Ransomware Identification Using Network Traffic

Author: Pete Garvin, pgarvin@protectus.com

Abstract

Ransomware is a type of attack software that uses encryption to make data unavailable for the purpose of collecting a ransom payment. It is a serious problem in today's threat landscape and there is no reason to believe it will go away anytime soon. Many individuals and organizations who were unable to recover data from backups have been forced to choose between either losing data or making a ransom payment. Both network-based and host-based solutions are available to help detect and stop ransomware. This paper explores the possibility of identifying ransomware by the network traffic patterns it generates. Traffic from the infectected endpoint bound for both the internet and the intranet is considered. Experiments show that certain ransomware variants are very noisy and do generate observable traffic patterns. Based on traffic data collected from ransomware, possible detection ideas are reviewed. Understanding the threat is a necessary part of a well-executed network security plan and allows for the best possible defense. The ideas presented can provide insight for additional layers of defense against ransomware.

1. Introduction

On March 16, 2017, a SANS webcast entitled 'Most Dangerous New Attack Techniques' occurred. This useful webcast continued a keynote panel discussion started during the previous month's RSA conference. At one point in the webcast, techniques to defend against ransomware were discussed including system and network hygiene, watching network shares, DNS sinkholing, honeypots, and web gateways (SANS, 2017). The importance of multiple layers of protection was also stressed. All of these defense techniques and other not listed here are good and should be considered for use by system / network administrators. The topic of this paper was inspired by the webcast and the desire to explore other possible methods to identify the presence of ransomware.

Many excellent resources are available to learn about and understand the concepts around ransomware prevention, protection, and behavior. "Ransomware is effective because it uses capabilities that are already on the operating system." (Bradley, 2016). This fact makes it difficult to distinguish ransomware-generated traffic from user-generated traffic and makes defense against it more difficult. When the ransomware is performing user-like actions such as reading and writing files on a network drive, then a cursory look at the corresponding traffic may show nothing out of the ordinary.

The Incident Response Methodology taught as part of the SANS GIAC Certified Incident Handler course contains six steps: prepare, identify, contain, eradicate, recover, and lessons learned. This paper will concentrate on techniques that are most useful during the identification phase of a ransomware incident. Furthermore, the techniques discussed will be network-traffic focused. In other words, answers to questions like 'What network traffic does a ransomware infection generate?' and 'Are there any observable, detectable patterns in ransomware-generated network traffic?' will be explored. For the ransomware to be generating network traffic, it must be installed or at least starting to install. So the focus will be on traffic generated after an infection has started to occur. This assumes that any defensive countermeasures already in place have been circumvented and the endpoint has downloaded some ransomware which has started to execute.

A post-infection focus is not being selected because that is the ideal time for identification. Rather, preventing the infection in the first place is of course prefered. But much attention has been given, and rightfully so, to preventing the infection from ever occurring. Anti-malware software and other countermeasures can be effective. The effectiveness of Windows Defender was demonstrated by the need to disable it, as shown in the Appendix A Build Procedure, prior to executing ransomware samples and generating network traffic to study.

It may be tempting to think that detecting ransomware in the post-infection timeframe is pointless but consider three factors that demonstrate this is not the case. First, it can take time for the encryption to occur depending on the circumstances. The infected workstation needs to read each file, perform the encryption, and then write the file back. "Depending on the speed of the infected machine and the resources to which it has access, the actual encryption process can take days." (Saiyed, 2016). This can allow time for identification and containment before all the damage is done. Network, workstation, and server performance are all factors. Second, deploying layers of security is a fundamental concept and post-infection, network-centric detection provides another layer. Third, the infected workstation will need to be identified in any case so it can be taken offline and rebuilt.

Much attention has also been given, again rightfully so, towards creating signatures to match internet traffic generated by ransomware. "If one can disrupt the communication channel between the workstation impacted and the command and control server, the malicious activity can often be intercepted." (Bradley, 2016) Unified Threat Management (UTM) firewalls, Intrusion Detection Systems (IDS), and other signature-based countermeasures are often effective, and should certainly be used.

To demonstrate how countermeasures commonly scrutinize internet traffic, a case-insensitive search for the word 'ransom' in a recent Emerging Threats Open ruleset was performed. The results show how ransomware detection signatures are often written with the assumption that at least one endpoint address will be in the $EXTERNAL_NET. Each administrator can of course define the $EXTERNAL_NET and $HOME_NET variables according to their needs. However, the default and common definitions are for '$HOME_NET'

to include subnets defined in RFC 1918 (Rekhter et al., 1996) and $EXTERNAL_NET to include all other subnets. The search results yield 462 hits with the following breakdown:

Rule count	Source network		Destination network	Comment
200	$HOME_NET	->	[IP list]	Known CnC server groups
119	$HOME_NET	->	$EXTERNAL_NET	Checkins, beacons, CnC traffic *Note1*
115	$HOME_NET	->	any	DNS requests on port 53/udp *Note2*
27	$EXTERNAL_NET	->	$HOME_NET	Checkins, landing pages, certificates
1	$HOME_NET	->	$HOME_NET	SMB related

Table 1 - Emerging Threat Open rules breakdown by source & dest networks

*Note1: Includes 8 rules for SMB port 445/tcp; these could be moved into another category.

*Note2: These rules could be used on traffic with internal or external DNS servers.

In Table 1, rows one, two, and four are specifically for internet traffic. Row three could be applied to traffic between workstations and an internal or external DNS server. These numbers in no way reflect anything lacking in the ruleset. Rather, they reflect a common practice of feeding IDS / IPS systems with internet traffic streams instead of intranet traffic streams. Most organizations don't have the resources to monitor all internal traffic. So if any traffic monitoring is done, it is typically on the traffic going to and coming from the internet. While the scope of this paper includes both internet and intranet ransomware traffic, the focus will be on understanding traffic patterns and not on writing detection signatures.

While watching the SANS webcast mentioned earlier, the thought occurred that detecting ransomware-generated internal broadcast traffic could be a useful technique. If such traffic occurs, it may allow network administrators to passively listen for the suspect broadcast traffic throughout an entire subnet. A study of post-infection ransomware traffic will show what broadcast or other network traffic occurs, review the traffic for patterns, and provide an opportunity to consider how it can be used for detection.

2. Capturing Ransomware Traffic

To understand ransomware traffic, decisions need to be made about where on the network to capture packets. Once the traffic is captured, then traffic analysis can provide insight into ransomware behavior. Some general traffic capture concepts will be reviewed before ransomware traffic capture is performed. After a selecting observation points on the network and creating a lab setup, the stage will be set to for safely collecting actual traffic samples in a lab environment.

2.1 Capturing Network Traffic

Network traffic zips around invisibly on cables or through the air on wifi networks. As long as everything is working properly and the network is running smoothly, we don't think much about it since the traffic is out of sight and out of mind. However, there are times when a need arises to capture, record, and analyze network traffic. Experience shows that such a need often originates in the areas of security, performance, or troubleshooting. During those times, the additional insight into network activity provided by traffic capture and analysis can be very useful or even necessary.

There are many types of networks and protocols. The scope of this paper is Internet Protocol (IP) on wired ethernet-based networks. On these types of networks, common packet sniffers or network protocol analyzers include tcpdump (Tcpdump, 2017), Wireshark (Combs et al., 2016), and Tshark (Combs et al., 2016). These tools allow for real-time analysis or can save traffic data to disk typically in the pcapng format. With the right equipment and setup, these tools can also be used on wireless networks. To remain focused on the topic at hand, packet sniffing will be done only on a wired network.

