SANS Institute
InfoSec Reading Room

Microsoft DNS Logs Parsing and Analysis: Establishing a Standard Toolset and Methodology for Incident Responders

Microsoft DNS request and response event logs are frequently ignored by incident responders within an investigation due to a historical reputation of being hard to parse and analyze. The fundamental importance of DNS to networking and the functioning of the Internet suggests this oversight could lead to a lack of crucial contextual information in an investigative timeline. This paper seeks to define a best practice for parsing, exporting and analyzing Microsoft DNS Debug and Analytical logs through the comparison of ex...

Microsoft DNS Logs Parsing and Analysis: Establishing a Standard Toolset and Methodology for Incident Responders

GIAC (GSEC) Gold Certification

Author: Shelly Giesbrecht, *headnerd@nerdiosity.com*
Advisor: *Tanya Baccam*

Accepted: October 2018

Abstract

Microsoft DNS request and response event logs are frequently ignored by incident responders within an investigation due to a historical reputation of being hard to parse and analyze. The fundamental importance of DNS to networking and the functioning of the Internet suggests this oversight could lead to a lack of crucial contextual information in an investigative timeline. This paper seeks to define a best practice for parsing, exporting and analyzing Microsoft DNS Debug and Analytical logs through the comparison of existing tool combinations to DNSplice, a purpose-built utility coded during the development of this paper. Findings suggest that DNSplice is superior to other toolsets tested where time to completion is a critical factor in the investigative process. Further research is required to determine if the findings are still valid on larger datasets or different analysis hardware.

1. Introduction

Incident responders execute their investigations accurately, efficiently, and in the timeliest a manner possible. Within the Digital Forensics and Incident Response (DFIR) industry, advanced tools such as Security Information and Event Management (SIEM) or log collection and correlation platforms (e.g., ELK) may not be available to organizations due to budget, resource or circumstance to facilitate a response. Similarly, when DFIR consultants are called to assist a client who is under duress due to a cyber-related attack or outage, they are likely to only have their go-bag with laptop and write-blockers to respond with (Giesbrecht, 2016). In these types of situations, responders are frequently called upon to innovate their tools or toolsets to accomplish the tasks required—a practice commonly referred to as "rolling your own." Once a script or tool initially created to solve a point- in- time problem proves useful, many responders choose to release their creation to the greater DFIR community on public code repositories like GitHub. Tools such as these exist for a wide variety of log sources and artifacts. Despite its contextual importance, one log type that is often neglected is Domain Name System (DNS) logs created by Microsoft's DNS Server service to record the query and response data generated by hosts within an organization's environment.

Given the foundational nature of DNS to the Internet, and the degree to which DNS requests and responses can add to an Incident Response (IR) investigation, the creation of a standard method for parsing and analysis of these logs is essential to establish. Without a SIEM or other advanced log correlation tooling, to what extent would the development of an individual tool to parse, analyze and provide statistics for Microsoft DNS logs within one interface increase the efficacy of an incident responder during an incident compared to using an amalgamation of currently available tools and scripts with a similar set of functions?

The testing methodology designed for use in this research paper utilized one or more Open Source or Commercial scripts, utilities or applications in the first three runs. DNSplice, a tool created and coded during the research phase of this paper with specific intent to parse, export and analyze DNS logs was used in the fourth run. The results of each run were compared to test the hypothesis that DNSplice was superior in both its

speed and accuracy in accomplishing all the testing criteria successfully. Understanding the results of this study ensured a definitive "best practice" was established for the parsing and analysis of Microsoft DNS logs without the assistance of a SIEM or a log collection and correlation platform. Security and Incident Response personnel without these more advanced toolsets now have a definitive option to ensure optimal efficacy when they desire to include Microsoft DNS log output into an investigative timeline.

2. Literature Review

DNS is a foundational protocol on which much of the Internet relies to function. Gonyea (2014) refers to DNS as a "phonebook for the Internet." In essence, it allows translation of a known domain name (e.g., google.com) into an assigned Internet Protocol (IP) address without knowledge of that IP address. As such, any organization allowing end users to browse the Internet has DNS, and therefore, should have DNS logs. For an Incident Responder, DNS logs are a treasure-trove of information about what domains or IPs users requested from computer systems on a network, and the responses to those requests.

Microsoft Windows Server operating systems have offered a DNS server service from Windows Server 2003 up to their latest offering of Windows Server 2016. However, DNS logging of interest to Incident Response (IR) is not turned on by default. From Windows Server 2003 to Windows 2008R2, DNS request and response logging was only available via the DNS Debug log, while a new DNS Analytical log was created to fulfill this role from the release of Windows 2012 onward (Microsoft, 2016). Additionally, while the new DNS Analytical logs are in a standard format with defined fields, DNS Debug logs are considered very difficult to parse due to non-standard spacing and lack of assigned columns or headers (Linley, 2014). Until recently and likely due to the non-standard format, DNS Debug logs were not included by many of the leading SIEM vendors in their supported log sources. As late as 2015, IBM QRadar did not have Microsoft DNS Debug logs listed in their Device Support Model (DSM) Configuration Guide (IBM, 2015). A possible reason for this lapse is that before the release of Windows 2012 with the new DNS Analytical log, enabling DNS Debug logs was not recommended

by Microsoft. The glaring exception to this advice being actual debugging activities as the process is considered performance intensive (Microsoft, 2016).

For IR practitioners, it is integral to have the ability to quickly and efficiently parse any log within the scope of their investigation. Many organizations do not have the budget or resources to deploy a SIEM, or other log collection and correlation platform, and therefore, Security or IR personnel are frequently called upon to create tools or find free or inexpensive tools to do the job. For organizations that have Microsoft DNS servers deployed in their environment and are without a modern SIEM, responders who choose to include DNS events in their incident timeline require a solution to parse one or both types of Microsoft DNS logs. Ideally, this solution would also be able to perform some additional analysis of the output including what domains or IPs are currently considered malicious by available online threat engines, such as VirusTotal or PassiveDNS, and provide statistics for context. A search for currently available tools that provide the ability to parse, in particular, DNS debug logs without requiring additional coding or configuration ended in one result: Win DNS Log Analyser. However, this tool supports only Windows 2008R2 and Windows 2003 and is limited to one log file at a time.

Additionally, there are no analysis features included beyond some basic statistics (Zedlan, 2009). As the tool provides no access to the source code, there is no avenue to add additional features or capabilities. Further searching found some PowerShell scripts available that parse the logs, but have little additional functionality (Technet, 2014) for reputation or statistics. Most of the script-based tools are intended for operational purposes such as finding DNS servers when demoting Active Directory Domain Controllers and may include statistics for the number of clients connecting to that server (Funk, 2015). The authors provide the scripts in raw format, granting flexibility to add additional modularity to them. However, for responders without established coding experience in PowerShell, where a GUI is desired or required, or where the actual parsing and analysis of the logs proc on a system not running on Windows, this would be a barrier. A review of popular programming languages for incident responders who seek to learn to code suggested that for multi-platform usability and ease of learning, Python is

considered a better choice (Giesbrecht, 2015). Lastly, a search conducted in a DFIR tools database on the website dfir.training for search terms, "domain name service" and "dns" did not produce any tools or scripts for the parsing or analysis of Microsoft DNS logs (dfir.training, 2017).

