Harness the power of SIEM

GIAC (GCIA) Gold Certification

Author: Dereck Haye, olbaidle@gmail.com

Advisor: Rodney Caudle

Accepted: April 15th 2009

Having problems with Conficker worm? The security community has witnessed firsthand a worm which, at the height of its reach, could have assimilated around 9+ million hosts across the internet (F-Secure, 2009a). This is a staggering amount which, when published, indicated the seriousness of the situation and stimulated security teams to verify activity in their environment. This whitepaper is designed as a guide through the process of defining metrics needed to accurately detect the Conficker worm. In addition, this paper will demonstrate how to use those metrics in Siem architecture for detection purposes. Security researchers and vendors have released guidelines and articles covering the Conficker worm and how it is infecting machines in various ways. Here the goal is to show how it is possible to take those individual security updates and, in Siem architecture combine them with other metrics to enhance and tune detection capabilities. This provides a powerful tool to defend an environment against the Conficker and other fast-spreading worms.

1. Introduction

On October 23, 2008 Microsoft released security update MS08-067 (Microsoft, 2009). Until the time the advisory was released the vulnerability was facing limited yet targeted attack. However, after the release of MS08-067 Conficker came into prominence. Naturally malware developers understood the potential of this vulnerability, incorporated it along with other exploitation vectors into their malware (SRI International, 2009) which, when released, has become known as Conficker. What happened next started causing concern for the security community almost immediately. Chronologically each metric of information released publicly (British broadcasting corporation, 2009) reveals an exact date. Specifically on November 21 2008, information on a worm with the official recognition of Conficker.A (Microsoft, 2009) was published. Security team analysts immediately turned to detection of activity relating to Conficker and that of subsequent variants that occurred (British broadcasting corporation, 2009). More importantly there was a need to detect and clean the infections as quickly as possible. After initial analysis of the worm had been carried out it was evident that the worm had been well designed. This was particularly recognized in the ability of Conficker to breech networks, rapidly infect hosts and defend itself (Leder, Werner, 2009) without containing many previously unseen exploitation metrics (Nahorny, 2009). Conficker was adequately armed to spread quickly and effectively taking advantage of two facts. The number of un-patched, infectable machines and the number of unregistered (pirated) Windows systems (SRI International, 2009) reachable through the internet. Conficker, coupled with the number of infectable hosts the MS08-067 vulnerability allowed, incorporated additional vectors, such as the ability to spread through useable drives with the simple but effective autoplay (Nahorny, 2009) feature. This would allow Conficker to cross airgaps and other physical security precautions. This document will demonstrate, the steps undertaken to identify, isolate and detect a set of individual metrics that identify activity relating to Conficker worm components as seen from inside a SIEM tool such as Arcsight (Voorhees, 2007). Once identified, the indicators need to be combined logically, or correlated, to provide accurate detection. If Conficker has infected a host it is already too late for that host, what matters is what the host will do next. Quick detection and isolation is paramount to minimize the damage caused by infected hosts, thus preventing the potential costs of infections escalating (Cyber Secure Institute, 2009). So the focus is detecting the activity during initial host infection and the subsequent attempts of that infected host to infect other hosts around it. In focusing on these areas Conficker can be detected quickly, and it can be detected while trying to infect other machines (F-Secure, 2009b). The approach taken encompasses facets of pre and post infection, purely as these were among the first detectable metrics outlined when Conficker had been analyzed (SRI International, 2009). Importantly the impact of preventing infections is much more cost effective than having to clean all of the infections (Microsoft, 2005). In the event of a massive outbreak of host infections understanding the cause of the outbreak is a foundation to forming effective mitigation techniques (US-CERT, 2005). What different patterns of activity can be seen? How often are they occurring? Is it confined to a specific subnet? Are hosts getting locked out of their domain? Is there excessive NetBIOS activity? All of these are valid questions which, if positively answered, are possibly symptoms of a Conficker infection. The key here is the ability to identify these metrics and use them as part of accurate detection.

2. Methodology

In order to detect Conficker it is imperative that an understanding of what exactly, detection rules should be looking for is established. For this reason the methodology follows a relatively simple path. First, plenty of research needs to be conducted to understand how Conficker was infecting machines (SRI International, 2009) and the processes the worm would follow after infection. Moreover if Conficker had the ability to defend itself or re-infect after the initial infection was cleaned (Leder, Werner, 2009). Then understanding these individual components and applying detection rules to a secured environment could verify the presence of Conficker components. Once the malicious activity has been confirmed in the environment, logically the activity can be correlated to enhance and tighten overall detection of Conficker. Each time a component of Conficker is identified the following steps are taken.

Step 1, Extract metrics: Define an attack vector Conficker uses to spread.

Step 2, Identify Conficker: Harness Arcsight tool to find Conficker behavior.

Step 3, Record activity: Write rules that detect Conficker and record the results.

The steps above outline the process of identifying what needs to be done for each identified component of Conficker. However there are greater steps which must be taken in the process of detecting Conficker within Siem. These more general steps encompass the more holistic approach taken to detecting the worm in action. Finally the following steps lead into eventual correlation possibilities.

2.1. Step 1: Research Malware Components and Activity Signatures

The references section of this document contains articles which analyze the various facets of the Conficker and the variants. It is certain that most of the security companies out there will have done their own analysis and they are all brilliant sources of information. Each published article could provide a unique twist which can be used to provide a slightly different analysis. For those with little available cycles due to worm activity, the time required to comb through the different sources of information can be tedious. However, preserving and reading as many data sources as possible to glean definable components of this malware is required. A metric is linked directly to a unique component of the malware that is detectable through a unique signature of activity. An example of a unique signature from Conficker is the initiation of SMB sessions on port 445/TCP of the victim (SRI International, 2009). Logically applying this to a monitored environment would mean looking for traffic heading from an internal host to port 445 on another internal host. In deriving and defining metrics in this manner it is possible to start detecting specific parts of Conficker A, B and variant activity inside a SIEM tool. All links to documents used in this whitepaper are included in the references section at the end of this whitepaper. Articles and research documents are continually updated as the worm evolved through A, B and C variants (Leder, Werner, 2009). The evolution of detection rules and the research and analysis has also kept pace (Nahorney, 2009). Ultimately it is most important during research and analysis that continual information streams can be harnessed and this information be used in a SIEM environment for malware detection. This paper uses examples which have been derived from such sources and applied within the Arcsight SIEM tool (Voorhees, 2007). In harnessing Arcsight correlation capabilities (Arcsight, 2008) irrespective of security device vendor and device deployment, it is essential individual Conficker metrics are defined and fed into Arcsight (Voorhees, 2007). In consolidating the various data feeds, the various signatures of the worm components are detectable. Arcsight provides the power to make statements against event driven metrics to verify that derived Conficker signatures are present within an environment. These metrics can then be leveraged within active channels, filters and rules (Voorhees, 2007).

2.2. Step 2: Extracting metrics.

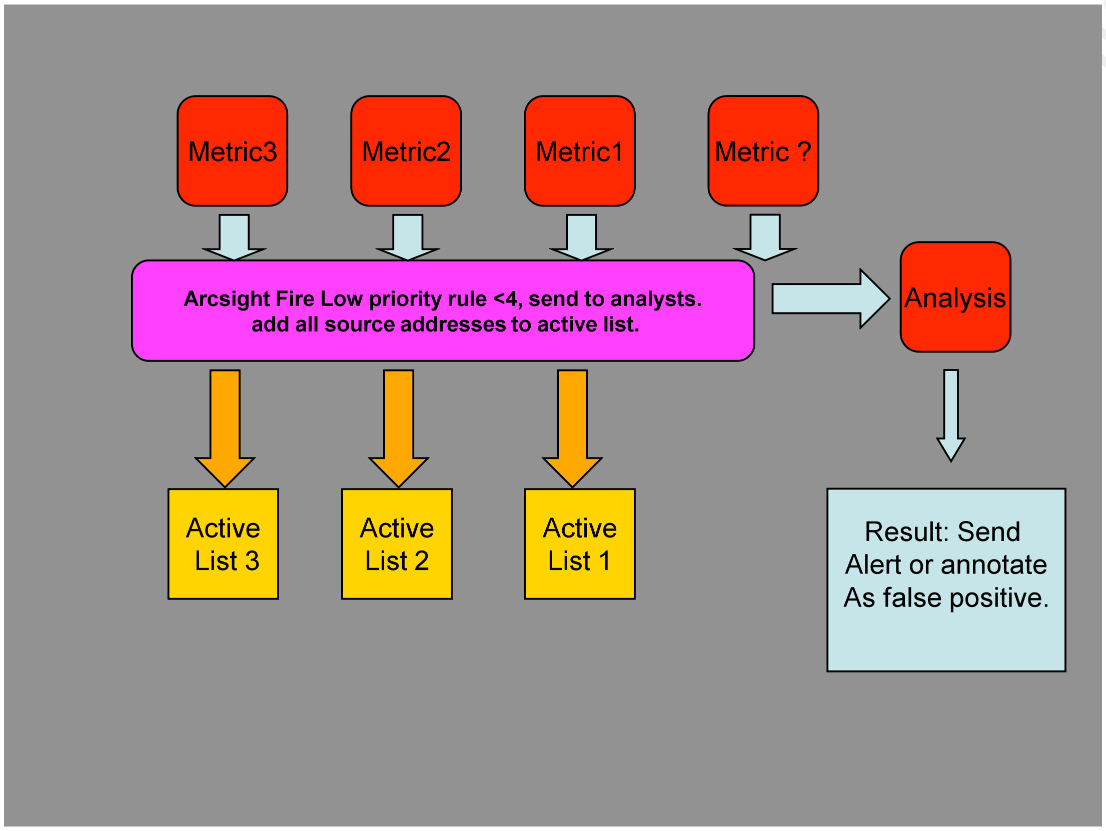

Metric3
Metric2
Metric1
Metric ?
Arcsight Fire Low priority rule <4, send to analysts.
add all source addresses to active list.
Analysis
Active List 3
Active List 2
Active List 1
Result: Send Alert or annotate As false positive.

Diagram 1 Logical process of metric fire and analysis phases.

Looking at diagram 1 the goal is to accomplish defining the individual metrics, as identified in the research completed in step 1. From these metrics only a low priority rule will be generated, importantly as the probability of a false positive is high so the priority of the rule must be low. In any network environment there exist a number of common components which are present and essential in getting the modern network to run correctly (Briddell, 2002). These common components interact inside the network architecture utilizing services and protocols such as DNS, DHCP, ARP and are passively detectable in any network (Ho, 2002). The Conficker worm is designed to specifically exploit common systems which exist on one or more of these common components (SRI International, 2009)! This means Conficker-like activity occurs normally on a daily basis in network environments, the separator is often in the extra noise the Conficker activity generates during the infection process (SRI International, 2009).Arcsight rules should have the ability to detect these traffic abnormalities and generate alerts accordingly. Then analysts should specifically evaluate the low priority incident generated and provide sufficient feedback through the subsequent analysis (Voorhees, 2007). This analysis should focus on looking at comparing normal levels of legitimate business traffic to the sudden bursts associated with a malware infestation. Analysts should follow techniques for incident best practices (NIST, 2004) if their evaluation provides sufficient grounds to conclude the host in question are indeed infected. As an example of a thorough investigation an analyst would utilize the Arcsight tool to investigate events around the initial alert for corroborative information. This investigation can use customized tools of preference along with creating active channels (Voorhees, 2007) to determine likelihood of host infection. If the conclusion of the analyst investigation and data mining evaluation (Jordan, 2007) is sufficient to warrant that the initial incident priority should be raised to a higher level, then this should occur given the analysts observation warrants a higher priority. Through incident investigation feedback, in the form of true positive or false positive, there is a better understanding of what is or is not working for detection and metrics can be adapted accordingly. For example, write a rule for each metric and allow the rule to run in Arcsight. As each rule is triggered, fire a low priority alert <4 and forward to a team of analysts for review. Additionally, each time the rule fires the potentially infected hosts should be added to an active watch list as the rule runs in Arcsight. Recording the results of this possible infection is a crucial step if at some point correlation via active lists is to take place, this would mean that a correlated rule could easily make use of these possible infected host lists combined together in one rule. Ultimately the IP address of a potentially infected machine is information which can be reused and is very important to track! Crucially as the process of writing this detection rule evolved along with Conficker through variants A, B and C (Nahorney, 2009), the method of defining Conficker metrics as the malware evolves scales well in rule correlation (Arcsight, 2008). This achieves the right balance so the final rule is going to detect and find current Conficker infections and alert analysts at different stages of the detection process.