The simplest case of observing network traffic is on a computer to which login-access is available. Simply install the tool of choice and start a packet capture. The results of a packet capture can quickly become overwhelming for any significant volume of network traffic but Wireshark, for example, does provide filtering, sorting, and other features to aid analysis. Tools to help visualize and understand network traffic over time are available (AOL, 2017). In the interest of full disclosure, the author has developed such a tool called the Sentry. The analysis

performed on captured traffic was done using using a combination of Tshark, Wireshark, and the Sentry. The Sentry allows for more efficient and productive traffic analysis but is not required to reproduce the results presented in this paper.

When performing a packet capture, a network interface must be specified. Some computers will have multiple network interfaces. On Linux systems, network interface names can be customized but are traditionally called eth0, eth1, eth2 and so on. On Windows systems, interfaces are referred to using both an intuitive text name like 'Local Area Connection' and a more cryptic name like '\Device\NPF_{A2D5...782A}'. When multiple network interfaces are present on a computer, some trial-and-error may be needed to identify the correct network interface for monitoring.

If login-access to the computer of interest is not available or if it is necessary to simultaneously observe traffic from multiple computers, then another option is using a network tap to intercept the traffic of interest and send a copy of the packets to a monitoring device running a packet sniffer. A disadvantage of this approach is that a brief network outage occurs when the tap is installed. Figure 1 shows installation details for a typical copper network tap with four physical network ports. Not show is the power source needed to activate the Tap A and Tap B ports.

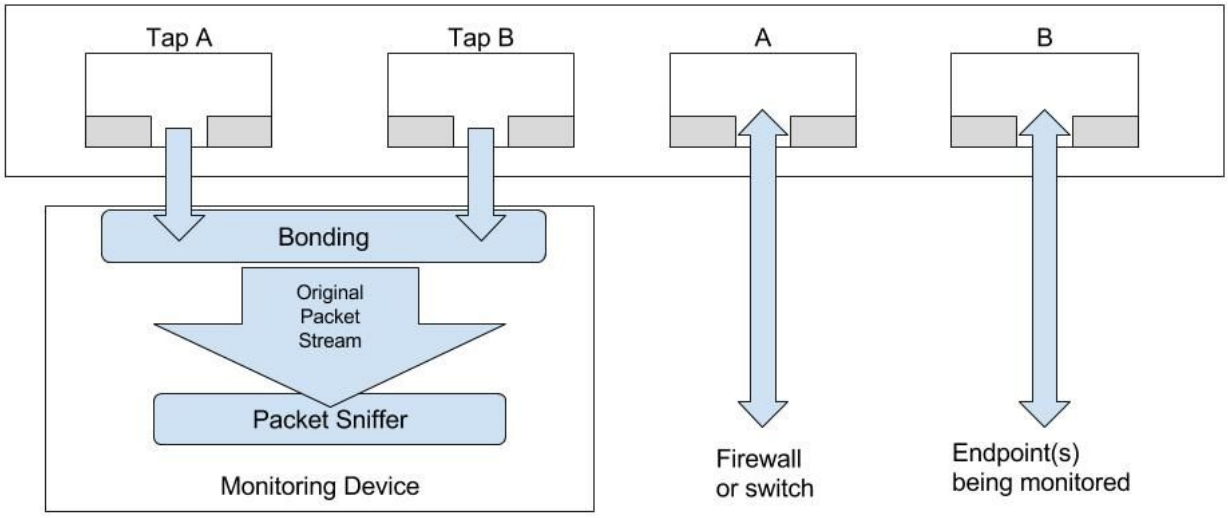

Figure 1 - Copper network tap example

Two ports, commonly labeled A and B, provide a pass-through and are inserted in-line on the ethernet cable to be monitored. The other two ports, commonly labeled Tap A and Tap B, each provide a copy of the packets flowing in one direction. Output from Tap A contains half of the conversation and output from Tap B contains the other half of the conversation. For this type of tap, the monitoring device needs to bond the two halves back together before performing analysis on the complete traffic conversation. Note that monitoring device interfaces being fed by the tap do not need IP addresses. Some taps, not shown here, will include the bonding logic and reproduce the original packet stream on a single physical output port.

Other network taps are available that are similar to the copper tap discussed above, except that a copy of the traffic flows to the monitoring device via a USB cable rather than two ethernet cables. Some of these taps appear physically similar to a brick-like Power Over Ethernet (POE) injector sometimes used in Voice-Over-IP systems. However instead of injecting power like a POE brick, these taps extract a copy of the traffic flowing through the tap and send it out over the USB cable. This is the type of tap used in the lab experiments to capture ransomware traffic.

If access is not available to the ethernet cable(s) of interest, then another option is using a mirror or SPAN (Switched Port ANalyzer) port on a managed network switch. Some firewalls with integrated switches also provide this feature. Just as the name implies, a mirror port sends a copy or mirror of all the traffic flowing on physical switch port X to physical switch port Y. A monitoring device can then be connected to physical switch port Y to capture the traffic of interest. One consideration is that copying traffic to a mirror port will consume some processing resources on the switch. First-hand experience with using mirror ports on many switches has show this to rarely be a problem but it is possible for an overworked, underpowered, and probably older switch to perform poorly when a mirror port is enabled. Note that the monitoring device interface being fed by switch mirror port Y does not need an IP address to capture traffic.

2.2 Ransomware Traffic Overview

The concepts described above are now applied to observe ransomware traffic on a lab network representing a typical network architecture. To begin with, imagine that an

unfortunately all-too-common scenario has occurred on your network - a successful phishing attack resulting in ransomware executing on an end-user's workstation.

To complete its work, ransomware typically generates some network traffic. "The first step after installation of ransomware is to contact the command-and-control (C&C) server in order to get further instruction or encryption key."(Mehmood, 2016) Since this paper's scope is post-infection traffic, a simple, network-centric diagram will be a useful aid in understanding ransomware's network activity. This diagram will be useful not only to understand the network traffic that ransomware could generate but also how to observe that traffic. Figure 2 shows a network-centric view of a ransomware attack.

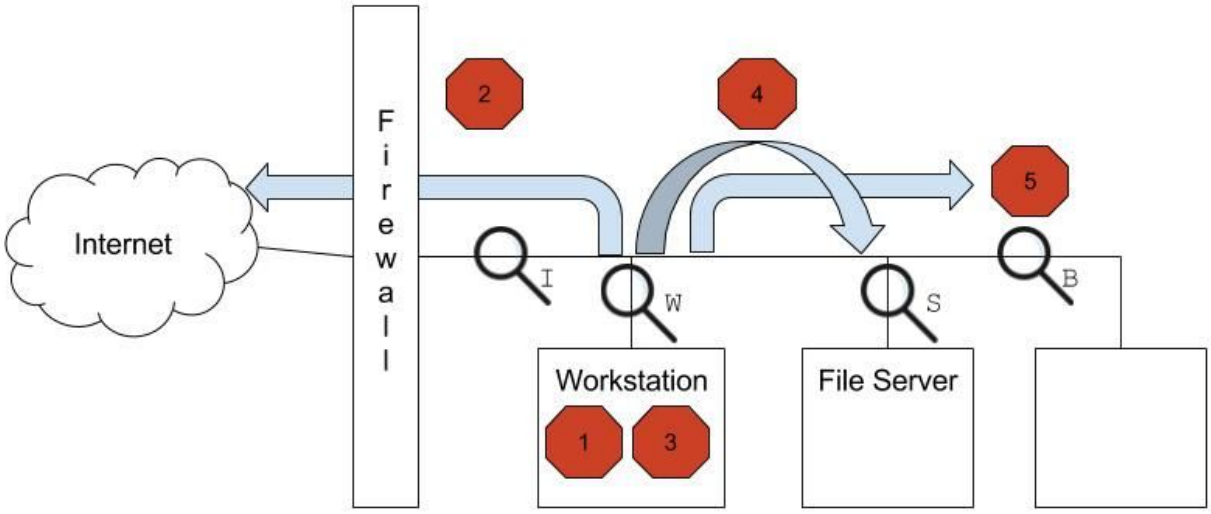

Figure 2 - Network-centric view of a ransomware attack

This simplified diagram represents essential events with a red hexagons and also represents places of interest for observing network traffic with a magnifying glass icon. For simplicity, assume all devices are hardwired with copper ethernet cables. A network switch is not shown but obviously would be present either embedded in the firewall or as a stand-alone piece of network infrastructure.