Since DNS provides functionality to many of the protocols that encompass Internet traffic including HTTP and SMTP (Hagen, 2017), the necessity for quick and accurate parsing and analysis of DNS logs to add context to an investigation is apparent. In the absence of an existing script or application to perform these functions, the development of such a tool can be considered to fill a current void in the IR community. Additionally, developing the tool in Python allows for multi-platform usage, as well as accessibility for additional features or capability when needed.

3. Methods

3.1. Lab Design

A virtual lab was built in VMWare Workstation Pro v12.1.1 build-3770994 containing a Windows domain (Wakanda.local) to create the logs required for testing purposes. The domain included a Windows 2003R2 DNS server (Zuri03), a Windows 2008R2 DNS server (Ayo08), a Windows 2012 DNS server (Okoye12) and a Windows 2016R2 DNS server (Ramonda16). The Windows 2016R2 server also acted as the Active Directory Domain Controller and Dynamic Host Configuration (DHCP) server. Additionally, the virtual lab contained three Windows workstations to generate DNS events from, and a pfSense virtual firewall to channel the traffic to the Internet.

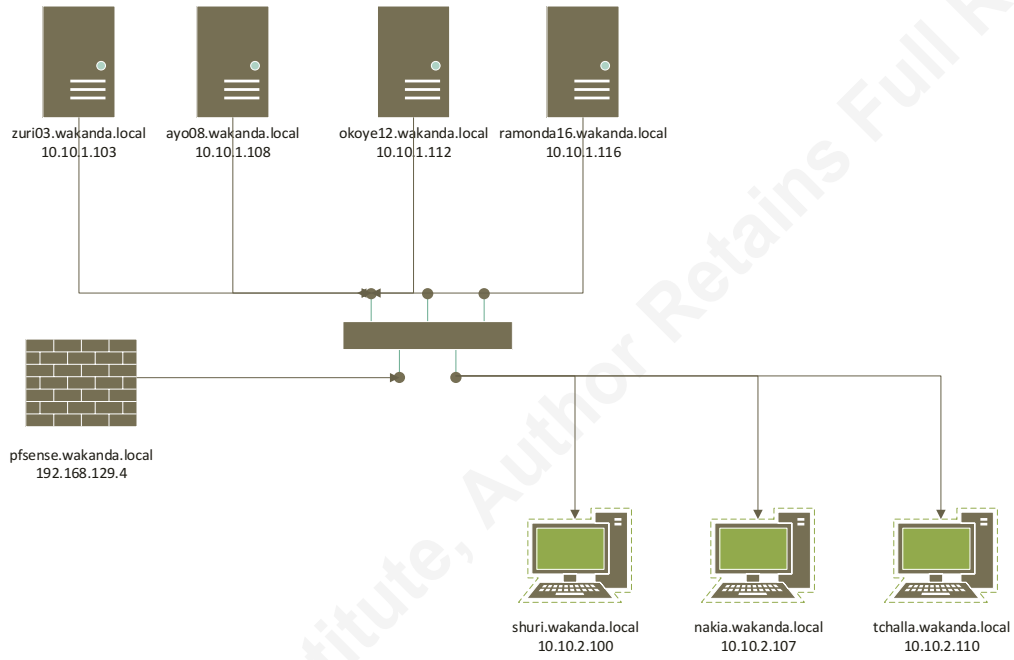

Figure 1. Wakanda.local Virtual Lab

Over the course of several days, DNS Debug and DNS Analytical logs were generated in the virtual lab. DNS Debug logs from Zuri03 and Ayo08 were collected by copying the logs from the directory in which they were created, onto a network share on Ramonda16. DNS Analytical logs were collected from Okoye12 and Ramonda16 by opening the Event Trace Log (ETL) format logs using native Windows binary tracerpt.exe and exporting to a comma-separated values (CSV) format using the denoted command below:

```
C:\Windows\system32>tracerpt -l C:\Windows\System32\winevt\Logs\Microsoft-Windows-DNSServer%4Analytical.etl -of CSV -o c:\<system_name>.csv
```

All logs were gathered to Ramonda16, then copied to the virtual host system and used in all test runs for parsing and analysis. An MD5 hash of each log file was taken upon copying of the logs from the virtual lab, and the files were hashed and compared to the original hashes before each test run began.

MD5	FileName
0f57aea6ac8a5e07fbb16a47705d6bf4	ayo08_dns.log
0e8145c4ccc6807aa705868b149a75eb	zuri03_dns.log
a9a87ca5b1c92aceaf7a90f1f6f11ab7	okoye12_dns.csv
8a111d762ba7cfa87f024f30c23fa922	ramonda16_dns.csv

SG

Table 1. MD5 Hash of collected DNS Logs

The virtual lab was hosted, and all testing was completed on the same analysis system with the following specifications:

- Windows 10 Pro
- 64GB RAM Memory
- SSD Hard Drive
- Microsoft Office 2016
- PowerShell v5.0

Additional tools used in the testing were as follows:

- Win DNS Log Analyser[1]
- Read-dns-debug-logs-Get-DNSDebugLog.ps1[2]
- Microsoft Excel 2016 MSO (16.0.9330.2073) 32-Bit
- VT_Domain_Scanner[3]
- DNSplice v1.0[4]

3.2. Test Design

3.2.1. Test Objectives

As described in the introduction, the primary, notable issue with DNS Debug logs is the non-standard format that makes corroboration with other more standard logs difficult. In designing the tests for this research, the primary objectives were to parse all collected DNS logs, if possible, into defined columns (DateTime, Client IP, URI Query, Domain) and then output the parsed data to a comma-separated values (CSV) file. With the successful completion of the first two objectives, secondary objectives were designed

[1] http://www.zedlan.com/win_dns_log_analyser.php

[2] https://gallery.technet.microsoft.com/scriptcenter/Read-DNS-debug-log-and-532a8504

[3] https://github.com/clairmont32/VT-Domain-Scanner/blob/master/VT_Domain_Scanner.py

[4] https://github.com/nerdiosity/DNSplice

to bring more context to the results. The first of these secondary objectives was to produce statistics from the DNS logs including the following:

- Client IPs with the most requests
- Domains requested with the least frequency
- Domains requested with the most frequency

Last of the secondary objectives was to provide web reputation information on the first ten domains parsed from each log file by performing a VirusTotal Domain Report via the VirusTotal Public API[5].

3.2.2. Test Criteria

The criteria used for each test run is described below in Table 2: Testing Criteria.

Criteria		Possible Results	Level of Completion	Description
Parsing	**Completion**	Time	Complete, Somewhat, Not at All	Overall time and completeness for parsing the files into standard fields was recorded
	Accuracy	Percentage	Complete, Somewhat, Not at All	Accuracy and completeness for parsing the files into standard fields was recorded
Output to CSV	**Completion**	Time	Complete, Somewhat, Not at All	Overall time and completeness for exporting the data to CSV was recorded
	Accuracy	Percentage	Complete, Somewhat, Not at All	Accuracy and completeness for exporting the data to CSV was recorded
Statistics	**Completion**	Time	Complete, Somewhat, Not at All	Overall time and completeness for the creation of the desired statistics was recorded
	Accuracy	Percentage	Complete, Somewhat, Not at All	Accuracy and completeness for the creation of the desired statistics was recorded
	Completion	Time	Complete, Somewhat,	Overall time and completeness for web reputation lookup of the

[5] https://developers.virustotal.com/v2.0/reference

			Not at All	first ten domains parsed was recorded
Domain Reputation Lookup	**Accuracy**	Percentage	Complete, Somewhat, Not at All	Accuracy and completeness for web reputation lookup of the first ten domains parsed was recorded
Overall Speed		Time	Success, Failure	Time for all steps of each test run for each log was added together, and then the total time of each log was added to give an overall time for each test run.