Carefully reading the document from SRI (SRI International, 2009), there are some metrics which can be extracted. SRI states the extracts highlight an infected host carrying out static metrics (SRI International, 2009) such as going to certain URL or website in order to verify or fulfill a part of its function. That is detectable behavior in Arcsight if data feeds coming from event sources such as DNS, Firewall and proxies are being fed into Arcsight. Log files from these event sources are scraped and in turn are fed into the Arcsight tool (Voorhees, 2007), when a host inside a network tried to connect to the following sites www.getmyip .org or checkup .dyndns.org (SRI International, 2009) for example , a rule in Arcsight can detect this behavior. As a result the rule fires a low priority rule (Voorhees, 2007) and sends it through to an analyst for evaluation of the suspect hosts behavior (see diagram 1). The analyst would then check the rule results looking at source and destination IP addresses (Voorhees, 2007). Harnessing the ability of Arcsight for investigation is an advantage here as the correlation options lead directly to correlate events simply by tool functionality.

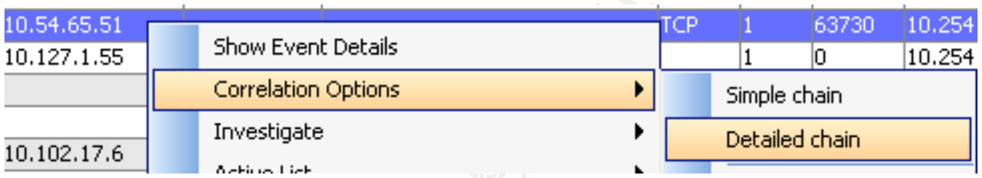


FIG 0 Correlation options in Arcsight tool.

Alternatively opening an active channel and evaluating other events which occurred around the same timeframe in an effort to establish correlation (Arcsight, 2008) can reveal events of interest. Indeed the Arcsight tool provides these options in interactive menus.

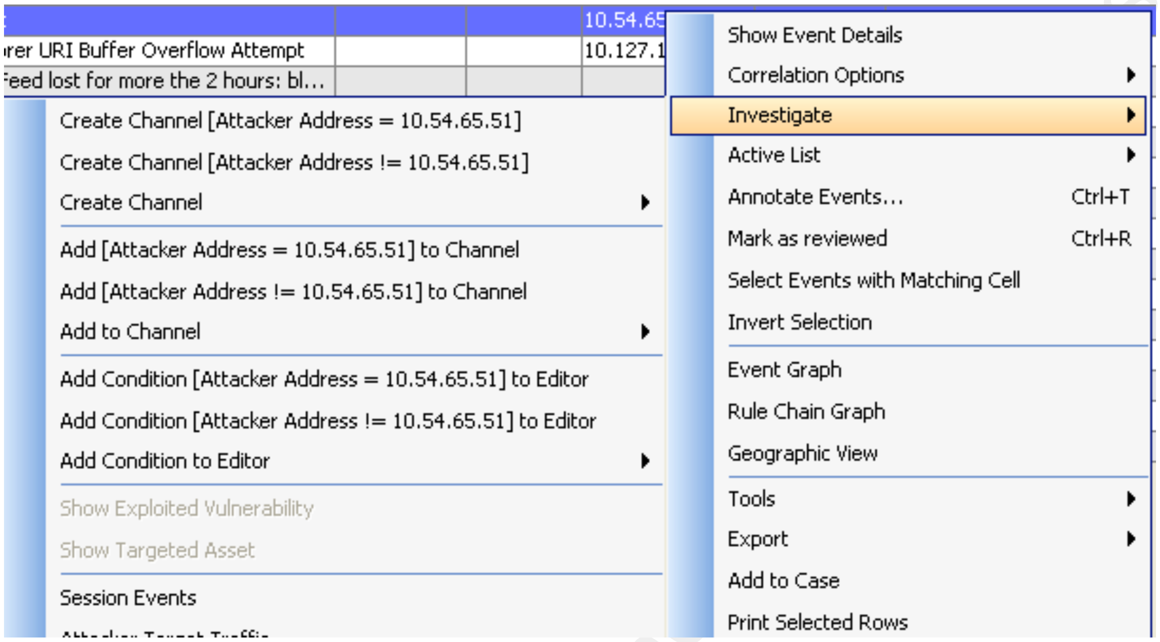

10.54.65
rer URI Buffer Overflow Attempt
10.127.1
eed lost for more the 2 hours: bl...
Create Channel [Attacker Address = 10.54.65.51]
Create Channel [Attacker Address != 10.54.65.51]
Create Channel
Add [Attacker Address = 10.54.65.51] to Channel
Add [Attacker Address != 10.54.65.51] to Channel
Add to Channel
Add Condition [Attacker Address = 10.54.65.51] to Editor
Add Condition [Attacker Address != 10.54.65.51] to Editor
Add Condition to Editor
Show Exploited Vulnerability
Show Targeted Asset
Session Events
Show Event Details
Correlation Options
Investigate
Active List
Annotate Events... Ctrl+T
Mark as reviewed Ctrl+R
Select Events with Matching Cell
Invert Selection
Event Graph
Rule Chain Graph
Geographic View
Tools
Export
Add to Case
Print Selected Rows

Fig0.1 Investigation options in Arcsight.

In addition, the SRI document reveals signatures describing Conficker propagation (SRI International, 2009). SRI states the Conficker propagation occurs over TCP port 445 which is associated with Microsoft CIFS File Sharing activity. The document also expands on multiple variants of the Conficker worm at this point. In expanding on the different variants it shows just how much the malware can be changed over a relatively short timeframe. As an example in analysis between variant B and variant C, the C variant “incorporates a major restructuring of B's previous thread architecture and program logic, including major functional additions such as a new peer-to-peer (P2P) coordination channel, and a revision of the domain generation algorithm (DGA)” (SRI International, 2009). Evidently it is reasonable to derive that the Conficker authors are tracking efforts to mitigate the worm and are making an effort to ensure it continues to propagate. It is important to realize that even though Conficker has evolved over time (British Broadcasting Corporation, 2009) some components will remain unchanged throughout progression. Additionally noted is some interesting high port/low port activity which could be of use as a correlation metric. The port 445 activity should be detectable in a firewall log or by identifying an IDS/IPS signature written to isolate this or similar activity. Once the TCP port 445 metric is identified set Arcsight to trigger a low priority rule (Voorhees, 2007) and send it to an analyst for evaluation (Diagram 1).

Finally SRI provides information outlining NetBIOS brute forcing and the USB infection mechanism (SRI International, 2009). The NetBIOS activity if breeching internal security policy should be detectable in log events providing the technology is placed to detect such events, policies are set to alert on host to host traffic and that these alerts are finally fed into Arcsight. The alert may be generated in the form of a firewall drop log event on a firewall interface where NetBIOS is not allowed to pass. Alternatively though analysis the IntruShield IDS/IPS signature “NETBIOS-SS: Microsoft Server Service Remote Code Execution Vulnerability” was identified as working to detect the NetBIOS host infection component of the malware (SRI International, 2009). Essentially in continuing to go through documents which identify components of Conficker an approach is needed which will exhaust as many detection avenues as possible. Looking at Conficker component NetBIOS could mean evaluating a combination of port hits, antivirus and IDS/IPS signatures in the Arcsight tool to find out what works best. In the case of autorun Windows event logging could pick up the USB vector and any alterations to running processes that occur thereafter. Detecting the autorun in Arcsight would require windows event logging be configured appropriately and fed into the Arcsight system. Focusing on the “NETBIOS-SS: Microsoft Server Service Remote Code Execution Vulnerability” signature and port activity at firewall a third metric has been identified. Again set Arcsight to fire another low priority rule (Voorhees, 2007) and send it through to an analyst for evaluation of the suspect hosts (Diagram 1).

Remember there is a lot of information on Conficker out there, up until this point only three metrics have been extrapolated from them. To recap on what has been enumerable of Conficker: - there are some internal hosts accessing external sites, some TCP port 445 activity, some NetBIOS port activity and an NetBIOS IDS/IPS signature. Certainly in the right infrastructures where many different log feeds are incorporated into the Arcsight system the more scope there is for cross platform correlation (Arcsight, 2008). Then a natural logical extrapolation means the metrics can be combined together for some correlation. The goal is to identify and isolate infected machines to prevent further spread of the infection. It is very important to understand the enemy (Leder, Werner, 2009). In reading up on how the enemy is behaving and defining the malicious vectors as metrics, it is possible to logically combine the results of each metric via correlation. Logic does not need to be complicated, in fact it pays to try and keep the approach generally as simple as possible (Security labs, 2007). Previously in this section there are three identified metrics. These were the SMB, NetBIOS (port + IDS/IPS) and the metrics for urls or IP addresses (SRI International, 2009) fed into the Arcsight tool (Voorhees, 2007).