The essential events (red hexagons) are labeled numerically and include:

- Event 1 - An end-user falls victim to an attack. Some phishing attacks are so authentic-looking that even a trained professional, who really knows better, can have a

bad day and click before thinking in the rush to meet a deadline. The human element is always present.

- Events 2 - The ransomware might invoke one or more internet resources. An exploit kit is often used to initiate download of additional malicious code.
- Event 3 - The downloaded ransomware starts to execute on the workstation. This event includes the ransomware encrypting files locally on the workstation hard drive.
- Event 4 - The ransomware interacts with a mapped drive on a file server. These interactions take the form of network-based request, usually Server Message Block v2 or v3 (SMB2) protocol, to list, read, and delete the unencrypted, original files on the mapped drive. Request to write encrypted files also occur.
- Event 5 - The ransomware may search for unmapped drives on which the user may have write permissions. If this event is successful, event 4 is repeated on the newly-discovered network drive.

These essential events are labeled in what could be called approximate chronological order since it is possible for some variation in the sequencing. For example, command and control communications (event 2) between the ransomware and internet-based resources could occur either before or after file encryption (event 4) occurs.

The points of interest for observing network traffic (magnifying glass icons) are labeled alphabetically and include:

- W - Workstation traffic; includes all traffic generated by the ransomware.
- I - Internet traffic; normally captured immediately inside the firewall and includes any traffic sent to the gateway.
- S - Server traffic; typically SMB or SMB2.
- B - Broadcast and / or scan traffic; created by Windows networking if network drives are searched for; does not flow between router-separated network segments.

Since the traffic of interest is being generated by the ransomware, all relevant traffic must flow past point W. Subsets of the ransomware traffic are observed at the other points. The following pseudo-equation shows how traffic is related between these different points on the network:

Point W traffic = Point I traffic + Point S traffic + Point B traffic

This pseudo-equation is not mathematically or technically precise since broadcast traffic can in reality be viewed at multiple observation points. In spite of this, the pseudo-equation is still useful by providing a divide-and-conquer strategy for separating the total traffic from point W into smaller subsets for analysis at points I, S, and B. As a simplifying assumption, broadcast traffic will only be analyzed at Point B and is ignored at points I and S. Another simplifying assumption is that any gateway traffic bound for the firewall's internal interface IP address will be grouped with the 'internet' traffic even though it terminates at the gateway and does not actually flow onto the internet. In practice, traffic was not actually captured at points I, S, and B. Rather, it was captured at point W and then display filters were used to identify traffic that would have flowed past other points in the network.

2.3 Lab Setup

Three laptops were used to create an isolated lab environment for ransomware traffic capture. A simple network architecture similar to that shown in Figure 1 was used. Figure 3 shows the lab setup.

Figure 3 - Lab setup - from left to right: WRKSTN-1, SERVER-1, SENTRY

A laptop named WRKSTN-1 represented the workstation. Another laptop named SERVER-1 represented the file server. WRKSTN-1 and SERVER-1 were built using a Windows 10 Enterprise CD with a 90 day trial license which can be legally downloaded at no cost.

To emulate a real environment, folders named ‘mapped’ and ‘unmapped’ were created on SERVER-1 and populated with various file types including .txt, .docx, .png, .pdf, and .odt. The network was browsed from Windows Explorer on WRKSTN-1 and the ‘mapped’ drive was mapped as drive Z. This allowed for ransomware on WRKSTN-1 to find a mapped drive and also have access to an unmapped drive if a search was performed.

Multiple laptop rebuilds were needed so a build procedure, available in Appendix A, was created and refined to ensure consistency over time. A third laptop (SENTRY) was used as a monitoring device. It ran Tshark, the terminal-based wireshark program, to capture all packets at point W and save them to a pcapng capture file for later analysis. Other network traffic ingest software was also run on the SENTRY laptop to allow for visualization of traffic which aided analysis.

All three laptops were connected using a small, unmanaged switch which can be seen in Figure 3 hanging above the middle laptop. Also plugged into the switch was an ethernet cable (yellow)coming from an internet-connected router. A network tap was installed between WRKSTN-1 and the switch to capture all traffic to and from WRKSTN-1. Location of the tap installation represents point W as shown in Figure 2. The tap sent a copy of all packets to the monitoring device via a USB connection and also received power from this same cable. Not shown in the picture is a firewall used to safely contain the entire lab setup on an isolated subnet.

The lab setup was made as realistic a possible to bypass any mechanisms the ransomware might use to evade analysis. “cyber criminals are developing techniques with which to circumvent the security measures put in place by malware analysts leveraging detection-based sandboxes.” (Keragala, 2016) This problem was avoided by using actual hardware in the lab instead of virtual environments.

For each ransomware variant, captured traffic was stored in a pcapng file for subsequent analysis. Wireshark display filters were applied during analysis to create views of traffic that

would have flowed past the individual points of interest - I, S, and B. Tshark was run from a prompt on the SENTRY laptop using the following command:

```
tshark -i bond0 -B 10 -b filesize:65536 -b files:50 -w /tmp/variant_name
```

Looking at each of the command's component parts:

```
tshark . . . . . the terminal-based version of wireshark
-i bond0 . . . . . listening interface created by monitoring device
-B 10  . . . . . increased buffer size to minimize dropped packets
-b filesize:65536. . . max capture file size
-b files:50. . . . . up to 50 ring-buffer capture files
-w /tmp/file_name. . . output capture file for raw packet data
```

2.4 Test Details

Ransomware samples were obtained from Malware-Traffic-Analysis.net. (Duncan, 2017) Many thanks to Brad Duncan for all his good work providing this website and blog. It is an excellent site for understanding all types of malware and includes analysis, code samples, and capture files. While the provided pcap files are certainly useful, only ransomware samples, not pcap files, were downloaded from malware-traffic-analysis.net onto WRKSTN-1. New capture files for both internet and intranet traffic were created from the lab network as the downloaded ransomware samples executed on WRKSTN-1.

When capturing packets, it is best to plan ahead and eliminate as much unrelated, uninteresting traffic as possible before even starting the packet sniffer. Analyzing only the traffic of interest, ransomware-generated traffic in this case, really speeds analysis. Having to extract ransomware traffic from the middle of a large volume of Windows Update traffic can result in a slow and tedious analysis. It is not a simple as filtering out an IP address from which the updates were downloaded. In today's cloud-based, distributed world, Windows Updates and other traffic such as Microsoft telemetry, will come from a range of IP endpoints having a wide variety of domain names and Autonomous System Numbers (ASN). For this reason, disabling Windows Update is part of the build procedure. Even so, capturing some amount of unrelated traffic is often unavoidable as is discussed when analyzing traffic at point I.

The build procedure also includes disabling of Windows Defender. This is for obvious reasons since we want the ransomware to run and not be caught or stopped. Note that after a

reboot, Windows Defender will re-enabled and needs to be disabled again. Another step in the build procedure is disabling of IPv6. The method used does not completely disable IPv6 but does force the ransomware-generated IP traffic to be IPv4 which makes the packets easier to read and helps to speed analysis. The build procedure in Appendix A is the fruit of much trial and error and is optimized for quick rebuilds.