Table 2. Testing Criteria

For each test run, each step was recorded for completion, accuracy, and time. Failure to complete any portion of the testing resulted in a 'Failure' rating for the test run for a particular log and toolset.

Time tracking for each step of test runs was taken using the stopwatch app on an iPhone 10 with iOS v11.3.1 (15E302) and also on an iPad Mini 2 with iOS v11. The two times were averaged for each step and then all averaged times were totaled and recorded for each test run.

All data was recorded in the Test Run Datasheet (Appendix A).

3.3. Test Runs

For all test runs, logs from Okoye12 and Ramonda16 were already in CSV format due to the method of extraction from the lab servers.

3.3.1. Test Run 1

For Test Run 1, the tools that compromised the testing suite included Win DNS Log Analyser, Microsoft Excel 2016 and VT_Domain_Scanner. Win DNS Log Analyser is advertised as a "free utility that will read and analyse your Windows Server (2000, 2003, 2008) DNS Logs" and also that "using this tool can help you track down problems on your network, even if those problems are silent, and help you optimise your home or corporate network" (Zedlan, 2009). In other words, the intention is to solve operational and not cybersecurity related issues. In designing this test run, an assumption was made regarding the ability of Win DNS Log Analyser to ingest, parse into columns, and export DNS Debug logs from Windows 2003 and 2008R2 servers to CSV. Microsoft Excel was to be used for:

- Filtering logs to limit records to only DNS requests
- Formatting to remove all but the required columns (DateTime, Client IP, URI Query, Domain)
- Creating statistics based on the list of domains

Lastly, VT_Domain_Scanner was to run using Python v2.7 to perform lookups on the reputation of each domain and store the answer in an additional CSV file.

The steps for completing Test Run 1 were designed as follows:

a. Ingest Windows 2003/2008R2 logs into Win DNS Log Analyser
   - Open Win DNS Log Analyser, click Edit > Settings and browse to the location of the first log file and click Ok.
   - Press F5 to start the analysis
b. Filter the CSV for Windows 2012/2016R2 logs to event ID 256[6] (DNS requests) only
   - Open the Windows 2012/2016R2 log CSV files in Microsoft Excel and using the Data > Filter option, select only event ID 256 from the Event ID column
c. Export Windows 2003/2008R2 logs from Win DNS Log Analyser to CSV
   - An assumption was made that this functionality was available
d. Filter Windows 2003/2008R2 for local DNS requests only
   - Open the Windows 2003/2008R2 log CSV files in Microsoft Excel and using the Data > Filter option, select only entries where the requesting IP is a non-routable address[7] and the Query/Response type is Rcv
e. Format data in the CSVs to only the following columns:
   - TimeStamp (DATETIME) – format YYYY-MM-DD HH:MM:SS

[6] https://github.com/nsacyber/Event-Forwarding-Guidance/tree/master/Events

[7] https://tools.ietf.org/html/rfc1918

1. For Windows 2003, the original format was YYYYMMDD
2. For Windows 2008R2, the original format was MM/DD/YYYY
3. For Windows 2012/2016R2, the original format was Microsoft Epoch (e.g., 131715077489008000)
    a. Conversion was done in Excel using formula: *=IF(C2>0, C2/(8.64*10^11) - 109205,"")*

- Requesting IP (CLIENT IP) – in IPv4 format
- URL Requested (URI QUERY) – full requested string including subdomains (e.g., www.google.com)
- Base Domain (DOMAIN) – domain only (e.g., google.com)
    1. The base domain was parsed from the full URL by splitting the URL at the '.', and concatenating the domain name (e.g.: google), a ".", and the top-level domain (TLD) suffix (e.g.: com)

f. Create statistics from the list of domains in each CSV using pivot table functionality in Excel:
    - Top ten clients with the most domain requests
    - Top ten most-requested domains
    - Top ten least-requested domains
g. Create domains.txt from the first ten unique domains in each CSV. This was done by hand-selecting the first ten unique domains.
h. Perform a domain reputation lookup for each CSV's domains.txt file with VT_Domain_Scanner.py. In a command console, the following command was run to execute the reputation lookup from the directory containing the VT_Domain_Scanner.py script:

    python VT_Domain_Scanner.py >> results.txt

    - Note: Due to an error encountered during the first run of the script where JSON output processing occurs, the script was changed to comment out any additional processing beyond the

reputation lookup functions. A line was added to print the domain and JSON response from VirusTotal to the console, which was then directed to *results.txt* via the command run. The original script link and modified script in full can be found in Appendix C and D, respectively.

3.3.2. Test Run 2

For Test Run 2, the tools that compromised the testing suite included Read-dns-debug-logs-Get-DNSDebugLog.ps1, Microsoft Excel 2016 and VT_Domain_Scanner. Read-dns-debug-logs-Get-DNSDebugLog.ps1is listed on Microsoft Technet's Script Center and is described as giving the ability to "(r)ead DNS debug log and generate output in readble CSV format" and "identify dependencies on the DNS server" (Microsoft Corporation, 2014). Similar to Win DNS Log Analyser in Test Run 1, the intention is to solve operational and not cybersecurity- related issues. In designing this test run, an assumption was made regarding the ability of this PowerShell script to ingest, parse into standard columns, and export DNS Debug logs from Windows 2003 and 2008R2 servers to CSV. As in Test Run 1, Microsoft Excel was to be used for filtering, additional formatting, and creating statistics. VT_Domain_Scanner was to be run again using Python v2.7 to perform lookups on the reputation of each domain and store the answer in an additional CSV file.

The steps for completing Test Run 2 were designed as follows. Most of the steps described without details were designed as identical to those in Test Run 1:

a. Parse and export Windows 2003/2008R2 logs to CSV with Read-dns-debug-logs-Get-DNSDebugLog.ps1. An example of the command run to execute the PowerShell script is:
   Get-DNSDebugLog -DNSLog "E:\DNSplice\Logs\ayo08_dns.log" | Export-CSV "E:\DNSplice\Research\Tests\Test 2\Test2-export-2008R2.csv"
b. Filter the CSV for Windows 2012/2016R2 logs to event ID 256 (DNS requests) only
c. Filter Windows 2003/2008R2 for local DNS requests only

d. Format the data in the CSVs to only the following columns:
    - TimeStamp (DATETIME) – format YYYY-MM-DD HH:MM:SS
    - Requesting IP (CLIENT IP) – in IPv4 format
    - URL Requested (URI QUERY) – full requested string including subdomains (e.g., www.google.com)
    - Base Domain (DOMAIN) – domain only (e.g., google.com)

i. Create statistics from the list of domains in each CSV using pivot table functionality in Excel:
    - Top ten clients with the most domain requests
    - Top ten most-requested domains
    - Top ten least-requested domains

j. Create domains.txt from the first ten unique domains in each CSV

k. Perform a domain reputation lookup for each CSV's domains.txt file with VT_Domain_Scanner.py

3.3.3. Test Run 3

For Test Run 3, the tools that compromised the testing suite included Microsoft Excel 2016 and VT_Domain_Scanner. Microsoft Excel was to be used for ingestion, parsing, and filtering of all logs, 2003 to 2016R2, as well as for additional formatting to ensure only the required columns (DateTime, Client IP, URI Query, Domain) were used, and to create statistics based on the list of domains. As in previous test runs, VT_Domain_Scanner was to run using Python v2.7 to perform lookups on the reputation of each domain and store the answer in an additional CSV file.