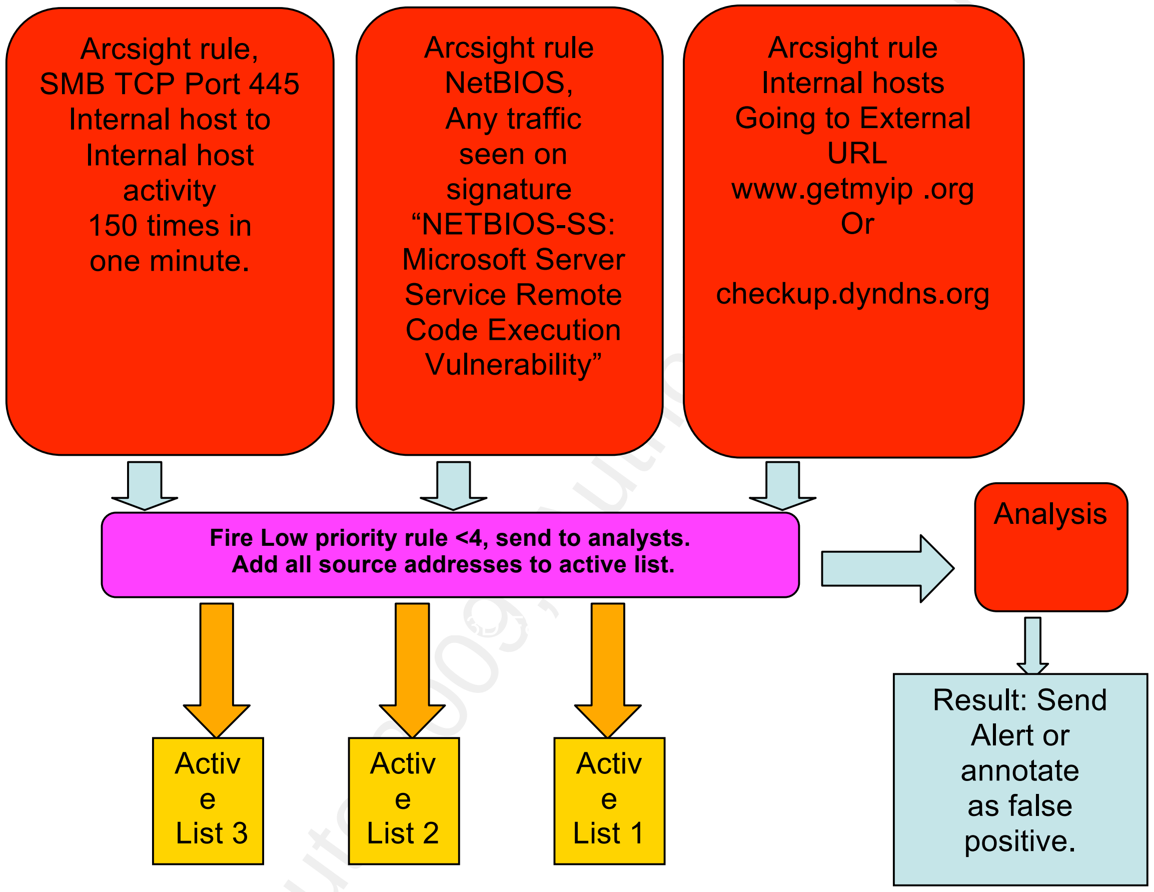

Arcsight rule, SMB TCP Port 445 Internal host to Internal host activity 150 times in one minute.
Arcsight rule NetBIOS, Any traffic seen on signature "NETBIOS-SS: Microsoft Server Service Remote Code Execution Vulnerability"
Arcsight rule Internal hosts Going to External URL www.getmyip .org Or checkup.dyndns.org
Fire Low priority rule <4, send to analysts. Add all source addresses to active list.
Analysis
Result: Send Alert or annotate as false positive.
Activ e List 3
Activ e List 2
Activ e List 1

Diagram 2 patterns of metrics with associated results.

The goal was to identify infected machines by firing low priority rules and have the resulting alert analyzed. Regardless of the outcome the host which caused the alert will be added to an active list.

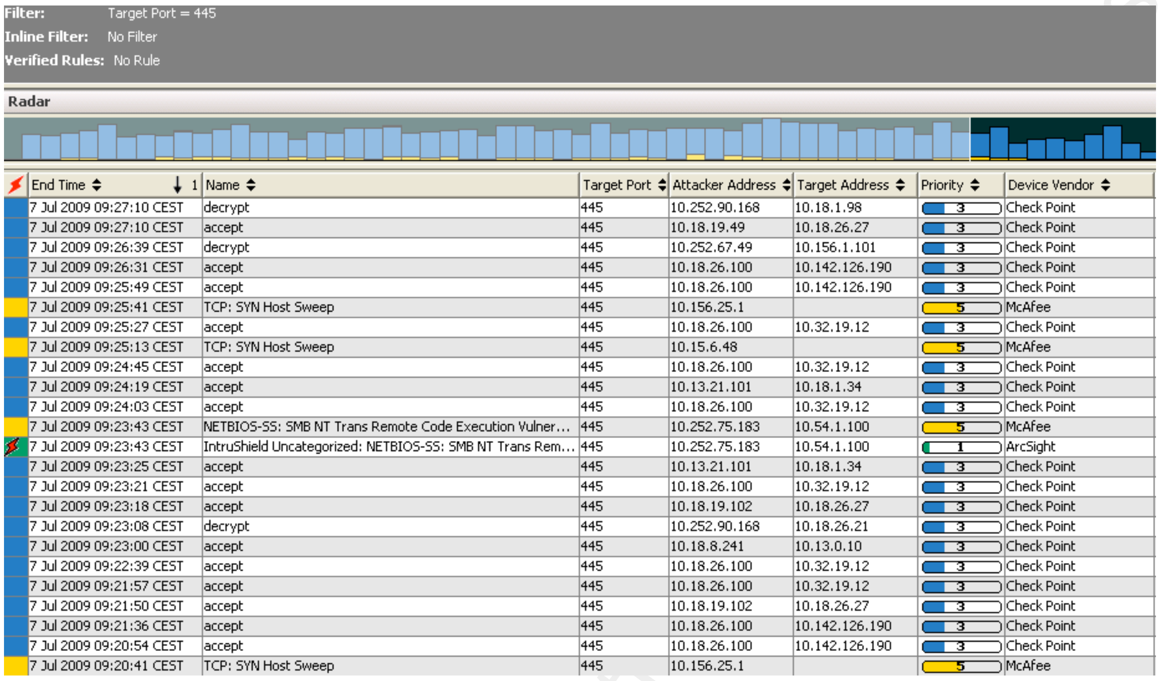
The Low priority rules detect each of the malicious acts individually. These rules parse against some event feeds fed into the Arcsight system (Voorhees, 2007). The net result is a few low priority rules detecting individual signatures of Conficker behavior and recording each event. The next step is to correlate each of the low priority alerts to increase the veracity and create a higher priority alert. For example combine the rules looking for TCP port 445 (and) UDP port 137 (and) www.getmyip .org to create a new detection rule.

Or more simplistically put a + b + c = MALWARE

This section examined how to logically define components of the Conficker worm. Now it is time to put some of the metrics to use for detection purposes. In the process of finding detectable metrics it's important to realize there are a lot of detectable metrics already defined in various papers that have been published. The more metric combinations the sharper and more balanced eventual detection rules are going to be. However there is going to be a limit to correlation (Arcsight, 2008). If too many different metrics of information are correlated then at some point the whole Arcsight process can be negatively impacted (Voorhees, 2007). This is why it is imperative that while establishing correlation metrics each metric is thoroughly tested for excessive noise and noise suppression possibilities. If the parameters of one metric are not correctly defined it can generate too much noise and have a negative impact on system performance.

2.2.1. Metric 1

Metric one: - SMB port 445



Filter: Target Port = 445
Inline Filter: No Filter
Verified Rules: No Rule

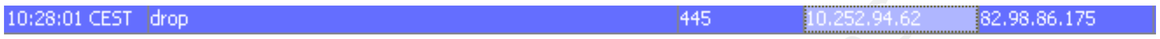
Radar

	End Time ↓ 1	Name	Target Port	Attacker Address	Target Address	Priority	Device Vendor
	7 Jul 2009 09:27:10 CEST	decrypt	445	10.252.90.168	10.18.1.98	3	Check Point
	7 Jul 2009 09:27:10 CEST	accept	445	10.18.19.49	10.18.26.27	3	Check Point
	7 Jul 2009 09:26:39 CEST	decrypt	445	10.252.67.49	10.156.1.101	3	Check Point
	7 Jul 2009 09:26:31 CEST	accept	445	10.18.26.100	10.142.126.190	3	Check Point
	7 Jul 2009 09:25:49 CEST	accept	445	10.18.26.100	10.142.126.190	3	Check Point
	7 Jul 2009 09:25:41 CEST	TCP: SYN Host Sweep	445	10.156.25.1		5	McAfee
	7 Jul 2009 09:25:27 CEST	accept	445	10.18.26.100	10.32.19.12	3	Check Point
	7 Jul 2009 09:25:13 CEST	TCP: SYN Host Sweep	445	10.15.6.48		5	McAfee
	7 Jul 2009 09:24:45 CEST	accept	445	10.18.26.100	10.32.19.12	3	Check Point
	7 Jul 2009 09:24:19 CEST	accept	445	10.13.21.101	10.18.1.34	3	Check Point
	7 Jul 2009 09:24:03 CEST	accept	445	10.18.26.100	10.32.19.12	3	Check Point
	7 Jul 2009 09:23:43 CEST	NETBIOS-SS: SMB NT Trans Remote Code Execution Vulner…	445	10.252.75.183	10.54.1.100	5	McAfee
	7 Jul 2009 09:23:43 CEST	IntruShield Uncategorized: NETBIOS-SS: SMB NT Trans Rem…	445	10.252.75.183	10.54.1.100	1	ArcSight
	7 Jul 2009 09:23:25 CEST	accept	445	10.13.21.101	10.18.1.34	3	Check Point
	7 Jul 2009 09:23:21 CEST	accept	445	10.18.26.100	10.32.19.12	3	Check Point
	7 Jul 2009 09:23:18 CEST	accept	445	10.18.19.102	10.18.26.27	3	Check Point
	7 Jul 2009 09:23:08 CEST	decrypt	445	10.252.90.168	10.18.26.21	3	Check Point
	7 Jul 2009 09:23:00 CEST	accept	445	10.18.8.241	10.13.0.10	3	Check Point
	7 Jul 2009 09:22:39 CEST	accept	445	10.18.26.100	10.32.19.12	3	Check Point
	7 Jul 2009 09:21:57 CEST	accept	445	10.18.26.100	10.32.19.12	3	Check Point
	7 Jul 2009 09:21:50 CEST	accept	445	10.18.19.102	10.18.26.27	3	Check Point
	7 Jul 2009 09:21:36 CEST	accept	445	10.18.26.100	10.142.126.190	3	Check Point
	7 Jul 2009 09:20:54 CEST	accept	445	10.18.26.100	10.142.126.190	3	Check Point
	7 Jul 2009 09:20:41 CEST	TCP: SYN Host Sweep	445	10.156.25.1		5	McAfee

FIG1 Target port 445

Following the process outlined in methodology, initially investigating events around the feeds in Arcsight for the SMB activity (CA, 2009) specifically on TCP port 445 as this is was mentioned as an attack vector (Leder, Werner, 2009). A rule was made to detect activity from an internal host on an enterprise network looking at multiple or single destinations on TCP port 445. FIG1 shows various activities in the active channel for destination port 445. Remember because of the associated interaction in Windows systems and server message block/ Common Internet File System traffic on TCP port 445, it is to a certain extent expected behavior in an enterprise network. Factor in that Conficker is primarily a windows based worm and is using TCP port 445 to spread (SRI international, 2009), then good evaluation for malicious verses non malicious is activity is needed. Also notice the 445 activity crosses platform and this can be seen in the device vendor/product column, this indicates that McAfee IntruShield and Checkpoint firewalls are detecting Conficker signature activity. Interestingly every event in FIG1 is of some interest because the name column contains a range of varying activity. Also in the Attacker Address and Target Address columns a few IP addresses appear multiple times. Focusing on the priority column there are some priority 3 events which are of particular interest because they are internal and they are allowed. This means they match the criteria defined in research (SRI International, 2009). The derived detection rule then written should fire when port 445 gets hit internally as this is how Conficker is spreading (SRI International, 2009). More specifically the focus of looking at internal hits is simply because external port 445 connections in this environment are blocked at the perimeter.

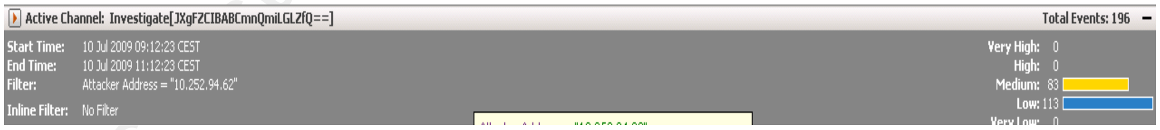
FIG1.1 port 445 outgoing traffic dropped at firewall.



```
10:28:01 CEST  drop                                      445    10.252.94.62    82.98.86.175
```

In FIG1 in the first row 'name' column, this would show up as 'Drop' this drop action is displayed in FIG 1.1. Important question here, how many times is the destination port 445 being accessed from the same source? This question is important because the differentiating between legitimate use of port 445 and malicious use of port 445 can be accomplished. To explain in more detail, what is the difference between a single hit from a source to destination and 500 hits from a single source to multiple destinations?

