releases and statements about the worm, the "P2P" and "50,000 domains" features were sometimes attributed to

[11] http://mtc.sri.com/Conficker/addendumC/index.html

Conficker.c. Conficker demonstrated a work in progress, as it had gone through 3 stages to date and was estimated to have a penetration of anywhere from 3 to 15 million machines. [12] It was an important feature of all internet traffic in late 2008 and 2009, with an estimated 10 million infected hosts in its botnets at the beginning of 2009[13] before the advent of Conficker.c in late February. By March 31, 2009, security organizations had determined that there was a trigger in the code for Conficker.c that activated on April 1st, although it was unclear what would happen. Conficker.d had already been released, and this trigger raised the prospect of severe problems in internet connectivity and loss of service, if all the machines with different versions started behaving like the latest version.[14] The population of Conficker-infected machines, with versions between .a - .d could scan at either TCP/445 or UDP/445, and compute between 250 and 50,000 new domains.

The Conficker Working Group produced the following distribution and maps for April 1 2009, estimating the penetration of the worm on a global scale.[15]

Global Penetration

Figure 2: Global Penetration of the Conficker Worm

U. S. Penetration

Figure 3: U.S Penetration of the Conficker Worm

[12] http://www.confickerworkinggroup.org/wiki/#toc6
[13] http://www.theregister.co.uk/2009/01/26/conficker_botnet/
[14] The Downadup Codex, ed 2.0 (Symantec Security Response, Ben Nahorney, Editor)
[15] http://www.confickerworkinggroup.org/wiki/pmwiki.php/ANY/InfectionDistribution

Heat Map

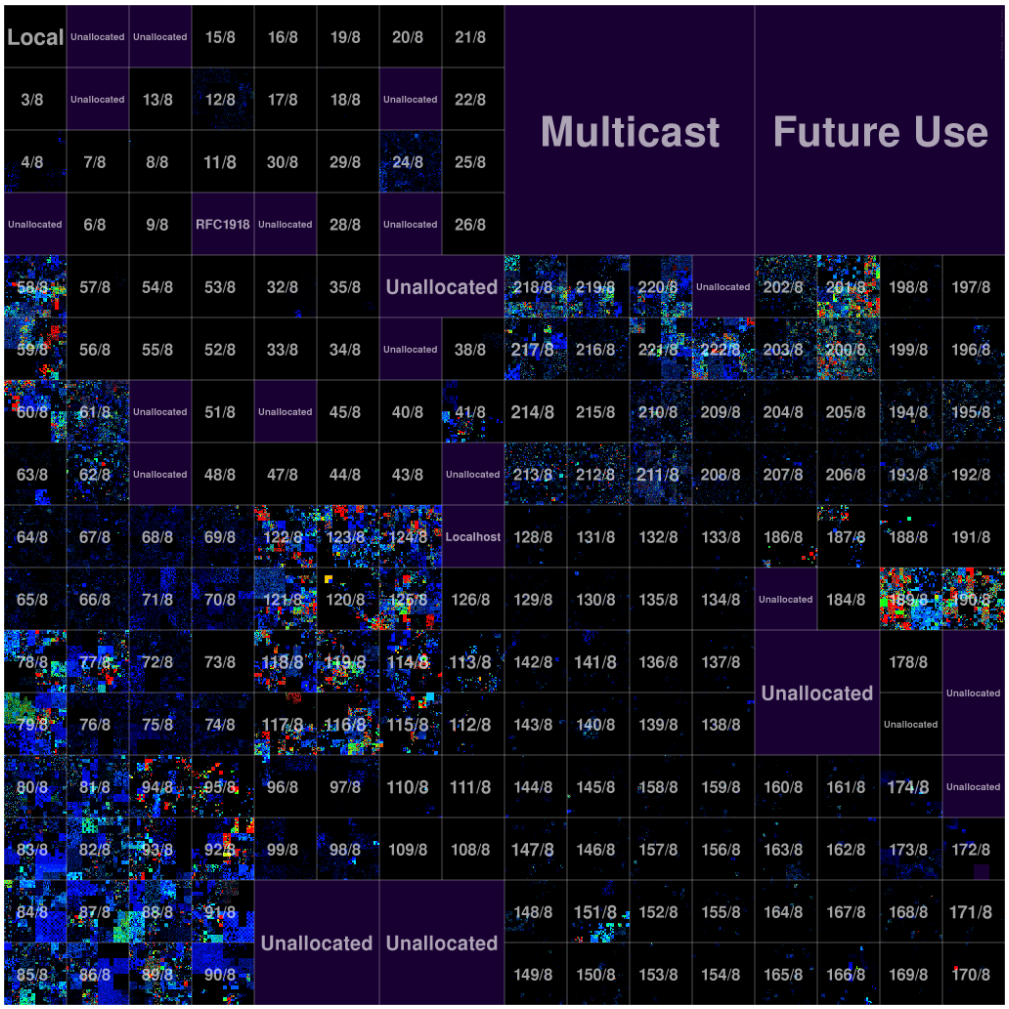

Figure 4: Heat Map for the Conficker Worm

One Day in late March, 2009

The MAWI working group,[16] based in Japan, sampled the WIDE[17] backbone on March 31st as part of the Day-in-the-Life-of-the-Internet project in 2009. Our version of that sample was converted into one-hour SiLK files with GMT-equivalent dates, but the dates in each file were offset by nine hours, corresponding to the time shift between GMT and Japan. Therefore, the first flows in our sample started at 3PM on 3/30/2009, rather than midnight GMT, since Tokyo is nine hours ahead of GMT. The traffic was apparently "anonymized" with the address prefixes preserved, so that two addresses with a common prefix were mapped to other addresses of the same length (subnet preserving). A total of 1.083 trillion bytes were sampled in 168.9 million unidirectional flows when counted by SiLK analysis tools.