Either from user-initiated activity or automatically via the Windows operating system, network traffic routinely occurs to discover and identify resources on network drives. Since this is a normal part of network traffic, a 'worst case' scenario was used for the testing. This means that the build procedure included a step in which the mapped network drive on SERVER-1 was viewed on WRKSTN-1 immediately before the traffic capture was started and the ransomware was executed. This simulated the case of a ransomware attack occurring when the workstation had just used the network drive so little or no discovery protocols would be needed. This approach should minimize the ransomware-generated discovery traffic and is therefore consider the 'worst-case'.

In the results presented below, the malware-traffic-analysis.net blog post from which the ransomware sample was obtained is shown. Also shown is the appendix in which a traffic summary is listed and a link to the relevant pcapng file for those wanting more details than are presented in the traffic summary. Certain protocols and their related acronyms appear routinely in the captured traffic. Appendix B provides some information about these protocols so only their acronyms will be used when discussing results.

2.5 Revenge

Blog post: 2017-03-15 -- EITest Rig EK sends Revenge Ransomware (Duncan, 2017)

Capture name: http://www.protectus.com/download/files/revenge.pcapng

Traffic summary: Appendix D

Revenge ransomware started with HTTP requests to an IP address in Poland. Files local on WRKSTN-1 and on the mapped drive were encrypted. SMB was used for authentication and SMB2 for file access. Files on the unmapped drive were untouched. File names changed in the following way:

```
123.txt ⇒ 24D140319C7D375BFB97EFBF35DDF9E7.REVENGE
```

For each target file, the SMB2 command sequence was:

Read original file; Delete original file; Write encrypted file

All SMB2 traffic occurred within a single TCP connection. No out-of-the-ordinary broadcast traffic was observed.

2.6 Sage 2.0

Blog post: 2017-01-23-- Ongoing malspam campaign spreading Cerber and Sage 2.0 ransomware (Duncan, 2017)

Capture name: http://www.protectus.com/download/files/sage.pcapng

Traffic summary: Appendix E

Sage ransomware traffic started with DNS request for some unusual, random-looking domain names which did not successfully resolve to an IP address. These were followed by 38,389 port 13655/udp packets totalling 7.3MB sent to 7698 unique internet IPs of which 1269 were unreachable. SMB2 was used both for authentication and file access. File names were changed in the following way:

`123.txt ⇒ 123.txt...` then `123.txt... ⇒ 123.txt.sage`

Encrypted files were first written to a filename similar to the original but suffixed with three dots. After a file was written, the filename suffixed with three dots was renamed to have a suffix of 'sage'.

For each target file, the SMB2 command sequence was:

Read original file; Write to new file; Delete original file; Rename new file

Additional DNS requests subsequently occurred for a total of 103 requests with no IPs successfully resolved. All SMB2 traffic occurred within one TCP connection. No out-of-the-ordinary broadcast traffic was observed.

2.7 CryptoShield

Blog post: 2017-02-28 -- EITest Rig EK from 81.177.140.75 sends CryptoShield ransomware (Duncan, 2017)

Capture file: http://www.protectus.com/download/files/cryptoshield.pcapng

Traffic Summary: Appendix F

CryptoShield traffic started with an HTTP post to an IP address in Turkey with no prior DNS request observed. SMB was used for authentication and SMB2 for file access. File names changed in the following way:

```
123.txt ⇒ .GKG.[R_SP@INDIA.COM].ID[9C7D375B607C01B2].CRYPTOSHIELD.
```

For each target file, the SMB2 command sequence was:

Read original file; Delete original file; Write encrypted file

All SMB2 traffic occurred within one TCP connection. No out-of-the-ordinary broadcast traffic was observed.

2.8 Other Variants

Several ransomware variants were tested that either did not successfully run or only encrypted local files and produced no noteworthy network traffic. These included an un-named Matrix variant, Locky, and Spora. In some cases, significant Windows update traffic occurred around the time of the attempted infection even when the test infection was repeated multiple times. In these cases, it was concluded that the operating system somehow detected and stopped the infection even though Windows Defender was disabled.

3. Analyzing Ransomware Traffic

In the subsequent traffic analysis, WRKSTN-1 is IP address 192.168.2.14 and SERVER-1 is 192.168.2.101. As discussed previously, the analysis will focus on identification of traffic patterns and not creation of IDS signatures. Certain traffic was observed originating from WRKSTN-1 both around the time of some ransomware infections and when the endpoint was uninfected. This traffic is summarized in Appendix C and is considered expected traffic. As such, it was not included in the ransomware traffic analysis unless it occurred in unexpected quantities and has generally been removed from the results summarizing specific ransomware variants.

3.1 Internet Traffic at Point I

Any generated traffic bound for the internet or for the network gateway is observed at point I. The Wireshark display filter used to create a 'point I view' varied slightly depending on the variant but was generally similar to the following:

```
ip.src_host==192.168.2.14 && !ip.dst_host==192.168.2.101 && !ip.addr==224.0.0.252 &&
                !ip.addr==239.255.255.250 && !ip.addr==192.168.2.255
```

This traffic included UDP, HTTP, and DNS protocols which were presumably used for command and control. The HTTP protocol traffic volume was relatively low, non-periodic, and could easily be lost within normal, user-generated network traffic on a production network. As such, it provides no opportunities for unique pattern-based detection. Signature-based mechanisms running on the SENTRY laptop could and did detect some of the ransomware-generated HTTP traffic as expected.

DNS and UDP traffic proved to be more interesting from a pattern-detection viewpoint. Sage traffic included 102 unanswered DNS requests within a three minute capture window. This is a very high number compared to the 1 unanswered DNS request in a 71 minute window when normal user activity was captured. In addition, the DNS server returned a 'no such name' or a 'failure' response 15 times.

The Sage traffic also produced six UDP traffic spikes over a four minute time window on an uncommon port 13655/udp. Each spike was approximately 1MB for a total of 6.4MB. This volume of unexpected UDP traffic might be detected but just as easily could be lost in the UDP traffic spikes that commonly occur from video streams, VoIP systems, and collaboration applications. A unique feature of this UDP traffic is that the destinations were spread out across a large group of 7698 IPs. Traffic sent to some of these IPs, 1269 to be exact, caused incoming ICMP packets mostly of type 3 (destination unreachable). The occurrences of specific ICMP type / code packets is broken-down in Table 2.

Count	ICMP Type.Code	Description
1196	3.3	Dest unreachable, port unreachable
53	3.10	Dest unreachable, comm w/ host administratively prohibited
11	3.1	Dest unreachable, host unreachable

5	11.0	Time exceeded, time-to-live exceeded in transit
2	3.0	Dest unreachable, net unreachable
2	3.13	Dest unreachable, communication administratively prohibited

Table 2 - ICMP traffic activity

This spike in ICMP traffic, is highly unusual and even if not a sign of malware, may be of interest to a network administrator as an indicator of a misconfiguration.

3.2 Server Traffic at Point S

The ransomware used SMB2 protocol (port 445/tcp) to read and write files on the shared drives and sometimes used NBSS, SMB, and LANMAN protocols (port 139/tcp) for authentication and session set-up. The Wireshark display filter used to create a 'point S view' was:

```
ip.src_host==192.168.2.14 && ip.dst_host==192.168.2.101
```

Server traffic is not commonly monitored in practice but doing so may be within reach of many companies that use Virtual Machine (VM) technology and want better protection against malware. If the file server is a VM, then monitoring traffic to it, and possibly other virtual servers, may be done with a packet sniffer VM instance being fed by a virtual mirror port from a virtual switch.

The file sharing protocols observed were all standard, common protocols for Windows file sharing and so their presence alone provides no useful information for ransomware detection. Sustained traffic volume for these protocols could be an indicator since the observed ransomware sequentially accessed most files on the network share which is not typical user behavior. To make use of a traffic-volume based indicator, a total bytes per unit time or total files per unit time threshold would need to be established and tuned.