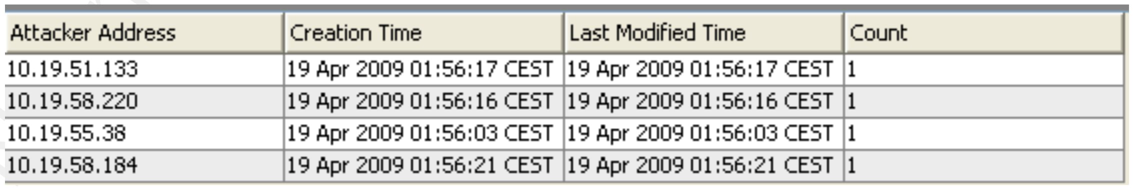
In answering this question the differentiation between malware and legitimate activity can begin. In FIG1 displayed in the name column are quite a number of 'accepts' this shows this activity to be allowed by the technology the event feed is originating from. However in reading the Conficker analysis (SRI International, 2009) it is clear Conficker is using TCP port 445 to spread. So look for a host which is excessively using TCP port 445. This can be accomplished by selecting an Attacker Address in the Attacker Address column of FIG1 and opening a channel to see how noisy this IP address is.



```
Active Channel: Investigate[JXgFZCIBABCmnQmiLGLZfQ==]                    Total Events: 196
Start Time:     10 Jul 2009 09:12:23 CEST                                Very High: 0
End Time:       10 Jul 2009 11:12:23 CEST                                     High: 0
Filter:         Attacker Address = "10.252.94.62"                           Medium: 83
Inline Filter:  No Filter                                                      Low: 113
```

FIG1.2 Results of two hour active channel

FIG1.2 shows an example of an active channel investigating an IP address detected as dropping multiple times while targeting internal hosts. The timeframe is a two hour period and the hits amounted to 196 in total. Evaluating these events does not reveal sufficient event volume to stand out as malicious. This is where writing a rule becomes very handy as it is possible to get the rule to look at an event pattern and record the results in an active list (Voorhees, 2007). Understanding the amount of time data is retained has a crucial role. This data retention time is important if Conficker tries to spread in an inconsistent manner, also during assimilation of different Conficker vectors for correlation purposes data retention time is crucial. Essentially if a host is infected via the NetBIOS vector (SRI International, 2009) the malware could attempt to propagate instantaneously via NetBIOS then wait for a finite time and attempt to propagate again via the same NetBIOS vector or a different malicious component signature. In retaining the data it is possible to track a host trying to propagate malware on different occasions. In this environment the attacker addresses are retained for a period of fifteen days, this timeframe mitigates any potential data overloads to Arcsight system memory. Naturally the data retention parameter can be changed should recording this information put to much stress on the Arcsight system. The number of times the rule fires is something which can be played with i.e. does a rule fire an alert on a host which hits port 445 15 times in 2 minutes or 1000 times in 30 seconds? It will depend on what background noise the network environment has. Generally it is good practice to experiment with different settings in order to achieve the best results. No two enterprise networks are the same so any detection rule will need tuning. Again these are parameters which can and should be played with in order to get the desired results, remember if there are windows machines on the internal network expect SMB traffic. It is important to filter out the real from the really bad when setting parameters for rules.



Attacker Address	Creation Time	Last Modified Time	Count
10.19.51.133	19 Apr 2009 01:56:17 CEST	19 Apr 2009 01:56:17 CEST	1
10.19.58.220	19 Apr 2009 01:56:16 CEST	19 Apr 2009 01:56:16 CEST	1
10.19.55.38	19 Apr 2009 01:56:03 CEST	19 Apr 2009 01:56:03 CEST	1
10.19.58.184	19 Apr 2009 01:56:21 CEST	19 Apr 2009 01:56:21 CEST	1

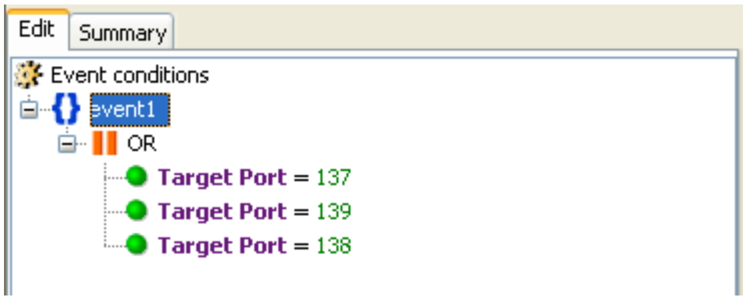
FIG2 Part of Active list with hosts hitting target port 445 one hundred and fifty times in one minute.

In review, what does this mean? It means that separating normal activity from the malicious activity so that if an internal host was to start hitting port 445 one hundred and fifty times to a singular or multiple destinations in one minute. An alert is raised for analysis and the offending host is added to an active list for fifteen days (FIG2). Additionally, some of the events were checked and investigated by analysts who look at the event and the surrounding timeframe for evidence of more Conficker attack vectors. Specifically the analysts here have an important role, they can act as confirmation that the rule written to detect malicious activity is providing the desired results or alternatively if the rule requires additional tuning. In this case the rule detects TCP port 445 activity deemed to be excessive and inherently suspicious. When the analysts investigate the alert they will look initially to evaluate the alert and then investigate events around the alert timeframe utilizing Arcsight as seen in FIG0 and FIG0.1. If during analysis a source host has exhibited a history of such high usage on TCP port 445, eventually the host has fired the rule, then a historical precedence exists for this host and a reasonable case for a false positive exits. Additionally should a network segment have had maintenance on a file share and the maintenance window has ended then the file server comes back online, TCP port 445 activities may breech the rule threshold and identify hosts as they re-negotiate with the file server. The important factor here is the analysts attempt to eliminate any reasonable doubt to the activities which lead to the rule firing and highlighting possible infections. Then following up on these possible infections the results were a reasonable amount of true positives, around forty percent. This number was derived by verifying the initial detection with an endpoint inspection by the analysts.

2.2.2. Metric 2

Metric two: - NetBIOS brute forcing.

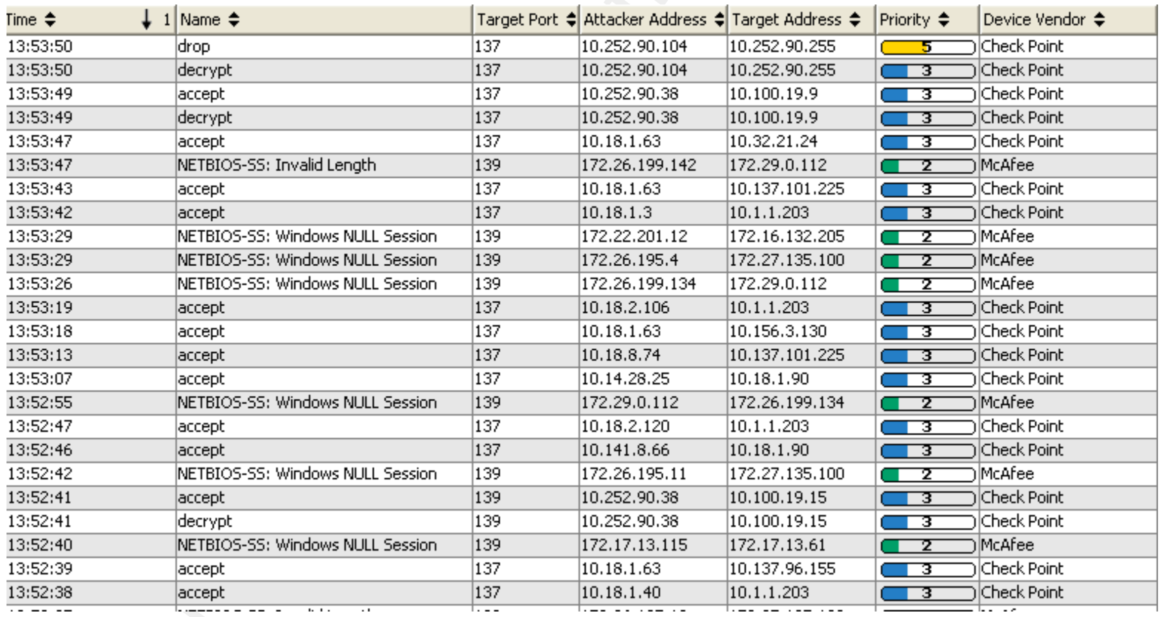
Metric 2 should follow a path similar to the approach taken with Metric 1 defined previously. In repeating the process of identifying metric one, the focus shifts to looking for the activity heading towards the NetBIOS target ports 137-139(SRI International, 2009) in an active channel.



```
Event conditions
  {} event1
     OR
       Target Port = 137
       Target Port = 139
       Target Port = 138
```

FIG3 Filter for NetBIOS active channel.

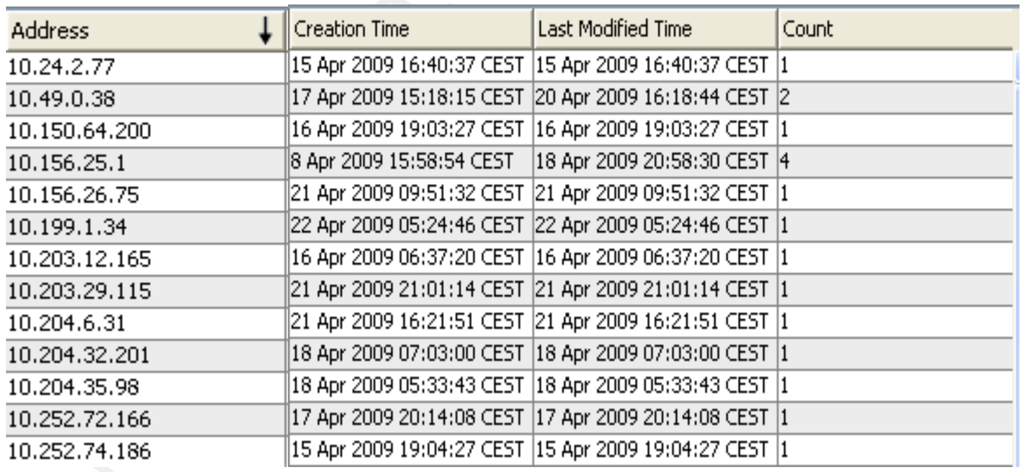
FIG3 shows an example of an active channel filter (Voorhees, 2007) which will isolate NetBIOS port activity. In defining this active channel an attempt to identify hosts which are using NetBIOS in a malicious manner which has been defined as an attack vector of Conficker (SRI International, 2009) is made. The filter has revealed some of the following results.