Date	Records	Bytes	Packets
2009/03/30T00:00:00	64,060,432.26	356,743,598,624.05	487,138,304.44
2009/03/31T00:00:00	104,886,809.74	726,733,412,103.96	1,056,716,142.56
	168,947,242.00	1,083,477,010,728.01	1,543,854,447.00

Figure 5: SiLK Counts of Overall MAWI network statistics

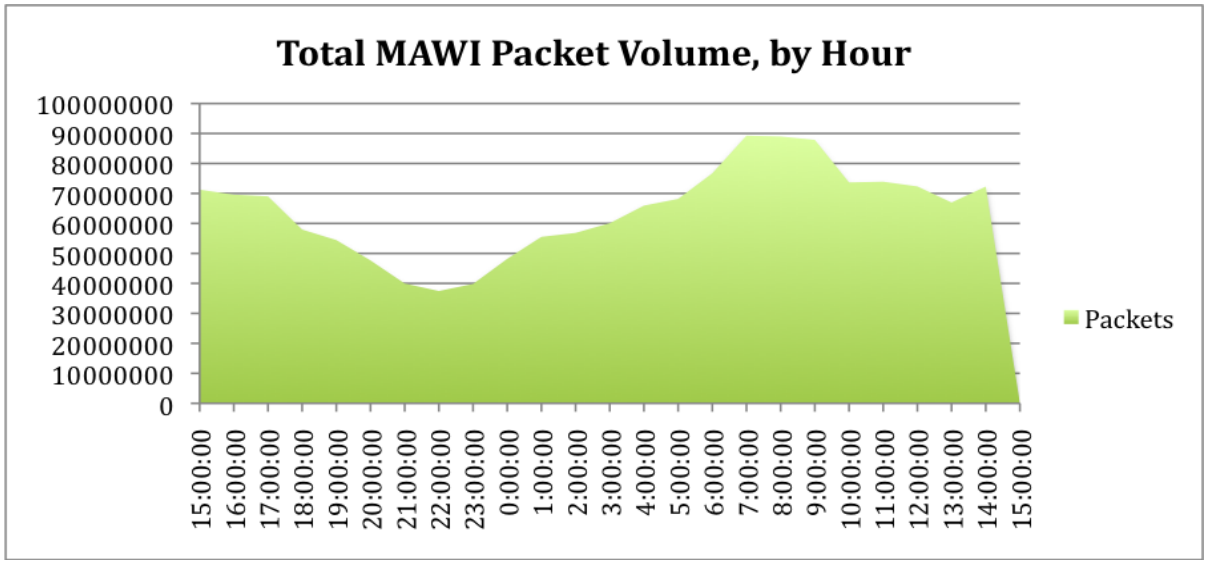

Figure 6: Packets by Hour for the MAWI Sample

[16] http://tracer.csl.sony.co.jp/mawi/
[17] http://www.wide.ad.jp/index.html

Our group counted 20,352 unique Class B networks in the sample. The top ten networks included the network we selected for further analysis (193.52.0.0/16), and which turned out to be the biggest in overall destination packet count:

INPUT SIZE: 168947242 records for 20404 unique keys
DESTINATION IP Key: Top 10 packet counts

dIP	Packets	%_of_total	cumul_%
193.52.0.0	224218579	14.523298	14.523298
173.94.0.0	106759225	6.915109	21.438407
36.87.0.0	55666816	3.605704	25.044111
168.35.0.0	49480549	3.205001	28.249112
144.44.0.0	39627021	2.566759	30.815871
203.110.0.0	34175345	2.213638	33.029508
192.34.0.0	31021403	2.009348	35.038856
192.42.0.0	26973859	1.747176	36.786032
222.5.0.0	19021968	1.232109	38.018141
149.101.0.0	18472574	1.196523	39.214664

Figure 7: Top Ten Networks by Packet count, MAWI

Among all UDP and TCP port traffic types, DNS was by far the largest percentage of all traffic with 78.8% of all flows, and we identified the 203.110.0.0/16 network as being largely comprised of this type.

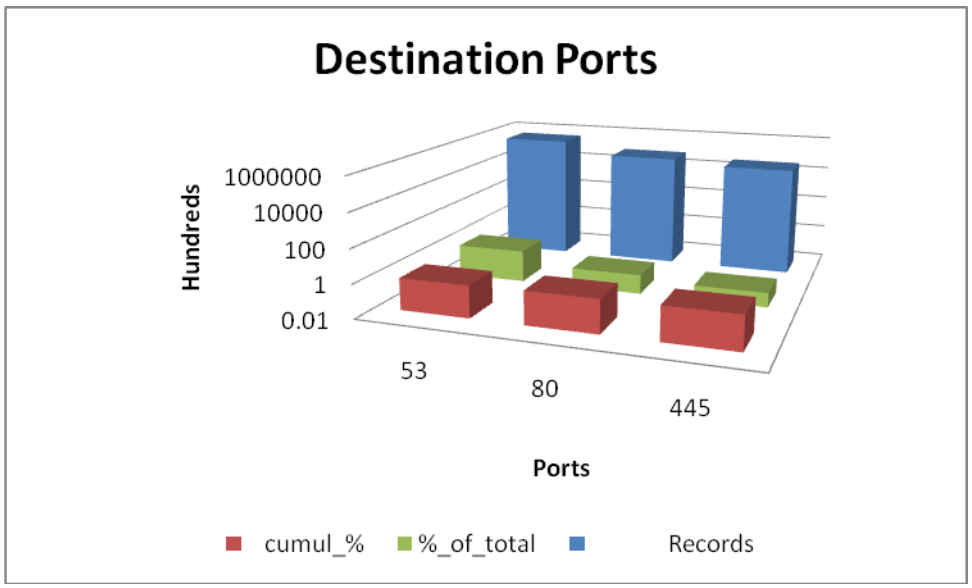

Figure 8: Top Three Destination ports in MAWI, TCP and UDP

But when networks were analyzed for just TCP, web flows[18], or port 80, accounted for the largest proportion (14.98%) of TCP destination traffic to all networks, and exceeded all other kinds of traffic by 7.2%. The second highest destination port was port 445, Microsoft-DS, which may have been related to the Conficker activity.