Some thought was given to the possibility of measuring payload entropy of traffic between the endpoint to the file server. In information theory, entropy "is a measure of randomness or uncertainty in a signal." (NIST, n.d.) The cleartext bit stream of a human-readable document, for example, will have detectable patterns and features while the corresponding ciphertext will be

more random. In other words, the cleartext will have lower entropy as compared to the ciphertext. Correspondingly, the files encrypted by ransomware will generally have higher entropy as compared to the unencrypted files.

A couple of weak spots arise with this method. First, compressed files are also highly random and some common file types, like docx, are compressed. Second, some data is stored encrypted. Encryption, like compression, removes patterns in the data and increase entropy of the file. It is likely that trying to detect an increase in entropy for files that are normally compressed or encrypted would create many false positive results.

A post on the SANS Internet Storm Center Diary discusses the entropy of files created by ransomware and suggests that the "magic bytes" in many files may offer another indicator. (VandenBrink, 2016) Magic bytes allow applications to more easily detect a file type by providing a well-defined byte signature at the beginning of a file. Encrypting a file will turn the magic bytes, and the rest of the file content, into ciphertext. So the combination of high entropy and the absence of recognizable magic bytes is a strong indicator of file encryption. Using both entropy and magic bytes, the detection algorithm would involve observing file reads with magic bytes and lower entropy followed by file writes with no magic bytes and higher entropy.

Watching SMB2 command sequences may be another possible means of detection. To do its work, the ransomware needs to perform a very repetitive command sequence involving reads, writes, deletes and possibly renames. While all of these commands occur with normal user activity, quick repetition of certain command sequences would not normally be generated by a user.

The two SMB2 command sequences observed from the ransomware samples were read-delete-write and read-write-delete-rename although other command sequences may be possible. The actual timing of individual SMB2 commands within the sequences may vary depending on file size. For a large file, a longer delay will occur between the individual commands but this is not necessarily a problem as long as the command sequence, and not the timing, is the focus of observation. Fortunately, all of the observed SMB2 traffic for each infection occurred within a single TCP connection. This simplifies the task of observing

sequences since reconstructing SMB2 command sequences from multiple TCP streams is not necessary.

A Machine Learning (ML) or Artificial Intelligence (AI) algorithm may be a good implementation option for command sequence detection. Two rules of thumb for when ML may provide an effective solution are when the task is something "a typical person can do in less than one second" and when the task "involves predicting outcome of the next in a sequence of events." (Ng, 2016, 1:04:36) SMB2 command patterns generated by a typical user are likely to be much different from the high volumes of repetitive "read-delete-write" or "read-write-delete-rename" patterns generated by ransomware. SMB2 command patterns are generally not scrutinized by humans. If such commands were summarized into a human-readable format, it seems likely that a human could easily recognize the difference between end-user-generated and ransomware-generated SMB2 command patterns. So the rule of thumb suggests that a ML solution may work well.

A disadvantage of using server traffic for detection is that some ransomware variants do not encrypt network shares and therefore will not generate traffic to the server. It is good that the damage is limited to the workstation in these cases but this does not help with detection using network traffic.

3.3 Broadcast Traffic at Point B

For file sharing to occur, the workstation needs information about which servers are hosting network shares and about the individual network shares being hosted. This information is collected by an automatic discovery mechanism in which endpoints announce and query each other's capabilities. UDP broadcast traffic of NBNS, NBDS, and other protocols are part of this mechanism. The Wireshark display filters used to create a 'point B view' were:

```
ip.src_host==192.168.2.14 && !tcp.port==445 && !tcp.port==443 && !tcp.port==80 &&
                          !tcp.port==139 && !udp.port==53

                                !eth.type == 0x0800
```

Observations from the captured traffic shows that while some file sharing related broadcast traffic is generated as the ransomware executes, it does not represent a noteworthy difference

from normal activity. Broadcast traffic of similar protocols and volume was generated by both normal user activity and by Windows automatic discovery mechanisms. Also, no ARP broadcasts were observed during the ransomware traffic captures but some were observed in traffic from file share usage on an uninfected WRKSTN-1.

Network traffic capture by a Sentry device was in progress on a production network earlier this year when a ransomware infection, possibly a Nemucod variant, occurred. This provides an additional datapoint for broadcast traffic. Although the associated pcapng file is not included, a visual summary of the broadcast traffic from the infected workstation (192.168.1.149) is shown in Figure 4. The x-axis represents time from 9am to approximately 11:15am. Each of the four stacked graphs represent traffic volume in bits-per-second for a specific protocol as labeled. Broadcast traffic spikes occurred around 9am as the user started their day in a routine way. Shortly after 11am, a ransomware infection occurred and was identified by user reporting, encrypted files appearing on a network share, and IDS events.

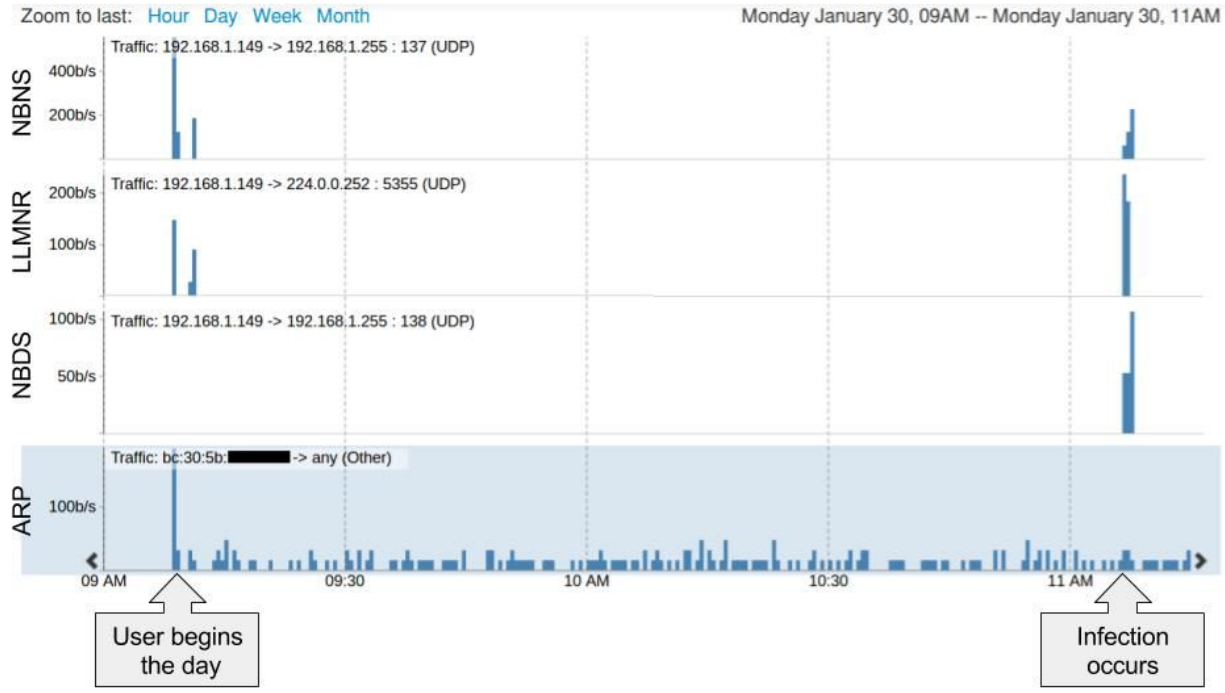

Figure 4 - Broadcast traffic from user activity and ransomware infection

Figure 4 shows that a spike in broadcast traffic did occur in NBNS & LLMNR protocols (top two graphs) at the time of ransomware infection shortly after 11am. For these two protocols, the traffic was consistent with and similar to broadcast traffic from the same workstation earlier in the morning (around 9am) when the user started their day. Lab results for these two protocols showed a similar traffic consistency between user activity on an uninfected workstation and ransomware related activity.