The steps for completing Test Run 3 were designed as follows. Most of the steps described without details were designed as identical to those in Test Run 1:

a. All versions of Windows DNS logs were ingested into and parsed by Microsoft Excel. Logs were parsed by opening Excel and importing the txt or CSV files via the Data > Get External Data > From Text functionality. Fields were separated via either comma, tab or space delimiters depending on the original data. Content that was not required outside of the specified, previously stated fields was deleted, and the file was saved as a new CSV.

b. Filter Windows 2003/2008R2 for local DNS requests only
c. Filter CSV for Windows 2012/2016R2 logs to event ID 256 (DNS requests) only
d. Format data for all logs in CSV format to only the following columns:
    - TimeStamp (DATETIME) – format YYYYMMDD HH:MM:SS
    - Requesting IP (CLIENT IP) – in IPv4 format
    - URL Requested (URI QUERY) – full requested string including subdomains (e.g., www.domain.com)
    - Base Domain (DOMAIN) – domain only (e.g., domain.com)
e. Create statistics from the list of domains in each CSV
    - Top ten clients with the most domain requests
    - Top ten most-requested domains
    - Top ten least-requested domains
f. Create domains.txt from the first ten unique domains in each CSV
g. Perform a domain reputation lookup for each CSV's domains.txt file with VT_Domain_Scanner.py

3.3.4. Test Run 4

The steps for completing Test Run 4 were designed as follows:

Note: DNSplice v1.0 was created by the author specifically for this research paper to perform all steps in Test Run 4.

a. Execute DNSplice v1.0 from a command console using the following command to include VirusTotal domain reputation lookup:
   python dnsplice_v1.py –i <filename> -v <vt_api_key>
   where:
   -i indicates the DNS log file name and location
   -v indicates the VirusTotal Public API key required for lookup

```
E:\DNSplice\Code>dnsplice_v1.py -i ..\Logs\okoye12_dns.csv -v 31226
```

Figure 2. DNSplice v1.0 command for full execution including VirusTotal lookup

Execution of the previously noted command string in Python v2.7 ensures the following occurs:

a. Ingest Windows 2003/2008R2/2012/2016R2 logs in memory
b. Filter Windows 2012/2016R2 logs to event ID 256 (DNS requests) only
c. Filter Windows 2003/2008R2 for local DNS requests only
d. Format data for all logs to only the following columns:
    - TimeStamp (DATETIME) – format YYYYMMDD HH:MM:SS
    - Requesting IP (CLIENT IP) – in IPv4 format
    - URL Requested (URI QUERY) – full requested string including subdomains (e.g., www.domain.com)
    - Base Domain (DOMAIN) – domain only (e.g., domain.com)
e. Export Windows 2003/2008R2/2012/2016R2 logs to CSV

```
DateTime,ClientIP,URIQuery,Domain

2018-05-22 20:29:53,10.10.1.103,_ldap._tcp.pdc._msdcs.wakanda.local,wakanda.local

2018-05-22 20:29:53,10.10.1.116,_ldap._tcp.pdc._msdcs.wakanda.local,wakanda.local

2018-05-22 20:30:22,10.10.1.116,wakanda.local,wakanda.local

2018-05-22 20:31:22,10.10.1.116,1.10.10.in-addr.arpa,in-addr.arpa

2018-05-22 20:34:26,10.10.1.117,fe2.update.microsoft.com,microsoft.com

2018-05-22 20:34:26,10.10.1.116,fe2.update.microsoft.com,microsoft.com

2018-05-22 20:34:26,10.10.1.117,fe3.delivery.mp.microsoft.com,microsoft.com

2018-05-22 20:34:26,10.10.1.116,fe3.delivery.mp.microsoft.com,microsoft.com

2018-05-22 20:34:27,10.10.1.117,v10.vortex-win.data.microsoft.com,microsoft.com
```

Figure 3. Output DNS logs using DNSplice v1.0 to CSV format

f. Create statistics from the list of domains parsed in step d and output to the console:
    i. Top ten clients with the most domain requests
    ii. Top ten most-requested domains
    iii. Top ten least-requested domains

SG

```
############################################################
DNSplice Statistics
############################################################
# The top 10 requesting client IPs are:
('10.10.1.116', 163), ('127.0.0.1', 118), ('10.10.1.103', 61),
------------------------------------------------------------
#  top 10 requested domains are:
('google.com', 26), ('go.com', 16), ('cloudfront.net', 12), ('
, 9), ('footprintdns.com', 8), ('microsoft.com', 8), ('live.co
------------------------------------------------------------
# The top 10 least domains are:
('dynect.net', 1), ('cedexis.net', 1), ('indexww.com', 2), ('o
cd.com', 2), ('appspot.com', 2), ('viawest.net', 2), ('adsrvr.
############################################################
```

Figure 4. Statistics created from unique domains parsed from DNS logs via DNSplice v1.0

g. Perform a domain reputation lookup for the first ten unique domains in each CSV and output to a separate CSV file

```
Domain,Domain Report
adform.net,"{u'BitDefender category': u'business', u'domain_siblings': [], u'undetected_urls': [[u'h
moatads.com,"{u'domain_siblings': [], u'undetected_urls': [[u'https://moatads.com/files/theme/addons
onion.to,"{u'BitDefender category': u'business', u'domain_siblings': [], u'undetected_urls': [], u'u
mediavoice.com,"{u'BitDefender category': u'business', u'domain_siblings': [], u'undetected_urls': [
cloudfront.net,"{u'BitDefender category': u'hosting', u'domain_siblings': [], u'undetected_urls': [[
turn.com,"{u'detected_downloaded_samples': [], u'undetected_downloaded_samples': [{u'date': u'2017-0
atdmt.com,"{u'detected_downloaded_samples': [], u'undetected_downloaded_samples': [{u'date': u'2013-
dotomi.com,"{u'BitDefender category': u'business', u'domain_siblings': [], u'undetected_urls': [[u'h
huffingtonpost.com,"{u'detected_downloaded_samples': [], u'undetected_referrer_samples': [{u'date':
```

Figure 5. VirusTotal Domain Reputation report output from DNSplice v1.0

The completion of Test Run 4 concluded the testing phase of the research, and the analysis of the results was begun.

4. Findings and Discussion

4.1. Testing Results

Of the four test runs, only three were completed successfully. While Win DNS Log Analyser was able to ingest and parse DNS Debug logs from Windows 2003 and

2008R2 during Test Run 1, there was no capability to export the parsed logs to CSV or any other format. Additionally, while statistics for top ten client IPs making requests and the top ten most-requested domains were available in Win DNS Log Analyser, the top ten least-requested domains were not available. As such, Test Run 1 was not complete and was assigned a 'Failure' rating.

As shown in Table 3, of the test runs that were completed with a 'Success' rating, Test Run 4 was completed approximately five times faster than either Test Run 2 or 3. The results of all steps of each test run are included in Appendix A.