Time ↓ 1	Name	Target Port	Attacker Address	Target Address	Priority	Device Vendor
13:53:50	drop	137	10.252.90.104	10.252.90.255	5	Check Point
13:53:50	decrypt	137	10.252.90.104	10.252.90.255	3	Check Point
13:53:49	accept	137	10.252.90.38	10.100.19.9	3	Check Point
13:53:49	decrypt	137	10.252.90.38	10.100.19.9	3	Check Point
13:53:47	accept	137	10.18.1.63	10.32.21.24	3	Check Point
13:53:47	NETBIOS-SS: Invalid Length	139	172.26.199.142	172.29.0.112	2	McAfee
13:53:43	accept	137	10.18.1.63	10.137.101.225	3	Check Point
13:53:42	accept	137	10.18.1.3	10.1.1.203	3	Check Point
13:53:29	NETBIOS-SS: Windows NULL Session	139	172.22.201.12	172.16.132.205	2	McAfee
13:53:29	NETBIOS-SS: Windows NULL Session	139	172.26.195.4	172.27.135.100	2	McAfee
13:53:26	NETBIOS-SS: Windows NULL Session	139	172.26.199.134	172.29.0.112	2	McAfee
13:53:19	accept	137	10.18.2.106	10.1.1.203	3	Check Point
13:53:18	accept	137	10.18.1.63	10.156.3.130	3	Check Point
13:53:13	accept	137	10.18.8.74	10.137.101.225	3	Check Point
13:53:07	accept	137	10.14.28.25	10.18.1.90	3	Check Point
13:52:55	NETBIOS-SS: Windows NULL Session	139	172.29.0.112	172.26.199.134	2	McAfee
13:52:47	accept	137	10.18.2.120	10.1.1.203	3	Check Point
13:52:46	accept	137	10.141.8.66	10.18.1.90	3	Check Point
13:52:42	NETBIOS-SS: Windows NULL Session	139	172.26.195.11	172.27.135.100	2	McAfee
13:52:41	accept	139	10.252.90.38	10.100.19.15	3	Check Point
13:52:41	decrypt	139	10.252.90.38	10.100.19.15	3	Check Point
13:52:40	NETBIOS-SS: Windows NULL Session	139	172.17.13.115	172.17.13.61	2	McAfee
13:52:39	accept	137	10.18.1.63	10.137.96.155	3	Check Point
13:52:38	accept	137	10.18.1.40	10.1.1.203	3	Check Point

FIG4 Active channel showing NetBIOS activity in Arcsight.

FIG4 reveals some interesting results. Again as with Metric 1 there are different platforms shown with some accept, drop and intrusion detection/prevention system signatures covering different investigation possibilities. In a methodical fashion to the previous metric parameters must be set correctly for a rule to fire on events of interest. Remember in Metric 1 the use of an active channel to identify events of interest led to the creation of a rule (Voorhees, 2007). The rule parameters use time to start determining what can be just normal network noise and what could be malicious activity. Then once the malicious noise parameters are set the rule is created to record possible infections to an active list. Metric 2 rule fires one hundred and fifty times to a singular or multiple destinations in one minute internally. Another aspect of these alerts to consider is the type of action that occurred. The action is captured in the column “Name”. Figure 4 indicates in the “Name” column as with Metric 1 some accepts, drops and some IDS/IPS signature have occurred. Again this has relevance in that the attack vector shows NetBIOS activity to “exploit weak security controls in enterprises and home networks” (SRI International, 2009). Finally the last step as with Metric 1 sees the results recorded to an active list and stored with a time to live of fifteen days.



Address	Creation Time	Last Modified Time	Count
10.24.2.77	15 Apr 2009 16:40:37 CEST	15 Apr 2009 16:40:37 CEST	1
10.49.0.38	17 Apr 2009 15:18:15 CEST	20 Apr 2009 16:18:44 CEST	2
10.150.64.200	16 Apr 2009 19:03:27 CEST	16 Apr 2009 19:03:27 CEST	1
10.156.25.1	8 Apr 2009 15:58:54 CEST	18 Apr 2009 20:58:30 CEST	4
10.156.26.75	21 Apr 2009 09:51:32 CEST	21 Apr 2009 09:51:32 CEST	1
10.199.1.34	22 Apr 2009 05:24:46 CEST	22 Apr 2009 05:24:46 CEST	1
10.203.12.165	16 Apr 2009 06:37:20 CEST	16 Apr 2009 06:37:20 CEST	1
10.203.29.115	21 Apr 2009 21:01:14 CEST	21 Apr 2009 21:01:14 CEST	1
10.204.6.31	21 Apr 2009 16:21:51 CEST	21 Apr 2009 16:21:51 CEST	1
10.204.32.201	18 Apr 2009 07:03:00 CEST	18 Apr 2009 07:03:00 CEST	1
10.204.35.98	18 Apr 2009 05:33:43 CEST	18 Apr 2009 05:33:43 CEST	1
10.252.72.166	17 Apr 2009 20:14:08 CEST	17 Apr 2009 20:14:08 CEST	1
10.252.74.186	15 Apr 2009 19:04:27 CEST	15 Apr 2009 19:04:27 CEST	1

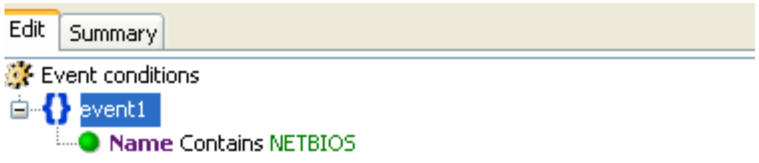
FIG5 Active list tracking Hosts on NetBIOS brute forcing

Looking at the output of the active list in FIG5 note the time based columns and in particular the ‘count’ column it is noticeable that a host has fired the rule on four separate occasions over a period of ten days. The results are again sent through to an analyst who as explained with metric 1, evaluates the rule fire and the events around the resulting alert. The analyst again should take very effort to eliminate false positives, NetBIOS does broadcast which by default is noisy or again if a host has not been connected to the LAN for sometime may become inadvertently noisy when reconnected to the LAN. Good analysis undertaken in a logical way can help to lower the false positive rate of the Arcsight rule. Previously as seen in FIG0 and FIG0.1 the tooling in Arcsight can be harnessed to evaluate certain specifics around the detection rule which should be investigated to help decide the eventual likelihood of a true or a false positive. Does the host broadcast a lot on its NetBIOS port? Was this a legitimate use of LAN communication between two specific hosts? These are two examples of questions which should look to be answered during analysis to eliminate reasonable doubt. Referring to Conficker propagation over NetBIOS "open network shares and brute force password attempts using a list of over 240 common passwords" In particular, it copies itself to the admin share or the IPC (inter-process communication) share launched using rundll32.exe (SRI International, 2009). An analyst can use this information by looking for these specifics in association with the rule fire e.g. the use of the admin share or some associated passwords used in the brute forcing. During the course of tracking these investigated events and their feedback results; the true positive rate for infection was around twenty percent. This is far from a good result however in spite of attempts to tune the rule the true positive rate would not rise. The net result makes this a good rule to correlate against yet underlines the fact that this rules severity should be so low.

2.2.3. Metric 3

Metric 3: - Intrusion detection/prevention signatures.

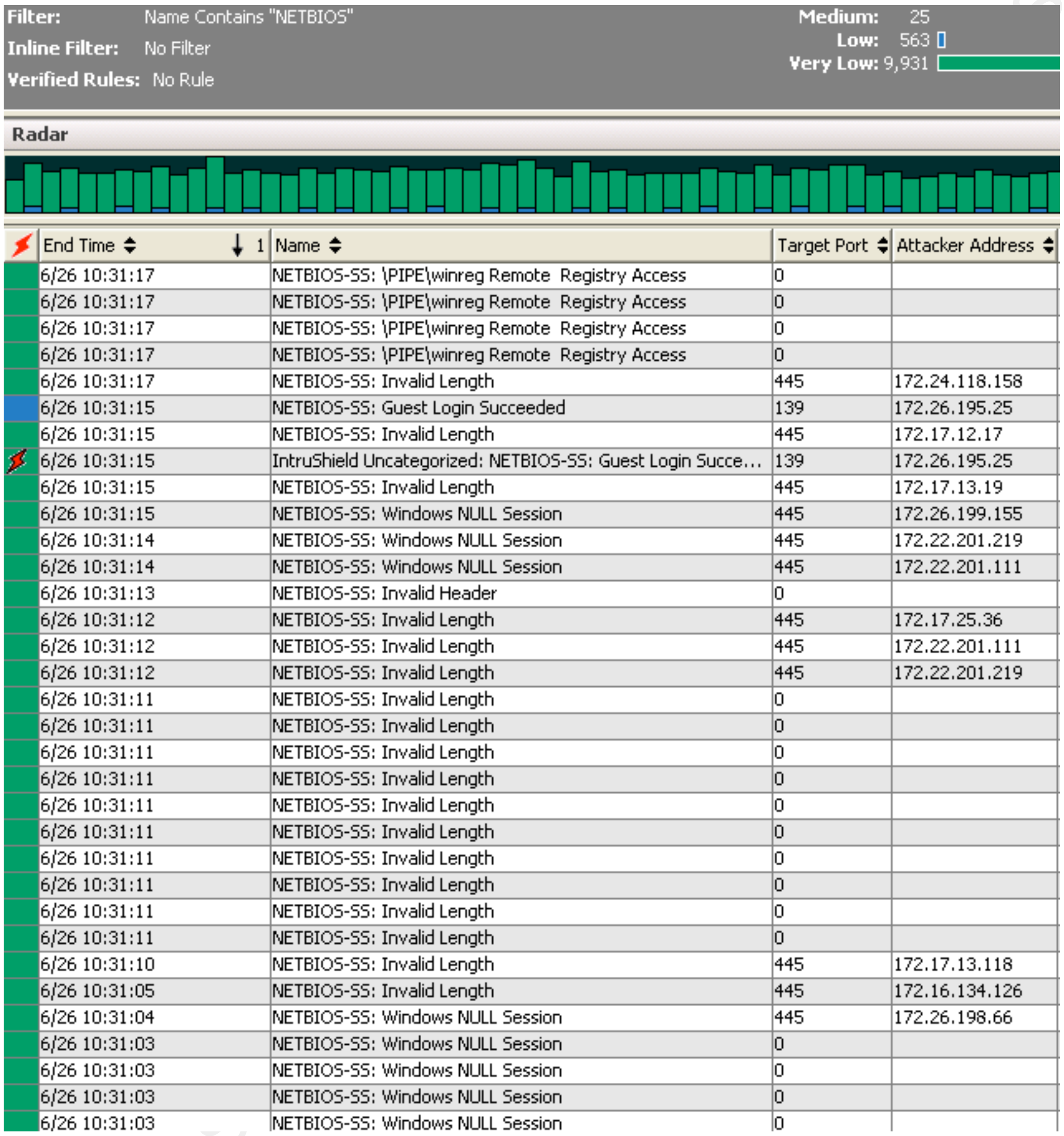
As was viewable in the active channels (FIG1, FIG4) included in the previous Metric definitions there are some Intrusion detection and prevention signatures which have alerted. At this point it is important to realize at the time Conficker first came to prominence there were no Intrusion detection/prevention signatures written specifically for Conficker (BT, 2009). Evaluating and assessing IDS/IPS systems on the basis they have no specific signatures deployed when malware propagates gains nothing even if custom signatures can take time to be deployed. The approach taken here was to investigate some of the IDS platforms feeding into Arcsight. Running an active channel as previously seen in the active channels (FIG1, FIG4) for Metric 1 and 2 there are some signatures which could be of interest. While investigating further continue with setting the active channel to attempt to find appropriate IDS/IPS signatures.



Event conditions
event1
Name Contains NETBIOS

FIG6 Simple filter to identify NetBIOS signature in Arcsight active channel.

Now here comes the difficult part! Patience and extreme caution is required when using the intrusion detection signatures. This requirement stems from the fact that there are a lot of IDS/IPS signatures and even after putting in a lot of effort to isolate a signature that discovers Conficker, it is always a possibility one can not be identified. This manifests itself in vast amounts of signature fires which after analysis are deemed false positives.