ARP traffic (bottom graph) shows only a beginning-of-the-day spike. It is expected that ARP traffic would spike as the user starts the day and the workstation becomes active again on the network. This graph is also consistent with lab results showing no noticeable change in ARP traffic at the time of infection.

A spike in NBDS protocol (third graph) is seen only when the infection occurs. Lab results show small amounts of NBDS traffic providing BROWSER protocol transport for most scenarios and unfortunately do not correlate with this data point from the field. This discrepancy is believed to be caused by the field user's SMB session being idle and timing out prior to infection. The build procedure step which simulates user activity immediately before ransomware execution prevents this timeout from occurring in the lab.

For the NetBIOS-based protocols, there are many variables to consider other timeouts such as the Windows operating system version in use and versions of their underlying NetBIOS implementations. Although broadcast traffic has the advantage of being visible over the entire network, the lack of clear traffic patterns and conflicting observations between the lab and field results suggests that it is not a fruitful place for pattern detection. Perhaps future, more aggressive malware variants that actively scan the network will produce more broadcast traffic while searching the network for files to encrypt.

4. Conclusion

Ransomware has been and will continue to be a serious threat to organizations of all sizes. Improved detection would help administrators more quickly contain and eradicate ransomware on an infected endpoint. This paper explored the possibility of ransomware detection by looking for post-install network traffic patterns. This approach unfortunately does not help detect

ransomware variants which only attack the local file system. However, unusual traffic patterns such as spikes in unanswered or failed DNS request, ICMP packets, UDP traffic to many endpoints, and certain SMB2 command sequences could be used to provide an additional layer of detection for certain ransomware variants. Experimental results did not reveal any ransomware-generated broadcast traffic that was notably different from user-generated broadcast traffic. Additional traffic patterns may be uncovered by studying other and possibly more aggressive ransomware variants. If automated, network-based ransomware detection can be made sufficiently accurate, then network-based containment techniques such as ARP spoofing may provide an option to neutralize the infected workstation.

5.References

AOL Inc. (2017). Moloch. Available from https://github.com/aol/moloch

Bradley, S. (2016). Ransomware - Protection and Prevention. Retrieved from http://www.sans.org/reading-room/whitepapers/awareness/ransomware-37317

Combs, G., et al. (2016). Tshark. Retrieved from https://www.wireshark.org

Combs, G., et al. (2016). Wireshark. Retrieved from https://www.wireshark.org

Duncan, B. (2017, March & April). Multiple blog posts referenced for malware sample download.
Retrieved from http://www.malware-traffic-analysis.net/2017/index.html

Keragala, D. (2016). Detecting Malware and Sandbox Evasion Techniques. Retrieved from https://www.sans.org/reading-room/whitepapers/forensics/detecting-malware-sandbox-evasion-techniques-36667

Mehmood, S. (2016). Enterprise Survival Guide for Ransomware Attacks. Retrieved from http://www.sans.org/reading-room/whitepapers/incident/enterprise-survival-guide-ransomware-attacks-36962

Ng, A. (2016). Nuts and Bolts of Applying Deep Learning. Retrieved from https://www.youtube.com/watch?v=F1ka6a13S9I

NIST (n. d.). True Randomness Can't be Left to Chance : Why entropy is important for information security (Pub Id 915121). National Institute of Standards and Technology. Retrieved from http://ws680.nist.gov/publication/get_pdf.cfm?pub_id=915121

Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear. (1996). Address Allocation for Private Internets", RFC 1918. Retrieved from http://www.rfc-editor.org/info/rfc1918

Saiyed, C. (2016). CryptoLocker. ISSA Journal, 14 (4), p.17

SANS. (2017). The Seven Most Dangerous New Attack Techniques, and What's Coming Next -- Continued. Retrieved from https://cc.readytalk.com/cc/s/meetingArchive?eventId=6oowksc223hm

Tcpdump. (2017). Tcpdump / Libpcap. Retrieved from http://www.tcpdump.org

VandenBrink, R. (2016). Using File Entropy to Identify "Ransomwared" Files. Retrieved from https://isc.sans.edu/forums/diary/Using+File+Entropy+to+Identify+Ransomwared+Files/21351

Appendix A - Build Procedure

Install laptop in DMZ; Wifi hardware switch to off; Win10 Enterprise trial in CD drive
Boot; Delete existing partitions; Create a new partition
When prompted, set all ‘Custom Settings’ to ‘Off’
Select ‘Join the local Active Directory Domain’
Add account named ‘victim’
Select ‘Not now’ for Cortana prompt
In Task Scheduler, unschedule Windows Updates & Windows Defender
Change hostname to SERVER-1 or WRKSTN-1: `WIN-R sysdm.cpl`
Reboot
Disable Windows Defender
Change sleep settings to never sleep when on AC power
Set time zone & ensure clock is accurate
Disable IPv6 under `Ethernet adapter’ settings
On SERVER-1
    Create ‘mapped’ share and populate with files
    Create ‘unmapped’ share and populate with files
On WRKSTN-1
    Turn on network discover in the Network and Sharing Center
    Map the ‘mapped’ drive on SERVER-1 as the Z: drive
Copy files from ‘mapped to the Documents folder
Remove CD media
Download ransomware sample
View contents on mapped Z: drive to simulate user activity
Start sniffer: `tshark -i bond0 -B 10 -b filesize:65536 -b files:50 -w /root/pcap/varient_name`
Run ransomware application

Note that for all traffic captures:
WRKSTN-1 is 192.168.2.14
SERVER-1 is 192.168.2.101

Appendix B - Common Protocol Acronyms

Acronyms displayed in the tshark output provided in subsequent appendices.

ARP - Address Resolution Protocol

BROWSER - Microsoft Windows Browser service, uses SMB over NBDS

LANMAN - Microsoft Windows Lanman Remote API Protocol, uses SMB over NBSS

LLMNR - Link-Local Multicast Name Resolution, queries on 224.0.0.252 port 5355/udp

ICMP - Internet Control Message Protocol

NBDS - NetBIOS Datagram Service, broadcasts queries over port 138/udp.

NBNS - NetBIOS Name Service, broadcasts queries over port 137/udp

NBSS - NetBIOS Session Service, uses port 139/tcp

SMB - Server Message Block protocol, uses port 139/tcp

SMB2 - Server Message Block protocol v2, uses port 445/tcp

SSDP - Simple Service Discovery Protocol, broadcast on 239.255.255.250 port 1900/udp

Note: tshark commands useful for understanding how protocols are decoded:

```
tshark -G decodes
tshark -G protocols
```

Appendix C - Expected Traffic Chatter

The following traffic was routinely generated by an uninfected WRKSTN-1 and was not considered to be ransomware related. Even if it did occur around the same time as the ransomware infection, the traffic volume and frequency was similar to normal user activity.

C.1 Internet

```
192.168.2.14 -> 192.168.2.254 HTTP 269 SUBSCRIBE /WANCommonInterfaceConfig HTTP/1.1
192.168.2.14 -> 192.168.2.254 HTTP 260 SUBSCRIBE /WANIPConnection HTTP/1.1
192.168.2.14 -> 192.168.2.254 HTTP 229 GET /RootDevice.xml HTTP/1.1
192.168.2.14 -> Many Microsoft-related IP address for updates & telemetry
      All IPs with ASN 8068 or 8075
      IPs resolving to: microsoft.com, windowsupdate.com, ocsp.msocsp.com, ...
```

Also observed was internet traffic to 209.53.113.5 (a.fc.namequery.com) sent by an anti-theft solution installed on the WRKSTN-1 laptop.