Criteria		Test Run #1	Test Run #2	Test Run #3	Test Run #4
Parsing	**Completion**	Complete	Complete	Complete	Complete
	Accuracy	Complete	Complete	Complete	Complete
Output to CSV	**Completion**	Somewhat	Complete	Complete	Complete
	Accuracy	Somewhat	Complete	Complete	Complete
Statistics	**Completion**	Somewhat	Complete	Complete	Complete
	Accuracy	Somewhat	Complete	Complete	Complete
Domain Reputation Lookup	**Completion**	Somewhat	Complete	Complete	Complete
Overall Speed		Failure	1:09:46.38	1:25:00.05	14:14.90

Table 3. Overall Results for all Test Runs

4.2. Discussion

Overall, the results of the testing indicate that a purpose-built script to perform all the steps set out in the design section (parsing, exporting to CSV, statistics, and web reputation lookup) is, by a large margin, faster than using disparate, non-specific tools to accomplish the same results. As previously established, an incident responder must conduct their investigation in an expedient, but accurate fashion. The difference in overall time to completion between Test Run 4 with DNSplice and Test Run 2 and 3 is

significant enough to suggest that where timeliness is desired or required, DNSplice or a similarly-coded utility would be beneficial.

However, aside from Test Run 1, where limitations of the tool used for parsing Windows 2003/2008R2 logs resulted in a 'Failure' rating, there were no differences in overall completion or accuracy of the successful runs. This suggests that the methods used in Test Run 2 and 3 could be used where preferred if time was not as much of a factor, or to validate the findings of a purpose-built script like DNSplice. Validation of forensic tools and their results is commonly-held to be a compulsory part of the investigative process as tools that are found to be untrustworthy result in untrustworthy findings as well (Dimpe & Kogeda, 2017).

During testing, several limitations to the current design were noted. This included a discernible opportunity for the researcher to "learn" over the course of the test runs. Where steps were the same in successive runs following Test Run 1, knowledge of how to accomplish the steps likely increased the speed at which those tasks were completed. Additionally, the log size, and therefore the number of distinct domains in the logs, was small compared to logs that would be found in a large enterprise environments. It is likely that larger log files would increase the overall times for all successful test runs.

Tools used for Test Runs 1 to 3 are limited to use on Windows platforms (Win DNS Log Analyser, and Read-dns-debug-logs-Get-DNSDebugLog.ps1) which restrict the analysis system to Windows-only if the research was repeated.

Lastly, the hardware configuration of the analysis system contributed to the overall speed of completion for the test runs. These speeds would also likely change depending on the hardware platform used for testing.

5. Recommendations and Implications

Future recommendations for this research would be to recreate the experiment using different variables including lesser or more robust hardware on the analysis system, or larger size log files. Additionally, it could be beneficial to have other subjects conduct the testing using the original log files, tools, and a similarly-configured analysis system.

SG

These results would assist in determining if the notable difference in time between the results of Test Run 4 and Test Run 2 and 3 persists. If the data were to hold valid, this would be a more significant indication that a best practice for the parsing and analysis of Microsoft DNS logs had been established.

A number of feature enhancements are suggested to improve the overall functionality of the DNSplice tool to further its standing as a de facto tool for DNS log analysis. First, the code should be ported to Python 3.0 from its current platform on Python 2.7, which is close to its end of life with regards to development and support. This allows more avenues of opportunity in function and growth including a more robust statistics module that does not currently exist in Python 2.7. Additionally, DNSplice can be improved by adding a more standard code format including established functions and error handling, as well as including capabilities to ingest multiple files at once, export to other formats such as XML, and remove, where applicable, internal or popular domains from the parsing to increase speed or focus.

6. Conclusion

When called to respond to a cyber incident, responders are charged with assisting their customer, whether internal or external, in determining, if possible, the 'what', the 'how', and the 'when' of an attack or breach. To do this, they must possess not only the skill and know-how to uncover these facts, but also the toolset to assist them in collecting and analyzing the relevant data. When frequently faced with a dearth of more automated tooling found in log correlation platforms or SIEMs, or if the log format in question is challenging to parse, responders must rely on their ability to be creative in the moment, or on established best practices for specific artifacts types. The latter can be seen in tools like Volatility, an open-source Python framework built to parse and analyze memory images extracted from Windows, Mac OS or Linux systems, and is considered the standard for memory-based forensic analysis (The Volatility Foundation, 2018). Similarly, when faced with the non-standard format of Microsoft DNS Debug logs, having a best practice in place for efficient and accurate ingestion, parsing and exporting

of this vital data source to facilitate a responder's investigation and further analysis seems a simple next step.

DNS will continue to be a fundamental part of the function of much of the traffic traversing the Internet for many years to come, and it is highly likely that there will continue to be large numbers of Microsoft DNS servers deployed to provide this service to end-users. Given the criticality of this service, and therefore the importance of the context it can add to an investigation, it is crucial that a standard is established for its analysis.

The findings of the research conducted in this paper suggest a compelling likelihood that DNSplice is better suited to handling the tasks set out within the testing design, given the parameters of most incident response engagements to find answers, where possible, in the most expeditious but precise manner.

SG

References

Dimpe, Precilla & Kogeda, Okuthe. (2017). Impact of Using Unreliable Digital Forensic Tools. Retrieved June 4, 2018 from https://www.researchgate.net/publication/320922374_Impact_of_Using_Unreliable_Digital_Forensic_Tools

DFIR Training (2018) DFIR Tools. Retrieved from https://www.dfir.training/index.php/tools/search?searchword=domain+name+service&search_cat=1

Funk, J. (2015, August 3). Parse Windows DNS Debug logs for Client IP and hitcount. Retrieved from http://www.anexinet.com/blog/parse-windows-dns-debug-logs-for-client-ip-and-hitcount/

Giesbrecht, S. (2016). What's in Your Incident Response Go-Bag? Retrieved June 6, 2018 from Cisco Security Blog: https://blogs.cisco.com/security/whats-in-your-incident-response-go-bag

Giesbrecht, S. (2015, July 22). Coding For Incident Response: Solving the Language Dilemma. Retrieved January 8, 2018 from SANS Reading Room: https://www.sans.org/reading-room/whitepapers/incident/coding-incident-response-solving-language-dilemma-36107

Gonyea, C. (2010, August). DNS: Why It's Important & How It Works. Retrieved December 06, 2016, from http://dyn.com/blog/dns-why-its-important-how-it-works/

Hagen, P. (2017, February 21). Passive DNS Monitoring – Why It's Important

SG

for Your IR Team. Retrieved from https://www.redcanary.com/blog/passive-dns-monitoring-your-ir-team-needs-it/

IBM Corporation (2015). DSM Configuration Guide v7.1.x and 7.2.x. Retrieved from ftp://ftp.software.ibm.com/software/.../qradar/.../DSMConfigurationGuide-71MR1.pdf

Linley, N. (2014, March 12). Parsing DNS Debug Logs. Retrieved from http://myitpath.blogspot.com/2014/03/parsing-dns-debug-logs-microsoft.html)

Microsoft Corporation (2016, August 31). DNS Logging and Diagnostics. Retrieved from https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/dn800669(v=ws.11)

Microsoft Corporation (2016). Windows Server 2003/2003 R2. Retrieved January 9, 2018 from https://www.microsoft.com/en-US/download/details.aspx?id=53314