DESTINATION	Records	%_of_total	cumul_%
80	7200533	14.986509	14.986509
445	3734987	7.773649	22.760158
25	2168688	4.513702	27.27386
3128	1950554	4.059699	31.333559
1433	1907314	3.969703	35.303262
2967	1554351	3.235079	38.538341
3124	1516582	3.15647	41.694812
3127	1510695	3.144218	44.839029
135	1125386	2.342272	47.181302
22	902311	1.877985	49.059286
443	574033	1.194738	50.254024
8080	360578	0.750473	51.004497
3389	351548	0.731679	51.736176
139	255271	0.531297	52.267473
4899	239722	0.498935	52.766408

Figure 9: Top Fifteen Destination ports, by Flow count, in the MAWI sample

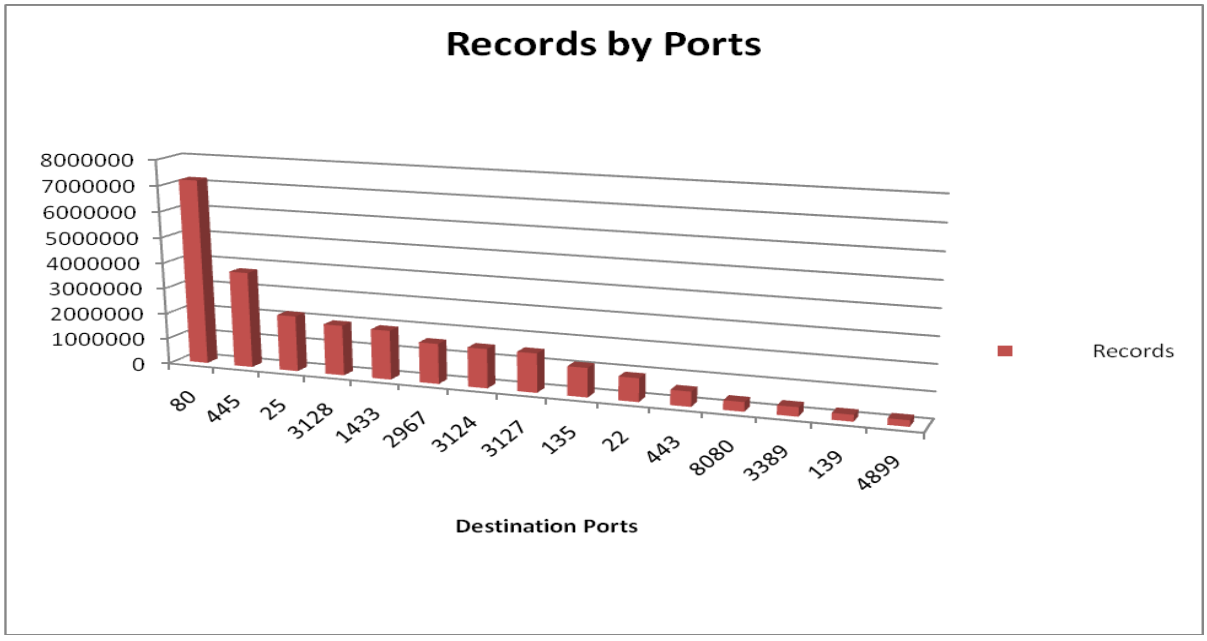

Figure 10: Top 15 Destination ports by Record/Flow in MAWI - graphic

[18] Includes streaming audio

When the MAWI traffic is analyzed apart from its ports, it becomes apparent that most of its packets occur at shorter intervals (83.15%), intervals of 3 seconds or less, suggesting that there **was** lots of scanning activity on the network that day. Flows moving through sample point F. were highest at the interval between 7-9:00 PM UTC, although this would have been equivalent to 10:00 AM to 12:00 PM Pacific Time. Ironically, when overall SYN-only traffic is sorted and matched to its port number for the whole sample, port 445 was only 4th in the ranking of overall destination scan ports:

Scan Count	Dest Port	Port Description
236428	2048	dls-monitor
164121	2967	IRCbot (also Symantec AV Corp Ed)
89717	1433	MS-SQL Server
65476	445	Microsoft-DS (SMB over
65280	135	EPMAP (End Point Mapper)
59887	22	SSH
32665	4899	radmin - Remote Administrator
32654	8080	Http-alt
32401	9090	OpenFire admin console, SqueezeCen
32401	3128	Squid cache
32379	6588	
13822	10000	Webmin, BackupExec
8927	10001	
4598	771	
3834	6881	Bit
3651	16001	
3542	53	DNS

Figure 11: Scan destination ports, sorted by count

Our sample was collected on the day just before the Conficker trigger was supposed to update the worm to a P2P service. In our whole day sample, the majority of TCP/445 packets (66% of packet counts) were coming from the 149.187.0.0/16 network:

Top source networks by TCP 445 packets
INPUT SIZE: 198286 records for 3529 unique keys
SOURCE IP Key: Top 20 packet counts

sIP	Packets	%_of_total	cumul_%
149.187.0.0	610889	66.456166	66.456166
193.52.0.0	49177	5.349769	71.805934
60.165.0.0	46992	5.112071	76.918006

Figure 12: Top source networks for TCP/445 traffic

So it seemed to us that the 149.187.0.0/16 was a likely source of infected and scanning machines, given the percent differential between that network and the one we chose, which came in second.