C.2 Server

Server traffic from 192.168.2.14 -> 192.168.2.101:

```
NBSS Session request, to SERVER-1<20> from WRKSTN-1<00>
SMB Negotiate Protocol Request
SMB Session Setup AndX Request, NTLMSSP_NEGOTIATE
SMB Session Setup AndX Request, NTLMSSP_AUTH, User: \
SMB Tree Connect AndX Request, Path: \\SERVER-1\IPC$
LANMAN NetServerEnum2 Request, Domain Enum
LANMAN NetServerEnum2 Request, Workstation, Server, SQL Server, Domain Controller,
Backup
   Controller, Time Source, Apple Server, Novell Server, Domain Member Server, Print
   Queue Server, Dialin Server, Xenix Server, NT Workstation, Windows for Workgroups,
   Unknown server type:14, NT Server, Potential Browser, Backup Browser, Master
Browser,
   Domain Master Browser, OSF, VMS, Windows 95 or above, DFS server, Unknown server
   type:24, Unknown server type:25, Unknown server type:26, Unknown server type:27,
   Unknown server type:28, Unknown server type:29, Local List Only, Domain Enum
SMB Tree Disconnect Request
SMB Logoff AndX Request
SMB2 178 Ioctl Request FSCTL_QUERY_NETWORK_INTERFACE_INFO

SMB2 190 Create Request File: srvsvc
SMB2 210 Create Response File: srvsvc
SMB2 162 GetInfo Request FILE_INFO/SMB2_FILE_STANDARD_INFO File: srvsvc
SMB2 171 Read Request Len:1024 Off:0 File: srvsvc
SMB2 146 Close Request File: srvsvc
```

C.3 Broadcast

```
192.168.2.14 -> 192.168.2.255 BROWSER 221 Request Announcement WRKSTN-1
192.168.2.14 -> 192.168.2.255 BROWSER 251 Host Announcement WRKSTN-1, Workstation,
                              Server, NT Workstation, Potential Browser
192.168.2.14 -> 192.168.2.255 NBNS 92 Name query NB SERVER-1<20>
192.168.2.14 -> 224.0.0.252 LLMNR 68 Standard query 0x009b A SERVER-1
192.168.2.14 -> 224.0.0.252 LLMNR 68 Standard query 0x897c AAAA SERVER-1
192.168.2.14 -> 239.255.255.250 UDP 666 62490 → 3702  Len=624
192.168.2.14 -> 239.255.255.250 IGMPv2    60    Membership Report group
239.255.255.250
2001:0:9d38:953c:38d3:2afd:52a6:a6bc -> ff02::1 IPv6 82 IPv6 no next head
fe80::ffff:ffff:fffe -> ff02::2 ICMPv6 103 Router Solicitation
192.168.2.14 -> 239.255.255.250 239.255.255.250 SSDP 179 M-SEARCH * HTTP/1.1
```

Appendix D - Revenge Traffic Summary

D.1 Internet

```
 39   8.227930 91.207.7.77  HTTP POST /images/temp/4gallery/temp_reserv/gallery.php
                                     HTTP/1.1  (application/x-www-form-urlencoded)
8379  14.656698 91.207.7.77  HTTP POST /images/temp/4gallery/temp_reserv/gallery.php
                                      HTTP/1.1  (application/x-www-form-urlencoded)
```

D.2 Server

Server traffic was from 192.168.2.14 -> 192.168.2.101:

```
50 10.980711 SMB2 Create Request File:
52 10.982311 SMB2 Find Request File: SMB2_FIND_ID_BOTH_DIRECTORY_INFO Pattern: *;Find
                  Request File: SMB2_FIND_ID_BOTH_DIRECTORY_INFO Pattern: *

58 11.111720 SMB2 Create Request File: 123.txt;SetInfo Request
                  FILE_INFO/SMB2_FILE_BASIC_INFO
60 11.113070 SMB2 Close Request File: 123.txt
62 11.114301 SMB2 Create Request File: 123.txt
64 11.115816 SMB2 Read Request Len:3 Off:0 File: 123.txt
66 11.117048 SMB2 Close Request File: 123.txt
68 11.118183 SMB2 Create Request File: 123.txt
70 11.119564 SMB2 SetInfo Request FILE_INFO/SMB2_FILE_DISPOSITION_INFO File: 123.txt
72 11.120438 SMB2 Close Request File: 123.txt
74 11.122602 SMB2 Create Request File: 24D140319C7D375BFB97EFBF35DDF9E7.REVENGE
76 11.124603 SMB2 Write Request Len:16 Off:0 File:
                  24D140319C7D375BFB97EFBF35DDF9E7.REVENGE
More write requests here……
88 11.130680 SMB2 Write Request Len:5 Off:49 File:
                  24D140319C7D375BFB97EFBF35DDF9E7.REVENGE
90 11.131564 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_NETWORK_OPEN_INFO File:
                  24D140319C7D375BFB97EFBF35DDF9E7.REVENGE
92 11.132426 SMB2 Close Request File: 24D140319C7D375BFB97EFBF35DDF9E7.REVENGE
94 11.133939 SMB2 Create Request File: 123.txt
Previous traffic block repeated for all target files

8184 14.378052 SMB2 Create Request File: # !!!HELP_FILE!!! #.TXT
8186 14.379571 SMB2 Create Request File: # !!!HELP_FILE!!! #.TXT
8193 14.382030 SMB2 Write Request Len:7012 Off:0 File: # !!!HELP_FILE!!! #.TXT
8196 14.383050 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_NETWORK_OPEN_INFO File:
                    # !!!HELP_FILE!!! #.TXT
8198 14.383283 SMB2 Close Request File: # !!!HELP_FILE!!! #.TXT
8200 14.384807 SMB2 Create Request File: # !!!HELP_FILE!!! #.TXT
8202 14.386319 SMB2 Write Request Len:102 Off:7012 File: # !!!HELP_FILE!!! #.TXT
8204 14.387176 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_NETWORK_OPEN_INFO File:
                    # !!!HELP_FILE!!! #.TXT
8206 14.388039 SMB2 Close Request File: # !!!HELP_FILE!!! #.TXT
```

```
8208 14.389677 SMB2 Create Request File: ;Find Request
SMB2_FIND_ID_BOTH_DIRECTORY_INFO
                    Pattern: *;Find Request SMB2_FIND_ID_BOTH_DIRECTORY_INFO Pattern:
*
8215 14.395444 SMB2 Close Request File:
Previous traffic block repeated for ‘Kaspersky Virus Removal Tool.exe’ and
                                    ‘Kaspersky Virus Removal
Tool.exe:Zone.Identifier’
```

D.3 Broadcast

None outside of expected traffic as shown in Appendix C.