Filter: Name Contains "NETBIOS"
Inline Filter: No Filter
Verified Rules: No Rule

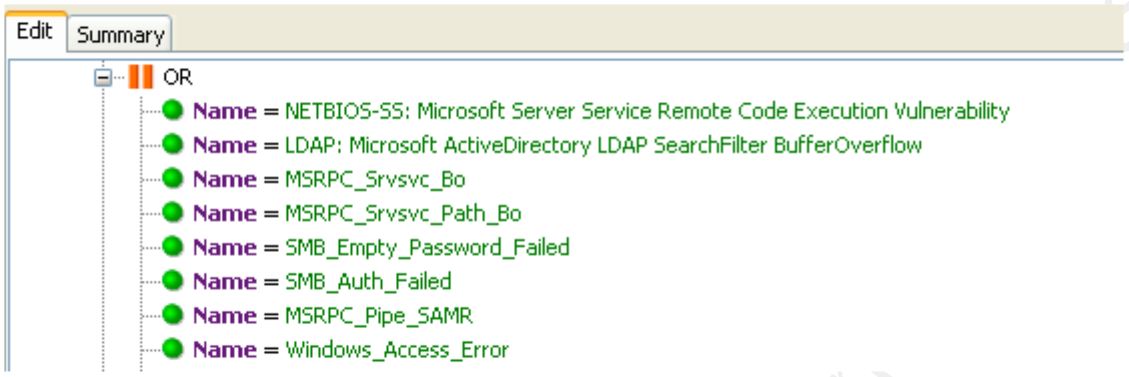
Medium: 25
Low: 563
Very Low: 9,931

Radar

⚡	End Time	Name	Target Port	Attacker Address
🟩	6/26 10:31:17	NETBIOS-SS: \PIPE\winreg Remote Registry Access	0	
🟩	6/26 10:31:17	NETBIOS-SS: \PIPE\winreg Remote Registry Access	0	
🟩	6/26 10:31:17	NETBIOS-SS: \PIPE\winreg Remote Registry Access	0	
🟩	6/26 10:31:17	NETBIOS-SS: \PIPE\winreg Remote Registry Access	0	
🟩	6/26 10:31:17	NETBIOS-SS: Invalid Length	445	172.24.118.158
🟦	6/26 10:31:15	NETBIOS-SS: Guest Login Succeeded	139	172.26.195.25
🟩	6/26 10:31:15	NETBIOS-SS: Invalid Length	445	172.17.12.17
⚡	6/26 10:31:15	IntruShield Uncategorized: NETBIOS-SS: Guest Login Succe...	139	172.26.195.25
🟩	6/26 10:31:15	NETBIOS-SS: Invalid Length	445	172.17.13.19
🟩	6/26 10:31:15	NETBIOS-SS: Windows NULL Session	445	172.26.199.155
🟩	6/26 10:31:14	NETBIOS-SS: Windows NULL Session	445	172.22.201.219
🟩	6/26 10:31:14	NETBIOS-SS: Windows NULL Session	445	172.22.201.111
🟩	6/26 10:31:13	NETBIOS-SS: Invalid Header	0	
🟩	6/26 10:31:12	NETBIOS-SS: Invalid Length	445	172.17.25.36
🟩	6/26 10:31:12	NETBIOS-SS: Invalid Length	445	172.22.201.111
🟩	6/26 10:31:12	NETBIOS-SS: Invalid Length	445	172.22.201.219
🟩	6/26 10:31:11	NETBIOS-SS: Invalid Length	0	
🟩	6/26 10:31:11	NETBIOS-SS: Invalid Length	0	
🟩	6/26 10:31:11	NETBIOS-SS: Invalid Length	0	
🟩	6/26 10:31:11	NETBIOS-SS: Invalid Length	0	
🟩	6/26 10:31:11	NETBIOS-SS: Invalid Length	0	
🟩	6/26 10:31:11	NETBIOS-SS: Invalid Length	0	
🟩	6/26 10:31:11	NETBIOS-SS: Invalid Length	0	
🟩	6/26 10:31:11	NETBIOS-SS: Invalid Length	0	
🟩	6/26 10:31:11	NETBIOS-SS: Invalid Length	0	
🟩	6/26 10:31:11	NETBIOS-SS: Invalid Length	0	
🟩	6/26 10:31:10	NETBIOS-SS: Invalid Length	445	172.17.13.118
🟩	6/26 10:31:05	NETBIOS-SS: Invalid Length	445	172.16.134.126
🟩	6/26 10:31:04	NETBIOS-SS: Windows NULL Session	445	172.26.198.66
🟩	6/26 10:31:03	NETBIOS-SS: Windows NULL Session	0	
🟩	6/26 10:31:03	NETBIOS-SS: Windows NULL Session	0	
🟩	6/26 10:31:03	NETBIOS-SS: Windows NULL Session	0	
🟩	6/26 10:31:03	NETBIOS-SS: Windows NULL Session	0	

FIG7 Output of Filter looking for name contains "NetBIOS"

Unfortunately there are a lot of signatures and it is apparent in FIG7 channel output that because of this noise a lot of extra filtration work is necessary. Before a more specific metric is identified and that the correct parameters are applied to get a useful alert some signature evaluation and eventual suppression will be needed. In order to accomplish this, a slightly different approach was taken. This approach followed using meticulously evaluated signatures logically (OR) together.


Edit Summary

OR
- Name = NETBIOS-SS: Microsoft Server Service Remote Code Execution Vulnerability
- Name = LDAP: Microsoft ActiveDirectory LDAP SearchFilter BufferOverflow
- Name = MSRPC_Srvsvc_Bo
- Name = MSRPC_Srvsvc_Path_Bo
- Name = SMB_Empty_Password_Failed
- Name = SMB_Auth_Failed
- Name = MSRPC_Pipe_SAMR
- Name = Windows_Access_Error

FIG8:- Intrusion detection signatures OR'ed together after signature evaluation.

Examining FIG7 reveals a number of signatures evaluated and found to be detecting some facets of Conficker. More specifically the 'NetBIOS-SS Microsoft server service remote code execution' signature was made to detect the MS08-067 vulnerability (Microsoft, 2009) which was specifically mentioned as a Conficker attack vector (Nahorney, 2009).Then in the case of 'SMB_Auth_Failed' this identifies a signature which "detects an excessive number of failures to authenticate to an SMB share. This may indicate a username/password guessing attack" (Yason, 2009). This SMB activity was mentioned as an attack vector of Conficker (SRI International, 2009). Then a signature such as 'MSRPC_Pipe_SAMR' which informs the: - "security event is categorized as an audit event. It is not necessarily indicative of an attack or threat to a network. This signature reports access attempts to the NT Security Accounts Manager (SAM) Database Management Services using named pipes" (Yason, 2009). This serves to highlight an excellent point that the 'MSRPC_Pipe_SAMR' signature pre dates the Conficker signatures (Yason, 2009). This proves IDS/IPS signatures can be useful if time and patience permit the correct signatures are identified before customized Conficker signatures were released (Yason, 2009). It must be stressed here as a cautionary note this signature evaluation is done internally and evaluated with direct feedback from endpoint hosts! This direct feedback meant signatures were identified by populating active channels and investigating the events until suspicion was sufficient to warrant host investigation. The host investigation means an analyst visited the suspected host and checked for signs of infection. The resulting rule is looking for the defined IDS/IPS signature to fire multiple times from the same source to the same internal destination. This means for internal hosts which fire an IDS/IPS signature more than twenty five times in 2 minutes the source address in the Arcsight alert is added to another active list for fifteen days. Then naturally the alert was analyzed accordingly by an analyst in a method similar to one which has been highlighted in previous metrics. The difference is that here a specific platform (IDS/IPS) has been used in an attempt to detect worm activity while no specific signature exists. This could be recipe for a vast amount of false positives which analysts have to deal with in order to perform an effective evaluation. To be more specific an analyst would look to the payload capture facility of the IDS/IPS system to evaluate the signature fire in more detail. Certainly the signature pattern is there however what else incriminating can be seen in the packets? Also is the source or destination host firing on multiple IDS/IPS signatures around the same timeframe? These are again events which can be evaluated using Arcsight tooling as shown in FIG0, FIG0.1 and will help analysts reach a more accurate conclusion. The feedback from analysis showed true positive rates of sixty percent, this is again significantly less than desired yet an improvement on metric 2.

2.2.4. Metric 4

Metric four: - IP addresses

Studying the Conficker agent process the worm has been programmed to carry out a number of tasks upon infection (SRI International, 2009). These tasks include procedures such as "proceeds to one of the following sites to obtain its external-facing IP address *www.getmyip.org, getmyip.co.uk*, and *checkip.dyndns.org* and attempts to download the GeoIP database from *maxmind.com*". This is for purposes of finding out where it has infected a machine will try to find out where in the world it is or what its external facing IP address is (SRI International, 2009). These URLs can be translated to IP addresses, by utilizing a name translation service. In continuing to follow the methodology originally set out in defining metrics create an active channel and use it to look for activity going to the external IP addresses derived from work disclosing this as valid Conficker activity (SRI International, 2009). Once the active list (Voorhees, 2007) confirms activity from internal hosts going to these enumerated IP addresses it is time to move on to the next step. Finally the IP addresses which could be enumerated should be assimilated into a rule and the rule populates an active list. The active list collects internal source IP addresses and kept them for a period of fifteen days.

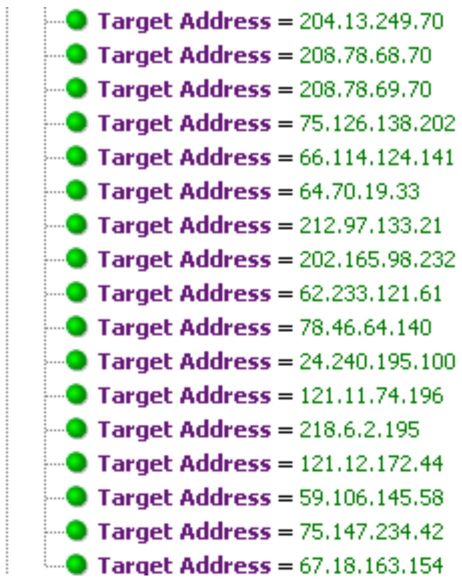
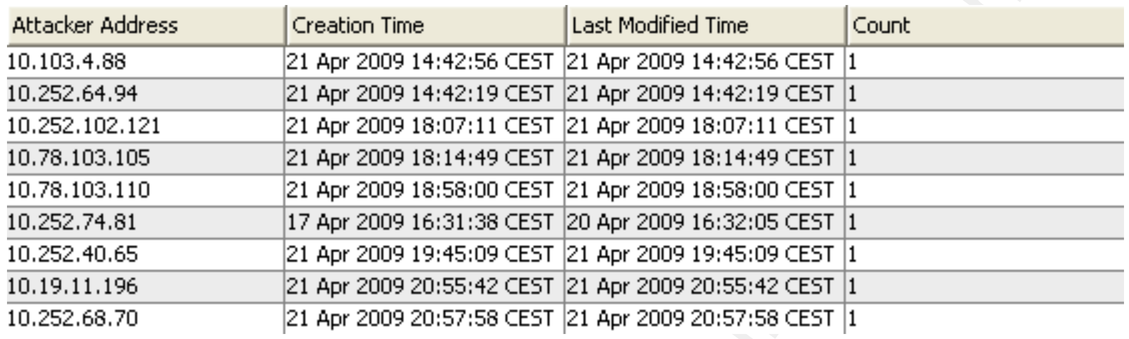

```
Target Address = 204.13.249.70
Target Address = 208.78.68.70
Target Address = 208.78.69.70
Target Address = 75.126.138.202
Target Address = 66.114.124.141
Target Address = 64.70.19.33
Target Address = 212.97.133.21
Target Address = 202.165.98.232
Target Address = 62.233.121.61
Target Address = 78.46.64.140
Target Address = 24.240.195.100
Target Address = 121.11.74.196
Target Address = 218.6.2.195
Target Address = 121.12.172.44
Target Address = 59.106.145.58
Target Address = 75.147.234.42
Target Address = 67.18.163.154
```

FIG11 Enumerated destination IP addresses.