Two Class B networks seemed to be the most popular targets for TCP/445 traffic, and were potential sources for newly-infected Conficker machines:

Top destination networks by TCP 445 packets
INPUT SIZE: 3007864 records for 27 unique keys
DESTINATION IP Key: Top 20 packet counts

dIP	**Packets**	**%_of_total**	**cumul_%**
133.44.0.0	**3583276**	**59.426304**	**59.426304**
144.44.0.0	800291	13.272306	72.69861
149.162.0.0	792762	13.147443	85.846053
149.180.0.0	226436	3.755294	89.601347
192.177.0.0	223247	3.702406	93.303753

Figure 13: Top Destination networks for TCP/445 packets

The sharp difference between the first and second target shows that the 133.44.0.0 Class B seemed to be an **intentional** target for this scanning, rather than part of random distribution over the course of this day. There may have been many vulnerable hosts, or many infected hosts already participating in the Conficker botnet, that might be eligible for the April 1st "upgrade".

This port 445 traffic exhibited sharp peaks in the first four hours of our sample, as shown in the plot below:

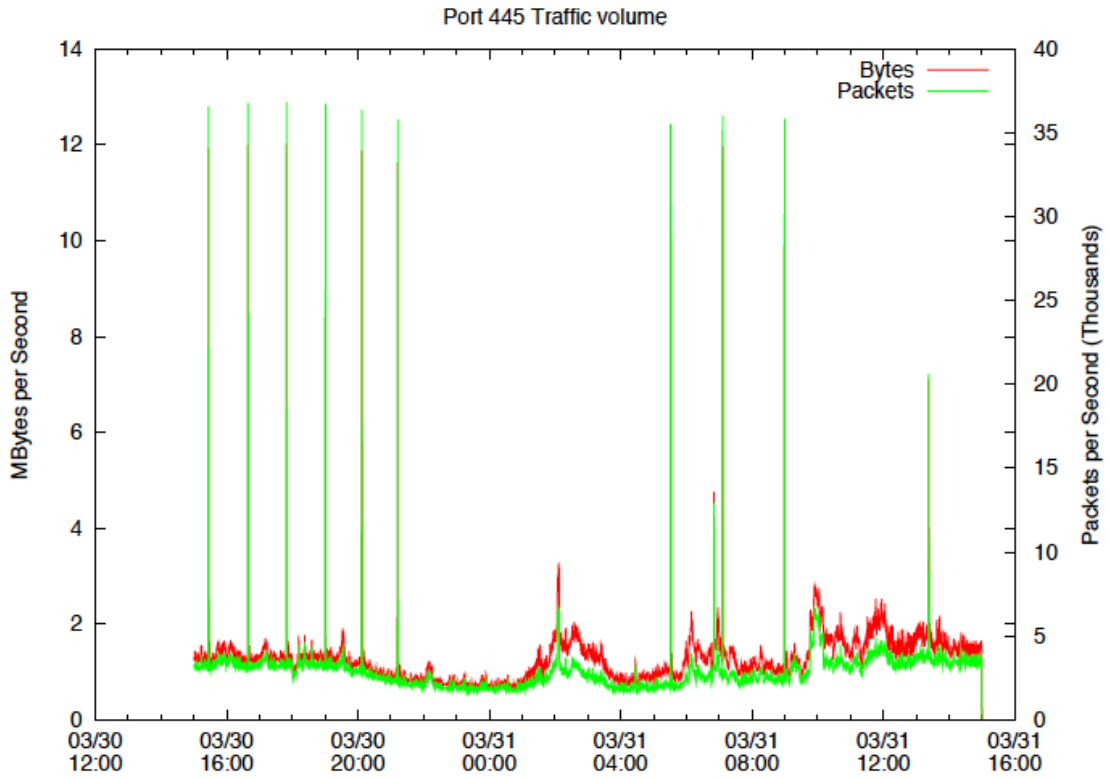

Figure 14: TCP/445 traffic, megabytes per second, for the whole day sample in the MAWI set

TCP/445 scanning was characteristic of the first three versions of Conficker (a,b,c), but the new version used UDP/445 to scan for new victims. We looked at the MAWI sample for this traffic, and found that there was relatively little UDP/445 traffic in the entire set, though it was beginning to appear at the end of the second day.[19] Comparisons of the two traffic counts, either as packets or bytes, were impossible due to scale, since the UDP traffic was such a small percentage of the whole. So Conficker.d, which uses UDP, was not widespread in this network sample by March 31, since we would have seen more UDP scanning. Byte counts started to rise above zero after 8:00 PM Pacific, but just slightly.

DATE	HOUR	RECORDS	BYTES	PACKETS

[19] This is consistent with the timing of the new version of the worm.

3/31/09	7:00:00	0	0	0
3/31/09	8:00:00	0	0	0
3/31/09	9:00:00	1	76	1
3/31/09	10:00:00	0	0	0
3/31/09	11:00:00	4	558	7
3/31/09	12:00:00	6	1632	18
3/31/09	13:00:00	6	947	10
3/31/09	14:00:00	2	577	6

Figure 15: UDP/445 counts by hour for the whole MAWI set

Our Network

Our group chose the 193.52.0.0/16 network for closer analysis. This network appears to offer mostly web services, with the highest percentage of its source traffic being http, squid proxy and two proxy alternates.[20] The presence of Gnutella (6881) and FTP (21) also suggest that some of this network could belong to an online service provider.