Microsoft Corporation (2014, October 28). Read DNS debug log and generate output in readable CSV format. Retrieved from https://gallery.technet.microsoft.com/scriptcenter/Read-DNS-debug-log-and-532a8504

Zedlan (2009). Win DNS Log Analyser (application). Retrieved from http://www.zedlan.com/win_dns_log_analyser.php

Appendix A: Test Run Data

Criteria		Test Run #1	Test Run #2	Test Run #3	Test Run #4
Parsing	Completion	Complete	Complete	Complete	Complete
	Accuracy	Complete	Complete	Complete	Complete
Output to CSV	Completion	Somewhat	Complete	Complete	Complete
	Accuracy	Somewhat	Complete	Complete	Complete
Statistics	Completion	Somewhat	Complete	Complete	Complete
	Accuracy	Somewhat	Complete	Complete	Complete
Domain Reputation Lookup	Completion	Somewhat	Complete	Complete	Complete
Overall Speed		Failure	1:09:46.38	1:25:00.05	14:14.90

Table A-1. Overall Test Runs Results

	Test Run #1	2003		2008R2		2012		2016R2	
	Criteria	Results	Level of Completion	Results	Level of Completion	Results	Level of Completion	Results	Level of Completion
Parsing	Completion	0:50.59/0:50.87	Complete	0:45.30/0:45.35	Complete	4:57.48/4:59.31	Complete	3:43.90/3:45.90	Complete
	Accuracy	100%	Complete	100%	Complete	100%	Complete	100%	Complete
Output to CSV	Completion	Function not available	Not at all	Function not available	Not at all	Already Complete	Complete	Already Complete	Complete
	Accuracy	Function not available	Not at all	Function not available	Not at all	100%	Complete	100%	Complete
Statistics	Completion	Top Clients Top Domains	Somewhat	Top Clients Top Domains	Somewhat	2:07.41/2:09.23	Complete	1:47.61/1:47.86	Complete
	Accuracy	67%	Somewhat	67%	Somewhat	100%	Complete	100%	Complete
Domain Reputation Lookup	Completion	Unable to complete	Not at all	Unable to complete	Not at all	10:13.75/10:13.75	Complete	11:08.28/11:08.53	Complete
Overall Speed		n/a	Failure	n/a	Failure	17:18.64/17:22.29 Avg: 17:20.47	Success	16:39.79/16:42.29 Avg: 16:41.04	Success

Table A-2. Results from Test Run 1

Tools

Parsing: WinDNSLog Analyzer, Excel
Output to CSV: WinDNSLog Analyzer, Excel
Statistics: Excel
Domain Reputation Lookup: VT Domain Scanner

SG

	Test Run #2	2003		2008R2		2012		2016R2	
	Criteria	Results	Level of Completion	Results	Level of Completion	Results	Level of Completion	Results	Level of Completion
Parsing	Completion	Together with Output	Complete	Together with Output	Complete	4:57.48/4:59.31	Complete	3:43.90/3:45.90	Complete
	Accuracy	100%	Complete	100%	Complete	100%	Complete	100%	Complete
Output to CSV	Completion	12:18.30/12:19.75	Complete	9:17.15/9:18.23	Complete	Already Complete	Complete	Already Complete	Complete
	Accuracy	100%	Complete	100%	Complete	100%	Complete	100%	Complete
Statistics	Completion	3:38.84/3:38.95	Complete	2:31.41/2:42.76	Complete	2:07.41/2:09.23	Complete	1:47.61/1:47.86	Complete
	Accuracy	100%	Complete	100%	Complete	100%	Complete	100%	Complete
Domain Reputation Lookup	Completion	4:34.30/4:35.12	Complete	3:17.06/3:17.86	Complete	10:13.75/10:13.75	Complete	11:08.28/11:08.53	Complete
Overall Speed		20:31.4/ 20:33.082 Avg: 20:32.63	Success	15:05.62/ 15:18.085 Avg: 15:12.24	Success	17:18.64/17:22.29 Avg: 17:20.47	Success	16:39.79/16:42.29 Avg: 16:41.04	Success

Table A-3. Results from Test Run 2

Tools

Parsing: Read-dns-debug-logs-Get-DNSDebugLog.ps1, Excel
Output to CSV: Read-dns-debug-logs-Get-DNSDebugLog.ps1, Excel
Statistics: Excel
Domain Reputation Lookup: VT Domain Scanner

	Test Run #3	2003		2008R2		2012		2016R2	
	Criteria	Results	Level of Completion	Results	Level of Completion	Results	Level of Completion	Results	Level of Completion
Parsing	Completion	Together with Output	Complete	Together with Output	Complete	4:57.48/4:59.31	Complete	3:43.90/3:45.90	Complete
	Accuracy	100%	Complete	100%	Complete	100%	Complete	100%	Complete
Output to CSV	Completion	15:43.95/15.44.15	Complete	9:33.16/9:34.02	Complete	Already Complete	Complete	Already Complete	Complete
	Accuracy	100%	Complete	100%	Complete	100%	Complete	100%	Complete
Statistics	Completion	1:28.23/1.30.10	Complete	1:31.37/1:31.71	Complete	2:07.41/2:09.23	Complete	1:47.61/1:47.86	Complete
	Accuracy	Complete	Complete	Complete	Complete	100%	Complete	100%	Complete
Domain Reputation Lookup	Completion	12:59.51/13:01.10	Complete	9.38.68/9:40.56	Complete	10:13.75/10:13.75	Complete	11:08.28/11:08.53	Complete
Overall Speed		30:11.69/30:15.35 Avg: 30:13.54	Success	20:43.21/ 20:46.29 Avg: 20:45.00	Success+-	17:18.64/17:22.29 Avg: 17:20.47	Success	16:39.79/16:42.29 Avg: 16:41.04	Success

Table A-4. Results from Test Run 3

Tools

Parsing: Excel
Output to CSV: Excel
Statistics: Excel
Domain Reputation Lookup: VT Domain Scanner

SG

	Test Run #4	2003	2008R2	2012	2016R2
	Criteria	Level of Completion	Level of Completion	Level of Completion	Level of Completion
Parsing	Completion	Complete	Complete	Complete	Complete
	Accuracy	100%	100%	100%	100%
Output to CSV	Completion	Complete	Complete	Complete	Complete
	Accuracy	100%	100%	100%	100%
Statistics	Completion	Complete	Complete	Complete	Complete
	Accuracy	100%	100%	100%	100%
Domain Reputation Lookup	Completion	Complete	Complete	Complete	Complete
Overall Speed		3:44.27	3:10.08	3:18.61	4:01.94

Table A-5. Results from Test Run 4

Tools

Parsing: DNSplice v1
Output to CSV: DNSplice v1
Statistics: DNSplice v1
Domain Reputation Lookup: DNSplice v1