Naturally it is entirely possible that regardless of system which logs events and feeds them into Arcsight name resolution has already take place. In this case the active channel can be set to look for the target URL itself and in the case of the source there is no impact as this will be recorded to an active list in exactly the same way an IP address will. The method of using a filter of target IP addresses and getting possibly infected internal IP addresses as a result proved to be the method which generated the most fires. The difficulty here is really the need to be looking at singular hits. One host hitting one IP address, one time and record it to an active list. This generates a lot of entries getting added to the active list and will need suppression. The suppression needed in this case can be executed by changing an active list parameter and adding a feedback loop to the rule. In keeping the active list timer down to around forty eight hours to flush out the sheer number or recorded addresses will force the active list to remain populated with data refreshed and keeps the list size smaller.



Attacker Address	Creation Time	Last Modified Time	Count
10.103.4.88	21 Apr 2009 14:42:56 CEST	21 Apr 2009 14:42:56 CEST	1
10.252.64.94	21 Apr 2009 14:42:19 CEST	21 Apr 2009 14:42:19 CEST	1
10.252.102.121	21 Apr 2009 18:07:11 CEST	21 Apr 2009 18:07:11 CEST	1
10.78.103.105	21 Apr 2009 18:14:49 CEST	21 Apr 2009 18:14:49 CEST	1
10.78.103.110	21 Apr 2009 18:58:00 CEST	21 Apr 2009 18:58:00 CEST	1
10.252.74.81	17 Apr 2009 16:31:38 CEST	20 Apr 2009 16:32:05 CEST	1
10.252.40.65	21 Apr 2009 19:45:09 CEST	21 Apr 2009 19:45:09 CEST	1
10.19.11.196	21 Apr 2009 20:55:42 CEST	21 Apr 2009 20:55:42 CEST	1
10.252.68.70	21 Apr 2009 20:57:58 CEST	21 Apr 2009 20:57:58 CEST	1

FIG12 List of IP addresses hitting suspicious IP addresses externally

As can be seen in FIG12 all alerts have a count of 1. The suppression loop added to the rule used for analysis with this metric takes the form of adding the active list populated into the rule itself. So the rule cannot fire on an IP address source more than once. This means the active list had to be put into the rule as a NOT statement and suppresses multiple alerts. The reason for this is that the number of times the rule fired was just too great and adding the active list as a 'NOT' suppresses the rule from firing on a previously seen host. Doing this can cause the rule to fire at an incredible rate when activated but fires drop exponentially as the active list populates and suppression kicks in.

Inline with previous methods this rule once fired resulting in an alert being analyzed and investigated. In order to carry out analysis here a premium must be placed on what else an incriminated host has done around the time of the rule fire. This time zone is important simply because the rule is only looking for a connection attempt to an external address. So using Arcsight in an attempt to find more events can increase the suspicion that a host is infected to provide the deciding factor in the decision to continue the investigation. In the case enterprise networks which are monitored and fed into the Arcsight tool are spanning a global infrastructure; it is not outside the bounds of reality to have some local applications which use a Geographical location tool (SRI International, 2009). That could make the host accessing maxmind.com a false positive. Alternatively if the application included a component which had to retrieve its external facing IP address each time the application ran this component then an analyst is unknowingly presented with a false positive. However if analysis reveals the incriminated host also attempting to suddenly access other blacklisted destinations after the rule fire it increases the probability of finding a true positive. Regardless of the analytical ability, rules like this will create a lot of noise which can overwhelm analysts with investigations to the extent the analyst has little time to focus on other possibilities. The resulting host investigations led to disappointing results confirmed by true positive detection rate of only fifteen percent. This is a less than desirable result and is obfuscated in the fact that in this case systems could legitimately hit theses destination IP addresses as a part of their local functionality. This ultimately is a possible explanation for such high rates of false positives.

So in taking a moment to review what is defined above there are low priority detection rules and active lists populated with suspected infected hosts defined. In this case active lists of internal IP addresses which have been flagged as potentially infected with Conficker. Realistically these detection rules can be left in place and it should be possible to find some infections but as previously mentioned true positive rates are lower than desired. Quite often security analysts are encouraged to 'think out of the box' In embracing this concept analysts can take some rules which all look for Conficker attack vectors with low rates of true positives and correlate them together!

2.3 Step 3: Correlating the different metrics

All of the previously define metrics provided alerts which were evaluated by a team of security analysts with some interesting results. During analysis the analysts will have used the Arcsight tool to investigate traffic patterns to and from each suspect host around the time of the alert. As an example the analysts will open an active channel around the time of the event and look of other Conficker component metrics to show themselves (SRI International, 2009). Sometimes the alerts came to nothing but sometimes there were confirmed true positives. The analysts in this case were using small aspects of correlation and individual components of a well engineered (SRI International, 2009) and costly (Cyber Secure Institute, 2009) worm. However defining each of the metrics by defining individual detection rules produces good results but does not give a complete picture. Deciding to combine these individual detection rules introduces the objective of finding more true positives. It is crucial to remember the individual rules were in their own way finding some true positives yet this is simply not good enough. What can be achieved is the creation of a rule which will look to combine and enhance the rules which are already in place and significantly enhancing the true positive yield.

Examine Diagram 3 below.

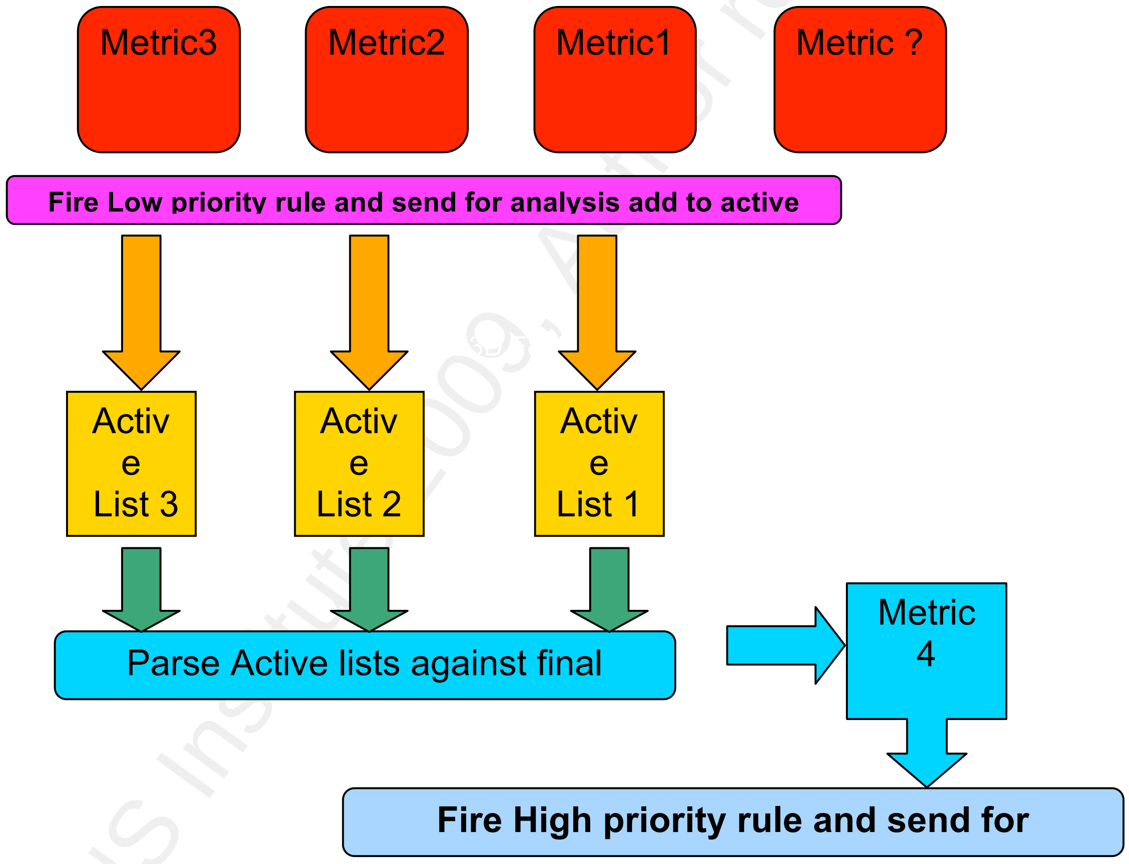

Metric3
Metric2
Metric1
Metric ?
Fire Low priority rule and send for analysis add to active
Active List 3
Active List 2
Active List 1
Parse Active lists against final
Metric 4
Fire High priority rule and send for

Diagram 3 Flow process of rule correlation.

Dereck Haye Olbaidle@gmail.com

The metrics in red are the metrics discussed previously. These metrics individually fire a low priority rule for analysis and add the results to Individual active lists shown in the diagram as yellow boxes. Then the cumulative active lists results are parsed of against another metric. Basically the results of the low priority rule are taken and reused, in reusing them they are components of correlation (Arcsight, 2008). However to complete the correlation process using the Arcsight tool an effective way of triggering the correlated results must be defined.

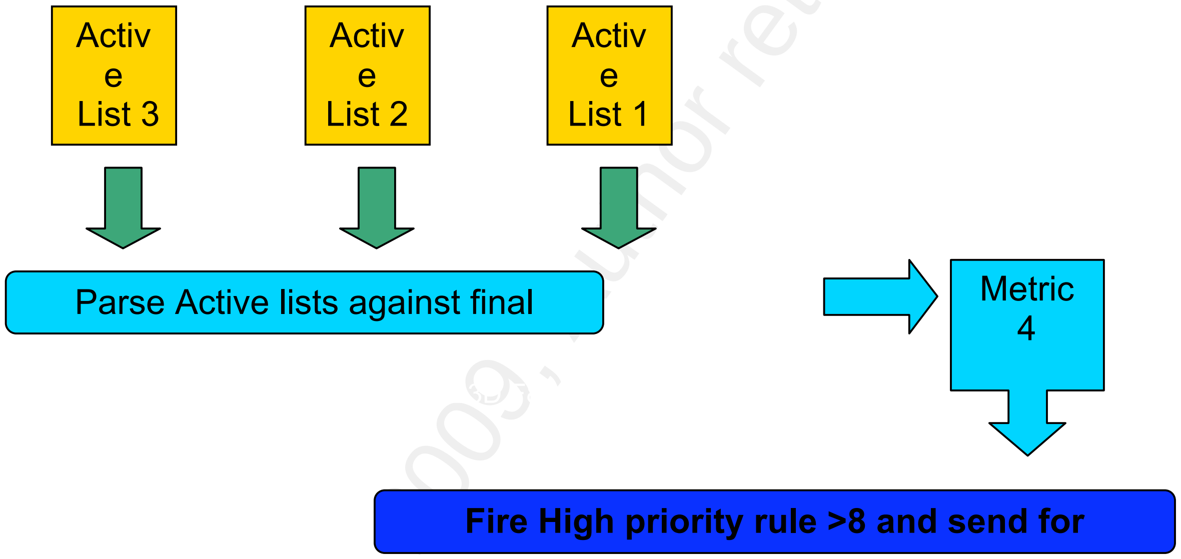

Activ
e
List 3
Activ
e
List 2
Activ
e
List 1
Parse Active lists against final
Metric
4
Fire High priority rule >8 and send for

Diagram 4 Final component of Rule correlation.