Source Port	Records	%_of_total	cumul_%
80	2282677	28.442365	28.442365
3128	1703379	21.224259	49.666624
3127	1395141	17.383585	67.050209
3124	1349260	16.811904	83.862113
25	104431	1.30122	85.163332
21	56677	0.706201	85.869533
445	48678	0.606532	86.476065
6881	46743	0.582422	87.058488
139	35380	0.440838	87.499326
22	22096	0.275318	87.774644

Figure 16: Top ten source ports in the 193.52.0.0/16 network

Our network also sent out lots of web and DNS traffic destined for other networks. Much of this network's source traffic to http/https came from ephemeral ports evenly distributed among the range of top 20 source ports derived from rwstats, which indicated to us that the outbound queries for most of the network were "normal" web transactions, outbound GETs to other networks, when analyzed as the *source* to the MAWI set as a whole.

[20] Technically, 3124 is "beacon port on the IANA list, and 3127 is "ctx bridge."

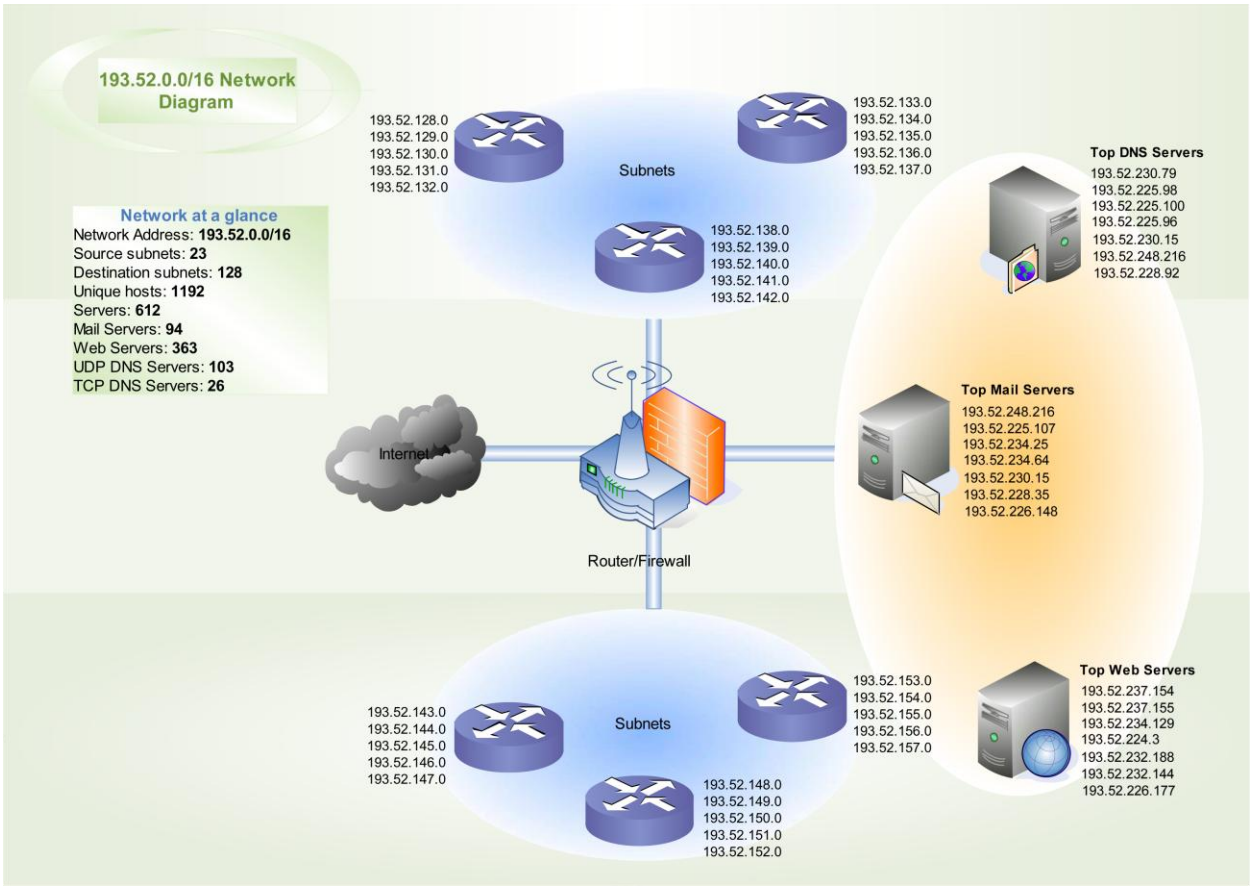

Figure 17: Network Diagram for the 193.52.x.x network, based on our analysis

We detected 1,192 unique hosts in the network, with the highest number of flows going to two hosts: 193.52.232.188 and 193.52.232.189. Of these hosts, there were 612 servers[21], estimated by determining which machines responded to SYN traffic with the "SA" flag set. There were 128 destination subnets and 23 source subnets, 94 mail servers, and 363 web servers. 26 servers generated TCP DNS traffic and were likely serving as authoritative DNS sources. We also calculated that the network generated 36,008,817 total flows, or 21.31% of the whole MAWI sample. Not only was packet volume on this network relatively high, but the ratio of bytes/packet accounted for nearly 30% of the MAWI sample at the lowest interval, and at least 20% for most of the other interval "slices" calculated by rwstats' overall count of the flows. A points analysis of the major Class Bs put this network at the top of the overall MAWI sample:

[21] We use this term loosely. A better phrase would be "hosts listening on a port to which a SYN packet was sent."