Appendix E - Sage 2.0 Traffic Summary

E.1 Internet

Port 13655/udp packet to 7698 unique target IPs repeated 38,389 times.

```
192.168.2.14 -> 138.197.53.223 UDP 195 61881 → 13655  Len=153
```

Of the 7698 target IPs, some were unreachable:
2 - ICMP Destination unreachable (Network unreachable)
2 - ICMP Destination unreachable (Communication administratively filtered)
5 - ICMP Time-to-live exceeded (Time to live exceeded in transit)
11 - ICMP Destination unreachable (Host unreachable)
53 - ICMP Destination unreachable (Host administratively prohibited)
1196 - ICMP Destination unreachable (Port unreachable)

98 occurrances: DNS Standard query A mbfce24rgn65bx3g.rzunt3u2.com
4 occurances: DNS Standard query A mbfce24rgn65bx3g.er29sl.com

E.2 Server

Server traffic was from 192.168.2.14 -> 192.168.2.101:

```
32 15.933407 SMB2 Create Request File:
34 15.934583 SMB2 Close Request File:
64 35.007448 SMB2 Close Request

17024 85.165231 SMB2 Session Setup Request, NTLMSSP_NEGOTIATE
17026 85.167223 SMB2 Session Setup Request, NTLMSSP_AUTH, User: WRKSTN-1\victim
17028 85.171094 SMB2 Tree Connect Request Tree: \\SERVER-1\IPC$
17030 85.172226 SMB2 Create Request File: srvsvc
17032 85.173323 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_STANDARD_INFO File: srvsvc
17034 85.174341 DCERPC Bind: call_id: 2, Fragment: Single, 2 context items: SRVSVC
V3.0 (32bit NDR), SRVSVC V3.0 (bind time feature negotiation)
17036 85.175327 SMB2 Read Request Len:1024 Off:0 File: srvsvc
17038 85.176328 SRVSVC NetShareEnumAll request
17040 85.177828 SMB2 Close Request File: srvsvc
17042 85.183473 SMB2 Tree Connect Request Tree: \\SERVER-1\mapped
17044 85.187721 SMB2 Ioctl Request FSCTL_DFS_GET_REFERRALS, File: \SERVER-1\unmapped
17046 85.189457 SMB2 Tree Connect Request Tree: \\SERVER-1\unmapped
17050 85.259732 SMB2 Create Request File:
17052 85.262205 SMB2 Find Request File: SMB2_FIND_ID_BOTH_DIRECTORY_INFO Pattern:
*;Find Request File: SMB2_FIND_ID_BOTH_DIRECTORY_INFO Pattern: *
17080 95.911374 SMB2 Tree Disconnect Request

25654 122.689867 SMB2 Create Request File: !Recovery_FAM.html
25736 122.697854 SMB2 Write Request Len:9378 Off:0 File: !Recovery_FAM.html
25813 122.704128 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_NETWORK_OPEN_INFO File:
!Recovery_FAM.html
25825 122.705367 SMB2 Close Request File: !Recovery_FAM.html
```

```
25834 122.706591 SMB2 Close Request File:

26932 122.831833 SMB2 Create Request File: 123.txt
26957 122.837204 SMB2 Read Request Len:3 Off:0 File: 123.txt
26965 122.839220 SMB2 Write Request Len:3 Off:0 File: 123.txt...
26972 122.839733 SMB2 Write Request Len:96 Off:3 File: 123.txt...
26982 122.840862 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_NETWORK_OPEN_INFO File:
                     123.txt...
26991 122.846094 SMB2 Close Request File: 123.txt...
26999 122.846712 SMB2 Close Request File: 123.txt
27009 122.847581 SMB2 Create Request File: 123.txt
27019 122.848473 SMB2 SetInfo Request FILE_INFO/SMB2_FILE_DISPOSITION_INFO File:
123.txt
27031 122.849581 SMB2 Close Request File: 123.txt
27043 122.850602 SMB2 Create Request File: 123.txt...
27054 122.852334 SMB2 SetInfo Request FILE_INFO/SMB2_FILE_RENAME_INFO File:
123.txt...
                     NewName:123.txt.sage
27064 122.853230 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_NETWORK_OPEN_INFO File:
                     123.txt...
27105 122.860845 SMB2 Close Request File: 123.txt…
Previous traffic block repeated for all target files

31275 123.181214 SMB2 SetInfo Request FILE_INFO/SMB2_FILE_DISPOSITION_INFO File:
35082 123.562097 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_NETWORK_OPEN_INFO
35092 123.562955 SMB2 Close Request
45435 149.405883 SMB2 Create Request File:
45437 149.407970 SMB2 Close Request File:
45439 149.439487 SMB2 Create Request File: !Recovery_FAM.html
45448 149.441991 SMB2 Write Request Len:9151 Off:0 File: !Recovery_FAM.html
45452 149.442837 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_NETWORK_OPEN_INFO File:
                     !Recovery_FAM.html
45454 149.443608 SMB2 Close Request File: !Recovery_FAM.html
```

E.3 Broadcast

None outside of expected traffic as shown in Appendix C.

Appendix F - CryotpShield Traffic Summary

F.1 Internet

```
10 1.068699 185.125.32.2 HTTP 872 POST /images/gallery/g3.php HTTP/1.1
```

F.2 Server

Server traffic was from 192.168.2.14 -> 192.168.2.101:

```
17 1.772100 SMB2 Find Request File: SMB2_FIND_ID_BOTH_DIRECTORY_INFO Pattern: *;
                Find Request File: SMB2_FIND_ID_BOTH_DIRECTORY_INFO Pattern: *

22 1.777980 SMB2 Create Request File: 123.txt;SetInfo Request
                FILE_INFO/SMB2_FILE_BASIC_INFO
24 1.779221 SMB2 Close Request File: 123.txt
26 1.780574 SMB2 Create Request File: 123.txt
28 1.782238 SMB2 Read Request Len:3 Off:0 File: 123.txt
30 1.783239 SMB2 Close Request File: 123.txt
32 1.784480 SMB2 Create Request File: 123.txt
34 1.785856 SMB2 SetInfo Request FILE_INFO/SMB2_FILE_DISPOSITION_INFO File: 123.txt
36 1.786844 SMB2 Close Request File: 123.txt
38 1.789488 SMB2 Create Request File:
                123.GKG.[R_SP@INDIA.COM].ID[9C7D375B607C01B2].CRYPTOSHIELD.
40 1.791456 SMB2 Write Request Len:16 Off:0 File:
                123.GKG.[R_SP@INDIA.COM].ID[9C7D375B607C01B2].CRYPTOSHIELD.
42 1.792579 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_NETWORK_OPEN_INFO File:
                123.GKG.[R_SP@INDIA.COM].ID[9C7D375B607C01B2].CRYPTOSHIELD.
44 1.793470 SMB2 Close Request File:
                123.GKG.[R_SP@INDIA.COM].ID[9C7D375B607C01B2].CRYPTOSHIELD.
46 1.794977 SMB2 Create Request File: 123.txt
```

Previous traffic block repeated for all target files

```
7927 3.038330 SMB2 Create Request File: # RESTORING FILES #.HTML
7929 3.039975 SMB2 Create Request File: # RESTORING FILES #.HTML
7933 3.042252 SMB2 Write Request Len:2894 Off:0 File: # RESTORING FILES #.HTML
7936 3.043205 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_NETWORK_OPEN_INFO File:
                  # RESTORING FILES #.HTML
7938 3.043703 SMB2 Close Request File: # RESTORING FILES #.HTML
7940 3.045217 SMB2 Create Request File: # RESTORING FILES #.HTML
7942 3.046721 SMB2 Write Request Len:126 Off:2894 File: # RESTORING FILES #.HTML
7944 3.047618 SMB2 GetInfo Request FILE_INFO/SMB2_FILE_NETWORK_OPEN_INFO File:
                  # RESTORING FILES #.HTML
7946 3.048468 SMB2 Close Request File: # RESTORING FILES #.HTML
```

Previous block repeated for 'Recovery Tools.exe' and 'Recovery Tools.exe:Zone.Identifier'

```
7968 3.062957 SMB2 Create Request File: ;Find Request
SMB2_FIND_ID_BOTH_DIRECTORY_INFO
```

```
                    Pattern: *;Find Request SMB2_FIND_ID_BOTH_DIRECTORY_INFO Pattern:
*
8123 3.149612 SMB2 Create Request File: ;Find Request
SMB2_FIND_ID_BOTH_DIRECTORY_INFO
                    Pattern: *;Find Request SMB2_FIND_ID_BOTH_DIRECTORY_INFO Pattern:
*
```

F.3 Broadcast

None outside of expected traffic as shown in Appendix C.