Since the active lists have assimilated a list of suspected Conficker infections it is advantageous to define exactly what must happen to them in order to fire the high priority rule. So what exactly must happen to a host to get it flagged as a priority 9 or 10 event?

First the host must have fired one of the low priority rules. Then the event was manually evaluated by analysts and the result was either a host was investigated or insufficient evidence was identified resulting in adding the host to an active list. Then if that host which was already suspected of a Conficker infection fired a second rule within the active list timeframe set, then that host has displayed two separate vectors of Conficker. Now what is needed is logically constructing the high priority rule and tuning it in such a way that it can accurately detect Conficker. This requires some experimentation and development to play various active lists and metrics against each other to identify what is most suitable for the last metric.

Remember all that is taking place is correlating a list of Internal IP addresses gained through smaller rules against a list of external IP addresses. In defining a correlation rule logically the goal is correlating using multiple metrics and combining them in one rule. The result was to have one rule which would wait to see of any of the hosts on the active lists populated by the smaller rules would again fire on metric 4. So the net result of correlation could chronologically be as follows: - Conficker infects a host, the infected host uses metric 1 to try and infect another host, metric 1 fires a low priority rule, the resulting alert is analyzed, the host is added to an active list, the host tries to execute metric 4 and a high priority rule fires.

The results after the correlation immediately looked very interesting. Initially the rule was fired at a developmental priority 0 to complete evaluation. After the first occurrences of the high priority rule fires were thoroughly investigated by analysts the hosts suspected of infection were investigated for Conficker infections. Then the true positives started coming back! These were verified by getting the endpoint host machines investigated, often the name of the malware discovered was recorded:-

```
"Please close the following ticket(s) with status "True Positive":
IMxxxxxxx: W32.Downadup infection Best regards,"
```

Using correlation the logic works on different levels. Correlating rules which detect different Conficker vector metrics has yielded a very high ratio of true positives. Interestingly if an analyst has investigated a low priority event and there was not enough evidence to convince that the host was infected with Conficker. Then crucially that information is still retained via active lists for a defined time. Alternatively it is possible that the low priority event was missed by a busy analyst team. Often when the high priority rule was fired, verifying the event analysis reveals long periods between a host firing on the first metric and then on the second metric. This confirms the need to set some active lists with a time to live of fifteen days. In some cases when the High priority rule fired thirteen days had passed between that host getting added to is original active list and using metric 4, and the alert still came back as a true positive!

Another excellent result of getting the active list time parameters right manifested itself on Conficker re-infections. Re-infections occur where a host was identified as infected then cleaned yet after the host was cleaned or re-imaged a failure to patch the host resulted in re-infection this naturally is a process failure. Unfortunately this process failure represents a real world scenario where a lack of understanding initial infection vectors of malware (SRI International, 2009) lead to inefficient mitigation. In this eventuality the time state is crucial if a low priority alert first fired on the Monday and analysts asked for an investigation. Then the HIGH priority rule fires the following Monday for the same source IP address. This means two different components of Conficker have been detected via correlation on one host over a period of seven days. Now either the machine was not investigated after the initial infection vector was detected or if it has been cleaned it has been re-infected during that seven day period. In constructing the rule correlation this way could be thought of as a fail safe mechanism. The high priority rule has returned true positive rates of ninety percent. This number proves, in this instance, correlating Conficker activity returns much higher rates of true positives than individual metrics.

2.3. Recommendations

It is not for just any old reason that the leaders of security community have pushed and explained that organizations in general need to focus on analyzing their logging. Methodically understanding malware components and using a correlation engine to detect these individual components (Arcsight, 2008) occurring with co-ordination is an incredibly powerful detection metric. All the different types of Malware out there do not have a uniform behavior Pattern so the detection mechanisms used should reflect this.

As an example, Monday there may be a drop on an internal firewall interface from address A on port 6666, then on Friday the same source IP address A could fire a drop on Thursday on port 6667 finally on Wednesday the following week IP address A on port 7777.   The point here is that the firewall log sees these events but in a SIEM environment there exists the ability to correlate one event with another. The capabilities to correlate one event with another have an incredibly powerful ability if a base of interesting events is isolated. Then against the events of interest start correlating in other metrics. These could take the form of certain ports associated with a Trojan horse or perhaps blacklists of IP addresses maintained by some security vendors. Perhaps some ports which have been mentioned in Dshield? How about looking for more extreme metrics? Perhaps looking into the number of bytes out from, or to a particular IP address? Bytes out might reveal end users sending out data or sending out data when nobody is sitting at the workstation at night? Should anyone who reads this whitepaper actually follow the methods used, it is advisable to take more time to analyze the traffic actually dropped on the inside perimeter of their network trying to get out and getting dropped. Certainly by the very definition 'dropped traffic' results will contain some excellent metrics to correlate against at the very least. The detection rules created in such a manner are also invaluable to any SIEM/Arcsight developer. Especially the statistics which can be garnered and fed back into the system in the form of true/false positive information. Even just to track the most infected segments of a network and recommend architectural actions based off this information. Remember Logs are an absolute goldmine of information. This whitepaper is recommending that plenty of time be taken to sift through exactly what has taken place and how to break that information down. Evaluate it, reuse it and correlate it against other logs. Its one thing to look into event logging but another thing to understand HOW or WHERE to look in the logging *drop and log!

A + B + C = malware

2.4 Conclusion

If an organization has multiple security devices capable of logging deployed and with logging enabled event correlation is possible. In the case of Conficker the worm itself has a lot of components and facets of its behavior can be daunting to evaluate. However even the most inexperienced security analyst can, given time, focus on areas of the worm and detect it. This white paper has shown that more than one component of a worm can be detected. That it is possible to detect the worm and track its state over time. Also that rapidly spreading malware like this is detectable and with a very high rate of detection success. What must be achieved now is with each evolution of Conficker keeping pace with its variants by defining the new metrics and correlating against them. This approach takes time and patience and in many ways follows the path the worm authors have taken. Intrusion detection/prevention systems, proxies, firewalls and any security device for that matter generate massive logs. This whitepaper concludes that taking time to isolate the correct events in logs can save organizations time and effort by accurately detecting malware and aiding effectively in preventing malware from spreading.

3. References

Arcsight (2008). *Mitigating Fraud with the ArcSight SIEM Platform*. Retrieved from http://www.arcsight.com/collateral/ArcSight_Mitigating_Fraud.pdf

British Broadcasting Corporation (2009 March 30). Timeline: The Conficker worm. Retrieved June 15, 2009, from British broadcasting corporation Web site: http://news.bbc.co.uk/1/hi/technology/7973829.stm

Briddell, Matt (2002 April 04). A Comprehensive Perimeter Security Architecture for GIAC Enterprises. Retrieved January 28, 2009, Web site: https://www.sans.org/reading_room/whitepapers/honors/a_comprehensive_perimeter_security_architecture_for_giac_enterprises_839

BT (2009). Appendix A IDS/IPS signatures which may detect Conficker.C presence Retrieved May 05, 2009, from http://bt.counterpane.com Web site: http://bt.counterpane.com/BT_MSSG_RA_Conficker.C_Update2.pdf

CA (2009 January 05). Win32/Conficker.B. Retrieved Feb 11, 2009, from http://www.ca.com Web site: http://www.ca.com/securityadvisor/virusinfo/virus.aspx?id=76852

Cyber Secure Institute (2009, April 23). Cyber Secure Institute on the Conficker Controversy. Retrieved may 29, 2009, from Cyber secure institute Web site: http://cybersecureinstitute.org/blog/?p=15

F-secure (2009a) www.f-secure.com. Retrieved May 19, 2009, from www.f-secure.com/weblog/archives Web site: www.f-secure.com/weblog/archives/00001584.html

F-secure (2009b). www.f-secure.com. Retrieved Feb 12, 2009, from http://www.f-secure.com/v-descs/ Web site: http://www.f-secure.com/v-descs/worm_w32_downadup_al.shtml

Guidgion , Kevin (2009). www.mcafee.com. Retrieved Feb 22, 2009, from http://download.nai.com Web site: http://download.nai.com/products/mcafee-avert/documents/combating_w32_conficker_worm.pdf

Johansson, Jesper M (2005). Help: I Got Hacked. Now What Do I Do? Retrieved January 28, 2009, Web site: http://technet.microsoft.com/en-us/library/cc512587.aspx

Jordan, Stacy (2007 August 13). Mining gold... A primer on incident handling and response. Retrieved March 22, 2009, Web site: https://www.sans.org/reading_room/whitepapers/incident/mining_gold__a_primer_on_incident_handling_and_response_32818

McAfee (2008). www.mcafee.com Retrieved Jan 05, 2009, from http://vil.nai.com Web site: http://vil.nai.com/vil/content/v_153464.htm

McAfee (2009). www.mcafee.com. Retrieved March 19, 2009, from http://www.mcafee.com/ Web site: http://www.mcafee.com/us/threat_center/conficker.html

Microsoft TechNet (2009 February 06). Conficker worm. Retrieved April 02, 2009, from http://technet.microsoft.com Web site: http://technet.microsoft.com/en-us/security/dd452420.aspx

Nahorney, B (2009). www.symantec.com. Retrieved May 19, 2009, from http://www.symantec.com Web site:http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/the_downadup_codex_ed1.pdf

NIST, (2004). Computer Security Incident Handling Guide. *Special Publication 800-61*, Retrieved April 15 2009, from http://csrc.nist.gov/publications/nistpubs/800-61/sp800-61.pdf

US-CERT, (2005, May 16). Malware Threats and Mitigation Strategies. Retrieved Dec 12 2008, from http://www.us-cert.gov/reading_room/malware-threats-mitigation.pdf

Security labs (2007). *K.I.S.S Principle*. Retrieved from http://securitylabs.websense.com/content/Blogs/2818.aspx

So, Hee (2002 February 16). GIAC Intrusion Detection In Depth. Retrieved Feb 05, 2009, Web site: https://www.sans.org/reading_room/whitepapers/honors/intrusion_detection_in_depth_836

SRI International (2009 February 04). An Analysis of Conficker's Logic. Retrieved Feb 10, 2009, from http://mtc.sri.com/ Web site: http://mtc.sri.com/Conficker/

Symantec (2009). www.symantec.com. Retrieved Jan 11, 2009, from http://www.symantec.com Web site: http://www.symantec.com/security_response/writeup.jsp?docid=2008-112203-2408-99

Voorhees, James (2007 August 26). Distilling data in Sim. Retrieved Dec 11, 2009, from http://www.sans.org Web site: http://www.sans.org/reading_room/whitepapers/detection/distilling_data_in_a_sim_a_strategy_for_the_analysis_of_events_in_the_arcsight_esm_1916

Watson, David (2009 March 30). Know your enemy: Conficker worm. Retrieved Apr 02, 2009, from http://www.honeynet.org/papers/conficker/ Web site: http://www.honeynet.org/files/KYE-Conficker.pdf

Yason , M (2009, January 22). *Conficker*. Retrieved from http://www.iss.net/threats/conficker.html