Ranking of Top Networks, By Flows, and by Packets

	By-Flows		By-Packets		Points	Rank
	src	dst	src	dst		
203.11	1	1	5	6	13	2
193.52	2	2	1	1	6	1
144.44	3	8	2	5	18	4
149.162	4	9	4	13	30	6
173.94	5	4	3	2	14	3
133.44	6	5	14	18	43	8
212.158	7	14	20		62	13
192.134	8	6	7	7	28	5
174.63	9	7		17	54	11
149.92	10	10			62	14
192.136	11		12		65	15
192.42	12	12	6	8	38	7
11.133	13				76	24
212.154	14	13			69	16
83.34	15				78	28
203.192	16				79	31
149.101	17	11	11	10	49	9
36.87	18	16	17	3	54	12
11.4	19				82	35
45.174	20	18		16	75	22

Figure 18: Ranking the Class Bs by flow and packet volume

As shown in the table below, 193.52.237.154 and 193.52.237.155 had the highest proportion of byte traffic, even though their flow records were smaller. Both of these hosts had traffic at non-standard ports, indicating possible peer-to-peer or Conficker traffic. Given the size of the byte traffic, rather than the number of flows, it's likely that it was P2P, though our network was the second-highest source of TCP/445 traffic to other nets.

sIP	Bytes	Packets	Records
193.52.232.188	8,127,573,453	16,788,131	2,225,728
193.52.232.189	9,446,741,826	15,846,460	1,735,649
193.52.232.177	7,353,419,760	12,953,217	1,555,569
193.52.237.154	111,139,086,293	79,578,962	550,720
193.52.237.155	101,242,886,140	74,303,753	513,613
193.52.224.3	56,284,917,170	45,057,576	264,830
193.52.234.129	44,396,962,674	30,789,641	174,286
193.52.226.144	1,261,441,169	1,392,027	80,266
193.52.229.232	2,984,888,480	2,913,505	60,969
193.52.229.233	7,709,648,630	5,655,579	60,753
193.52.229.164	19,772,439	335,511	52,085
193.52.229.240	826,845,167	1,287,134	47,964
193.52.228.35	36,241,979	286,294	46,293
193.52.234.138	2,096,796	48,584	45,892
193.52.230.218	32,144,665	254,770	30,387

Figure 19: Top fifteen traffic generators in our network

Hourly packet traffic for the network was relatively stable, with the highest number of flows recorded at 16:00, and the lowest[22] at 22:00 Tokyo time. The network averaged 334,398 flows per hour, or a very respectable 92.8 flows/second,[23] and passed an average 1.218 million packets an hour. The two o'clock hour was also the time of its largest byte traffic.

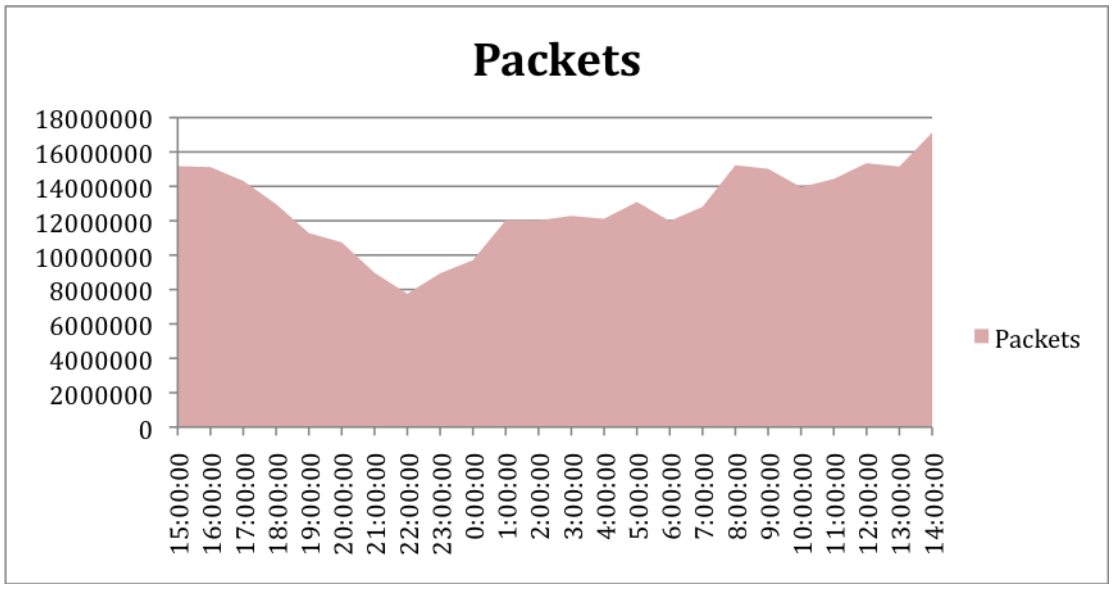

Figure 20: Hourly packet volume for the 193.52.0.0/16

[22] Here we have lopped off the last hour of the traffic, because there were only 77 flows recorded, and we felt the measurement had been distorted by the arbitrary break of one calendar day into another.
[23] The CMU network's core currently averages ~40 flows/second.

Though there was some variation in the hours where byte volume exceeded packet volume, the two measurements were relatively synchronous over the 24-hour period. Because of the difference in scale, these two metrics are compared logarithmically below:

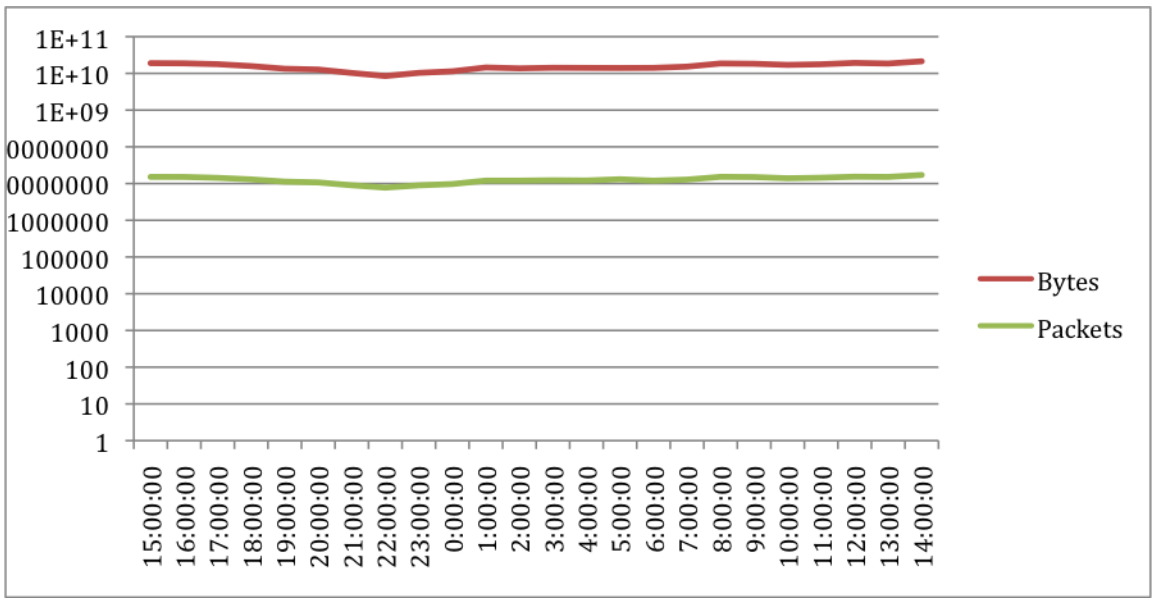

Figure 21: Logarithmic ratio of bytes to packets on our network

Our network sent a substantial number of TCP/445 packets to other networks, suggesting that a good number of clients attached to this "ISP" were infected.

Networks receiving port 445 traffic from 193.52.0.0
SIZE: 48678 records for 122 unique keys

INPUT dIP	Packets	%_of_total	cumul_%
193.46.0.0	9545	11.278906	11.278906
11.40.0.0	9198	10.868872	22.147778
51.25.0.0	7036	8.314131	30.461909
52.4.0.0	6520	7.704397	38.166306
51.254.0.0	6205	7.332175	45.498482
216.69.0.0	5510	6.510924	52.009406
195.17.0.0	5466	6.458932	58.468338
138.140.0.0	4783	5.651861	64.120198
202.143.0.0	4683	5.533695	69.653893
193.39.0.0	4549	5.375353	75.029246
195.184.0.0	4375	5.169745	80.198991

Figure 22: Networks receiving our port 455 traffic

TCP/455 traffic also followed a pattern of abrupt spikes when measured as packets or bytes per second.

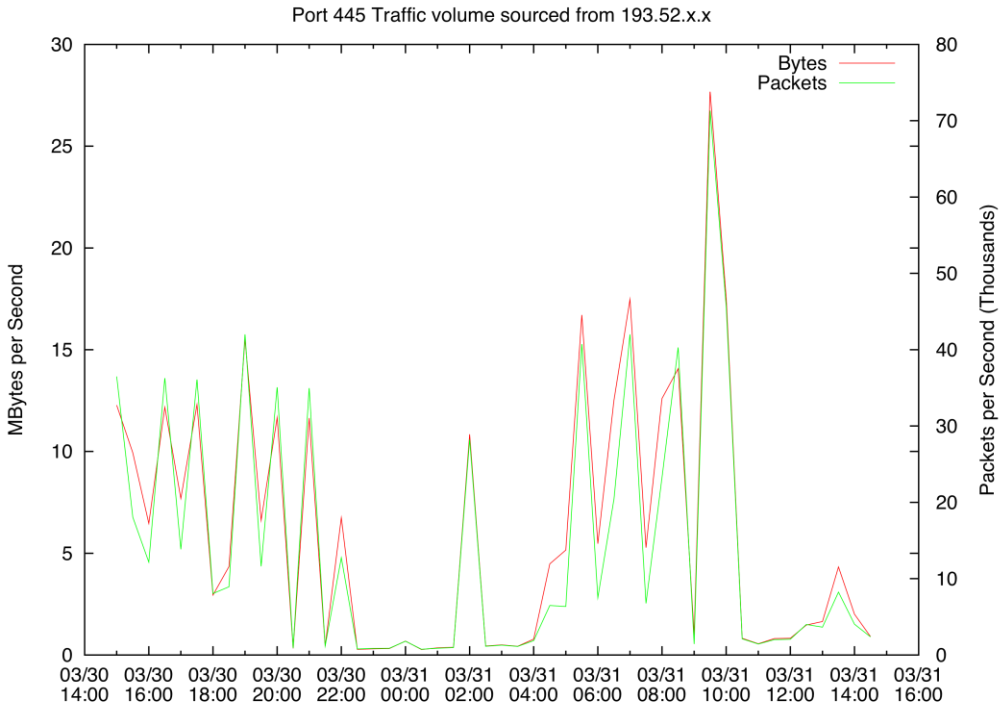

Figure 23: Port 445, packets per second, from our network

In fact, the pattern of DNS, web and SMB traffic **in concert** most closely resembles a diagram of Conficker.a included in SRI's analysis of the worm (our drawing is beneath). [24] If there was infection on our network, then this resemblance could indicate that the worm was still predominantly type "A", and it would be looking for updates from a web host that had long since been removed from the internet. Getting those hosts to version "B" and beyond would involve manual (PnP) methods, or a wave of inbound attacks from the more advanced versions of the worm. But the resemblance is slight enough to make this supposition just conjectural.

[24] SRI International Technical Report: An Analysis of Conficker's Logic and Rendezvous Points (Porras, Saidi, Yegneswaran), 4 February, 2009, page 26.

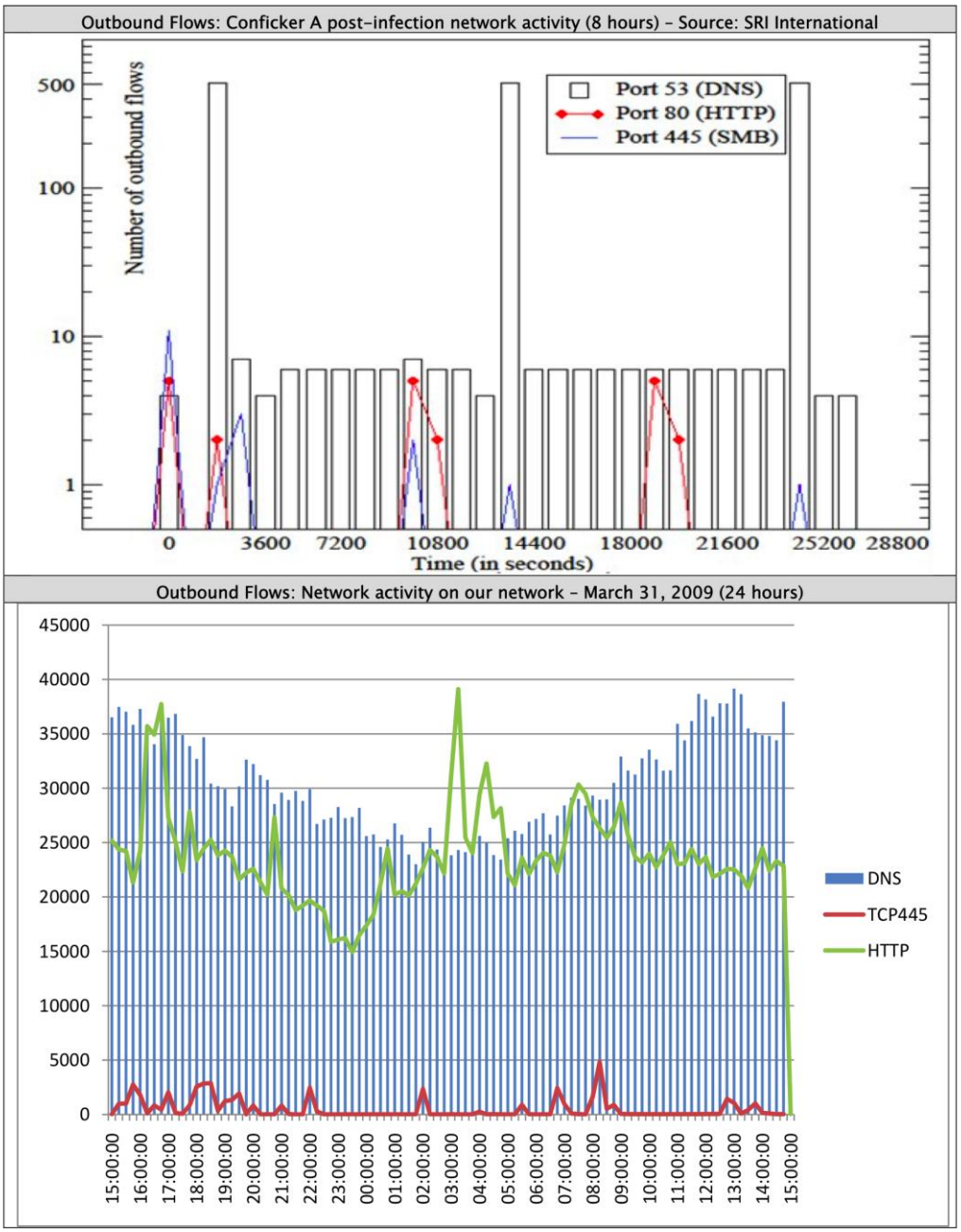

Figure 24: Comparison of SRI traffic peaks in Conficker A to our outbound flows of the same type

Our conclusion, then about 193.52.0.0/16 is that it was not wildly infected, though some portion of the machines connecting through it were. Its sheer size magnified the effect it had on other networks, but it wasn't necessarily an "attacker." The sharp, once-an-hour spikes in the illustration above are indicative of Conficker, especially the long spike at 10:00, and this might be harbinger of impending problems on this network the next day.

Mitigation strategies for our network

The primary mitigation strategy against Conficker is patching, since the worm exploits a vulnerability that has since been patched. If the network owner has a private VLAN where they can sequester infected hosts, the unpatched hosts can be identified with a Nessus or nmap scan and transferred to this network, assuming that the Microsoft Update site could be proxied from there. As a large ISP, our network would be hindered by its size, and it's unlikely that round-the-clock scanning would prevent the spread of infection among its subscribers. Networks that are not required to provide service without discrimination, like corporations, can elect to block port 445 at the border, especially from subnets that are not part of their enterprise. But an ISP might have difficulty chopping itself into different service profiles for the many different kinds of customers it serves. User education, in the form of notices to its subscribers about the worm and the necessity of patching, coupled with the notice that it reserves the right to suspend service in the case of infection, might also contribute to a greater number of patched machines.

Heuristic and signature-based intrusion detection can identify Conficker symptoms, and since Snort signatures for Conficker have been available at Tenable and Emerging Threats[25] since version A, these can be incorporated into a detection strategy. Any signature that detects excessive scanning at port 445, or rapid and excessive DNS querying would pinpoint a Conficker-infected host.

[25] Tenable is the maintainer of the "official" snort, and sells subscription services to signatures. Emergingthreats.net publishes user-submitted signatures that are often "faster", because they're not subject to extensive testing.

DNS servers can be "firewalled" with iptables, which can employ rules to drop packets from a source address when they exceed a threshold. Large networks can be subnetted into VLANs, and firewall rules applied to inter-subnet exchanges of certain traffic. Once divided, VLANs could also be isolated from the internet completely if the number of infected hosts warrants this drastic intervention.