SG

Appendix B: DNSplice FAQ and How-To

DNSplice v1.0 can be downloaded from https://github.com/nerdiosity/DNSplice

```
DNSplice was created by Shelly Giesbrecht (nerdiosity) to assist
incident responders to quickly and easily parse client query
events from ugly DNS logs for Microsoft Windows 2003/2008R2 (DNS
debug log) to Windows 2012R2/2016 (DNS Analytical log) into a
format (CSV) suitable for additional analysis or insertion into a
larger timeline.
version: DNSplice v1.0
date of release: June 8, 2018

This project was created in answer to a problem encountered by me
over years of doing IR, and as a way of learning to code.
Comments or suggestions are greatly appreciated.
email: info@nerdiosity.com twitter: @nerdiosity
github: https://github.com/nerdiosity/DNSplice
################################################################

Requirements:
DNSplice uses the requests module for python. This will need to
be installed to run.
command: pip install requests

To run:
At command prompt: python dnsplice_v1.py -i <inputfile> -v
<virustotal apikey> -t <threatgrid apikey>

Options:
-i, --input : DNS log filename (REQUIRED)
-v, --vtkey : VirusTotal API key (OPTIONAL)
-t, --tgkey : Cisco ThreatGrid API key (OPTIONAL)

Output:
DNS logs are parsed to include timedatestamp, client IP, uri
requested, and domain, and are outputted automagically to
output.csv
in the directory DNSplice is run from.
VirusTotal Domain Report lookups are performed every 20 seconds
(3/min) and are outputted to vt_output.csv. For large files, this
may take some time.
```

SG

```
ThreatGrid Lookups are limited to 50 lookups per day. Top ten
most and least requested domains are requested from ThreatGrid
and
are outputted to tg_output.csv. Requests are made every 20
seconds.
```

SG

Appendix C: Links to Additional Tools

1. Win DNS Log Analyser http://www.zedlan.com/win_dns_log_analyser.php
2. Read-dns-debug-logs-Get-DNSDebugLog.ps1 https://gallery.technet.microsoft.com/scriptcenter/Read-DNS-debug-log-and-532a8504
3. VT Domain Scanner https://github.com/clairmont32/VT-Domain-Scanner/blob/master/VT_Domain_Scanner.py

Appendix D: VT Domain Scanner – Script Versions

Original Version link is referenced in Appendix C.

Modified Version (bolded where changed):

```
__author__ = 'Matthew Clairmont'

__email__ = 'matthew.clairmont1@gmail.com'

__version__ = '1.0'

__date__ = '07/01/2015'

#Designed for Python 2.7, this is the initial creation of this script with no error checking. Any errors may be related to incorrect API key or bad domain formatting in input file. Updates will be provided in intervals over the next few months.

#VT Domain Scanner takes a file of domains, submits them to the Virus Total domain scanning API and outputs the domain and AV hits to a text file.


import urllib

import urllib2

import time

import json as simplejson


#submits domain to VT to generate a fresh report for DomainReportReader()

def DomainScanner(domain):

    url = 'https://www.virustotal.com/vtapi/v2/url/scan'
```

SG

```
        parameters = {'url': domain,
                      'apikey':
'31226d3572b202db21fcb8859b3b5fbd406f3dc791dbd623f5848da33846599c
'}

        #URL encoding and submission
        data = urllib.urlencode(parameters)
        req = urllib2.Request(url, data)
        response = urllib2.urlopen(req)
        print('Domain scanned successfully ')

        #for URL scan report debugging only
        #print(response)

    def DomainReportReader(domain):
        #sleep 15 to control requests/min to API. Public APIs
only allow for 4/min threshold, you WILL get a warning email to
the owner of the account if you exceed this limit. Private API
allows for tiered levels of queries/second.
        time.sleep(15)

        #this is the VT url scan api link
        url = 'https://www.virustotal.com/vtapi/v2/url/report'
```

SG

```
        #API parameters
        parameters = {'resource': domain,
                      'apikey': ''} #API Key Removed for
publishing

        #URL encoding and submission
        data = urllib.urlencode(parameters)
        req = urllib2.Request(url, data)
        response = urllib2.urlopen(req)
        json = response.read()
        #Script change made by Shelly Giesbrecht to ensure the
basic lookup would report back due to issues encountered with the
JSON processing code below
        print domain, ",", json

        #"""response harvesting in json dictionary form. See
sample response --> {'permalink':
'https://www.virustotal.com/url/dd014af5ed6b38d9130e3f466f850e46d
21b951199d53a18ef29ee9341614eaf/analysis/1435364830/',
'resource': 'http://www.google.com', 'url':
'http://www.google.com/', 'response_code': 1, 'scan_date': '2015-
06-27 00:27:10', 'scan_id':
'dd014af5ed6b38d9130e3f466f850e46d21b951199d53a18ef29ee9341614eaf
-1435364830', 'verbose_msg': 'Scan finished, scan information
embedded in this object', 'filescan_id': null, 'positives': 0,
'total': 63, 'scans': {'CLEAN MX': {'detected': false, 'result':
'clean site'}, 'VX Vault': {'detected': false, 'result': 'clean
```

SG

site'}, 'ZDB Zeus': {'detected': false, 'result': 'clean site'},
'Tencent': {'detected': false, 'result': 'clean site'},
'MalwarePatrol': {'detected': false, 'result': 'clean site'},
'ZCloudsec': {'detected': false, 'result': 'clean site'},
'PhishLabs': {'detected': false, 'result': 'unrated site'},
'Zerofox': {'detected': false, 'result': 'clean site'},
'K7AntiVirus': {'detected': false, 'result': 'clean site'},
'Quttera': {'detected': false, 'result': 'suspicious site'},
'Spam404': {'detected': false, 'result': 'clean site'}, 'AegisLab
WebGuard': {'detected': false, 'result': 'clean site'},
'MalwareDomainList': {'detected': false, 'result': 'clean site',
'detail':
'http://www.malwaredomainlist.com/mdl.php?search=www.google.com'}
, 'ZeusTracker': {'detected': false, 'result': 'clean site',
'detail':
'https://zeustracker.abuse.ch/monitor.php?host=www.google.com'},
'zvelo': {'detected': false, 'result': 'clean site'}, 'Google
Safebrowsing': {'detected': false, 'result': 'clean site'},
'Kaspersky': {'detected': false, 'result': 'clean site'},
'BitDefender': {'detected': false, 'result': 'clean site'},
'Dr.Web': {'detected': false, 'result': 'clean site'},
'ADMINUSLabs': {'detected': false, 'result': 'clean site'}, 'C-
SIRT': {'detected': false, 'result': 'clean site'}, 'CyberCrime':
{'detected': false, 'result': 'clean site'}, 'Websense
ThreatSeeker': {'detected': false, 'result': 'clean site'},
'CRDF': {'detected': false, 'result': 'clean site'},
'Webutation': {'detected': false, 'result': 'clean site'},
'Trustwave': {'detected': false, 'result': 'clean site'}, 'Web
Security Guard': {'detected': false, 'result': 'clean site'}, 'G-
Data': {'detected': false, 'result': 'clean site'}, 'Malwarebytes

SG

hpHosts': {'detected': false, 'result': 'clean site'}, 'Wepawet': {'detected': false, 'result': 'clean site'}, 'AlienVault': {'detected': false, 'result': 'clean site'}, 'Emsisoft': {'detected': false, 'result': 'clean site'}, 'Malc0de Database': {'detected': false, 'result': 'clean site', 'detail': 'http://malc0de.com/database/index.php?search=www.google.com'}, 'SpyEyeTracker': {'detected': false, 'result': 'clean site', 'detail': 'https://spyeyetracker.abuse.ch/monitor.php?host=www.google.com'}, 'malwares.com URL checker': {'detected': false, 'result': 'clean site'}, 'Phishtank': {'detected': false, 'result': 'clean site'}, 'Malwared': {'detected': false, 'result': 'clean site'}, 'Avira': {'detected': false, 'result': 'clean site'}, 'OpenPhish': {'detected': false, 'result': 'clean site'}, 'Antiy-AVL': {'detected': false, 'result': 'clean site'}, 'SCUMWARE.org': {'detected': false, 'result': 'clean site'}, 'FraudSense': {'detected': false, 'result': 'clean site'}, 'Opera': {'detected': false, 'result': 'clean site'}, 'Comodo Site Inspector': {'detected': false, 'result': 'clean site'}, 'Malekal': {'detected': false, 'result': 'clean site'}, 'ESET': {'detected': false, 'result': 'clean site'}, 'Sophos': {'detected': false, 'result': 'unrated site'}, 'Yandex Safebrowsing': {'detected': false, 'result': 'clean site', 'detail': 'http://yandex.com/infected?l10n=en&url=http://www.google.com/'}, 'SecureBrain': {'detected': false, 'result': 'clean site'}, 'Malware Domain Blocklist': {'detected': false, 'result': 'clean site'}, 'Blueliv': {'detected': false, 'result': 'clean site'}, 'Netcraft': {'detected': false, 'result': 'unrated site'}, 'PalevoTracker': {'detected': false, 'result': 'clean site'},

SG

```
'AutoShun': {'detected': false, 'result': 'unrated site'},
'ThreatHive': {'detected': false, 'result': 'clean site'},
'ParetoLogic': {'detected': false, 'result': 'clean site'},
'Rising': {'detected': false, 'result': 'clean site'},
'URLQuery': {'detected': false, 'result': 'unrated site'},
'StopBadware': {'detected': false, 'result': 'unrated site'},
'Sucuri SiteCheck': {'detected': false, 'result': 'clean site'},
'Fortinet': {'detected': false, 'result': 'clean site'},
'ZeroCERT': {'detected': false, 'result': 'clean site'}, 'Baidu-
International': {'detected': false, 'result': 'clean site'}}}"""


      ##stores JSON response to variable for calling specific
sections in the next block of code


     #Script change made by Shelly Giesbrecht to ensure the basic
lookup would report back due to issues encountered with the JSON
processing code below. Change was to comment out everything until
the line of #'s below
          #response_dict = simplejson.loads(json)


          ##pull critical snippets from report and convert to
strings for output formatting
          #permalink = response_dict.get('permalink', {})
          #scanDate = response_dict.get('scan_date', {})
          #avHit = response_dict.get('positives', {})
          #total = response_dict.get('total', {})
```

```
        ##convert numbers to string for output formatting
        #avHit = str(avHit)
        #total = str(total)
        #ratio = avHit + '/' + total

        ##format results and write to screen and results.txt
        #resultsString = (domain + ' was scanned on ' + scanDate
+ ' and contained a ' + ratio + ' AV detection ratio. See full
report in results file for further information')
        #print(resultsString)
        #resultsOutput = domain + ',' + ratio + ',' + permalink
+ '\n'
        #print('Writing to results.txt\n')
        #results.write(resultsOutput)
    ################################################################


    #input/output files. Domains should NOT be obfuscated!!! Can
accept HTTP, www., or straight domain
    badDomains = open('domains.txt', 'r') ###### change path for
input file

    #Script change made by Shelly Giesbrecht to ensure the basic
lookup would report back due to issues encountered with the JSON
```

SG

processing code below. Change was to comment out everything until the line of #'s below

```
    #results = open('results.txt', 'w') ###### change path for input file
    #results.write('') #clear any current data in output file
    ##################################################################


    #iterate through domain input file and pass 'domain' to functions
    for line in badDomains:
        domain = line.rstrip('\n')
        print('Passing ' + domain + ' to VirusTotal\n')
        DomainScanner(domain)
    DomainReportReader(domain)
```

SG

Last Updated: December 13th, 2018

Upcoming SANS Training

Click here to view a list of all SANS Courses

SANS Sonoma 2019	Santa Rosa, CAUS	Jan 14, 2019 - Jan 19, 2019	Live Event
SANS Threat Hunting London 2019	London, GB	Jan 14, 2019 - Jan 19, 2019	Live Event
SANS Amsterdam January 2019	Amsterdam, NL	Jan 14, 2019 - Jan 19, 2019	Live Event
SANS Miami 2019	Miami, FLUS	Jan 21, 2019 - Jan 26, 2019	Live Event
Cyber Threat Intelligence Summit & Training 2019	Arlington, VAUS	Jan 21, 2019 - Jan 28, 2019	Live Event
SANS Dubai January 2019	Dubai, AE	Jan 26, 2019 - Jan 31, 2019	Live Event
SANS Las Vegas 2019	Las Vegas, NVUS	Jan 28, 2019 - Feb 02, 2019	Live Event
SANS Security East 2019	New Orleans, LAUS	Feb 02, 2019 - Feb 09, 2019	Live Event
SANS SEC504 Stuttgart 2019 (In English)	Stuttgart, DE	Feb 04, 2019 - Feb 09, 2019	Live Event
SANS Anaheim 2019	Anaheim, CAUS	Feb 11, 2019 - Feb 16, 2019	Live Event
SANS Northern VA Spring- Tysons 2019	Vienna, VAUS	Feb 11, 2019 - Feb 16, 2019	Live Event
SANS London February 2019	London, GB	Feb 11, 2019 - Feb 16, 2019	Live Event
SANS Zurich February 2019	Zurich, CH	Feb 18, 2019 - Feb 23, 2019	Live Event
SANS Secure Japan 2019	Tokyo, JP	Feb 18, 2019 - Mar 02, 2019	Live Event
SANS Scottsdale 2019	Scottsdale, AZUS	Feb 18, 2019 - Feb 23, 2019	Live Event
SANS New York Metro Winter 2019	Jersey City, NJUS	Feb 18, 2019 - Feb 23, 2019	Live Event
SANS Dallas 2019	Dallas, TXUS	Feb 18, 2019 - Feb 23, 2019	Live Event
SANS Riyadh February 2019	Riyadh, SA	Feb 23, 2019 - Feb 28, 2019	Live Event
SANS Brussels February 2019	Brussels, BE	Feb 25, 2019 - Mar 02, 2019	Live Event
SANS Reno Tahoe 2019	Reno, NVUS	Feb 25, 2019 - Mar 02, 2019	Live Event
Open-Source Intelligence Summit & Training 2019	Alexandria, VAUS	Feb 25, 2019 - Mar 03, 2019	Live Event
SANS Baltimore Spring 2019	Baltimore, MDUS	Mar 02, 2019 - Mar 09, 2019	Live Event
SANS Training at RSA Conference 2019	San Francisco, CAUS	Mar 03, 2019 - Mar 04, 2019	Live Event
SANS Secure India 2019	Bangalore, IN	Mar 04, 2019 - Mar 09, 2019	Live Event
SANS St. Louis 2019	St. Louis, MOUS	Mar 11, 2019 - Mar 16, 2019	Live Event
SANS London March 2019	London, GB	Mar 11, 2019 - Mar 16, 2019	Live Event
SANS Secure Singapore 2019	Singapore, SG	Mar 11, 2019 - Mar 23, 2019	Live Event
SANS San Francisco Spring 2019	San Francisco, CAUS	Mar 11, 2019 - Mar 16, 2019	Live Event
SANS Bangalore January 2019	OnlineIN	Jan 07, 2019 - Jan 19, 2019	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced