CrowdResponse User Guide v1.0.6

CROWDSTRIKE PROFESSIONAL SERVICES
web: WWW.CROWDSTRIKE.COM | twitter: @CROWDSTRIKE
email: SERVICES@CROWDSTRIKE.COM

CrowdStrike

Contents

REVISION HISTORY

Version	Date	Revisions	Author
1.0	March 13, 2014	Initial Draft	J. Weissert
1.0.2.0	May 16, 2014	Added Revision History Added three new modules: Drivers, Handles, and PSStrings (Functionality and Features section)	J. Weissert
1.0.3.0	August 23, 2014	Add two new modules: RegFile and RegDump (Functionality and Features section)	C. Tilbury
1.0.4.0	December 18, 2014	Add three new modules: Prefetch, Superfetch, and Shim (Functionality and Features section)	C. Tilbury
1.0.5.0	April 17, 2015	New module: Mal (Functionality and Features section)	R. Keir
1.0.6.0	May 10, 2016	New modules: Jobs, Tasks	Reed Pochron

EXECUTIVE SUMMARY

CrowdResponse

CrowdResponse is a lightweight Windows console application designed to aid in the gathering of system information for incident response and security engagements. The application contains numerous modules, each of them invoked by providing specific command line parameters to the main application. Modules are all built into the main application in C++ language utilizing the Win32 API to achieve their functionality.

Supported Operating Systems

CrowdResponse supports 32-bit and 64-bit versions of:

- Workstation:
  - Windows XP
  - Windows Vista
  - Windows 7
  - Windows 8
  - Windows 8.1
  - Windows 10
- Server:
  - Windows Server 2003
  - Windows Server 2008
  - Windows Server 2008 R2
  - Windows Server 2012

Additional Requirements

- For best results, CrowdResponse should be run with administrative privileges

FUNCTIONALITY & FEATURES

General Overview

CrowdResponse is intended for use by organizations to run on-demand data scanning of their host environment. The tool can be used to scan a system for malware by using embedded YARA (Yet Another Regex Analyzer) signatures and collect contextual information such as process and file listings to assist incident responders.

Functionality & Features

The key functionality of CrowdResponse is provided through modules. The primary module is the Main Tool, which is supported by additional modules known as sub-tools, including Directory Listing, Process List, and YARA Rule Process. Together, the Main tool and associated sub-tools allow the investigator to collect different types of information from the scan. Each of these modules is discussed in more detail in the following sections. Additional sub-tool modules will be released over time to further enhance the overall capabilities of the tool for the investigator.

Please see the accompanying *readme-CrowdResponse.html* file for comprehensive details on the various modules and their usage.

Deployment

Embedding YARA Rules

The main functionality provided by CrowdResponse is the ability to search for specific YARA rules across an environment. Using these rules, the investigator can search for hits against running process binaries and memory.

The YARA scans can be accomplished in two different ways. The first option is for the user to denote the specific directory or directories at the command prompt to indicate where the rules are located, which is accomplished through an HTTP POST function. The second option is for the user to embed YARA rules into the configuration file. This configuration file can also contain specific options and variables associated with the other sub-tools. The configuration file input option is likely the easier option for most users. Additionally, the user may choose to use different configurations for different sets of hosts or for initial/subsequent runs. The image below shows a configuration file with embedded YARA rules.

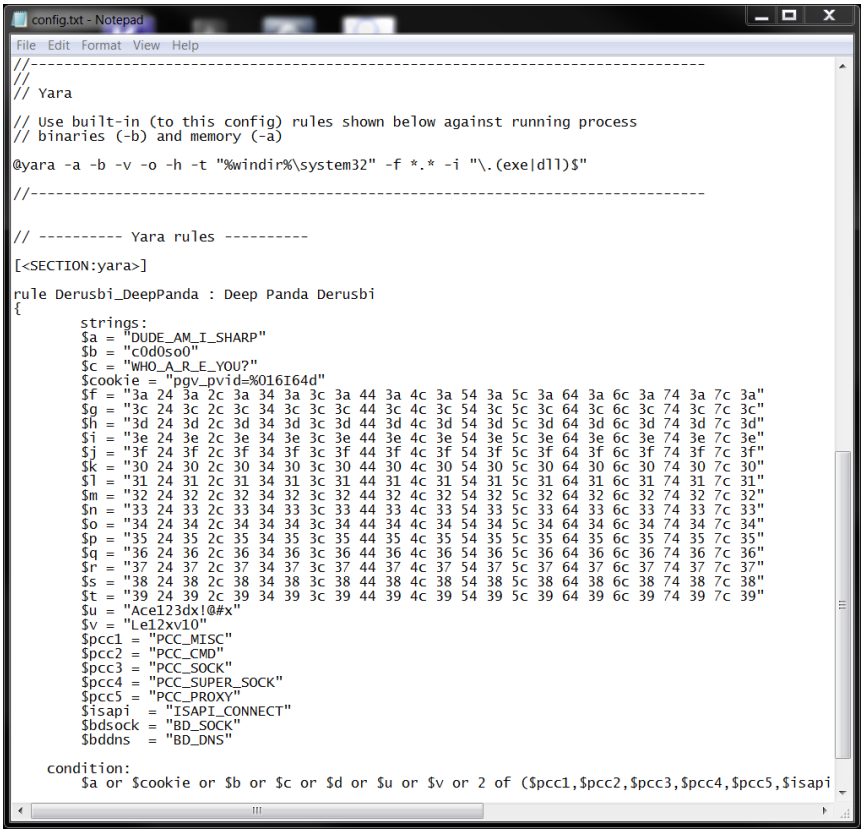


Figure 1: Example Config File

Deployment Methods

CrowdResponse can be deployed via several methods, including those discussed below:

- **SCCM** – In a managed environment, organizations can utilize System Center Configuration Manager (SCCM) to deploy CrowdResponse to multiple workstations and servers at once. The tool will look to the config file for the variables and rules to use in the scan and associated output.

- **PSExec** – Similarly to SCCM, organizations can utilize PSExec to run a script that will deploy CrowdResponse to multiple workstations and servers at once. The tool will look to the config file for the variables and rules to use in the scan and associated output.

- **Falcon Host** – For organizations that utilize CrowdStrike's Falcon Host product, the agent can be used to deploy CrowdResponse as well. The tool will look to the config file for the variables and rules to use in the scan and associated output.

- **Command Line** – Windows Command Line can be utilized to manually execute the CrowdResponse scan on individual hosts. In this manner, users can actually specify individual actions at the command prompt rather than relying on the configuration file if desired. This is the method utilized throughout this user guide for ease of use.

Note: This user guide focuses on execution of the CrowdResponse tool via the command line. As such, the instructions and screenshots that follow are associated with execution and output from the command line deployment method. The other deployment methods are partially dependent on individual organizational settings and environments, making it difficult to provide a guide for each.

SCAN EXECUTION AND OUTPUT

The CrowdResponse sub-tools (modules) and the associated options can be executed at the command line. The format utilized for executing the tool and options is as follows:

```
CrowdResponse [opts] @tool_name [params] @tool_name [params]... etc.
```

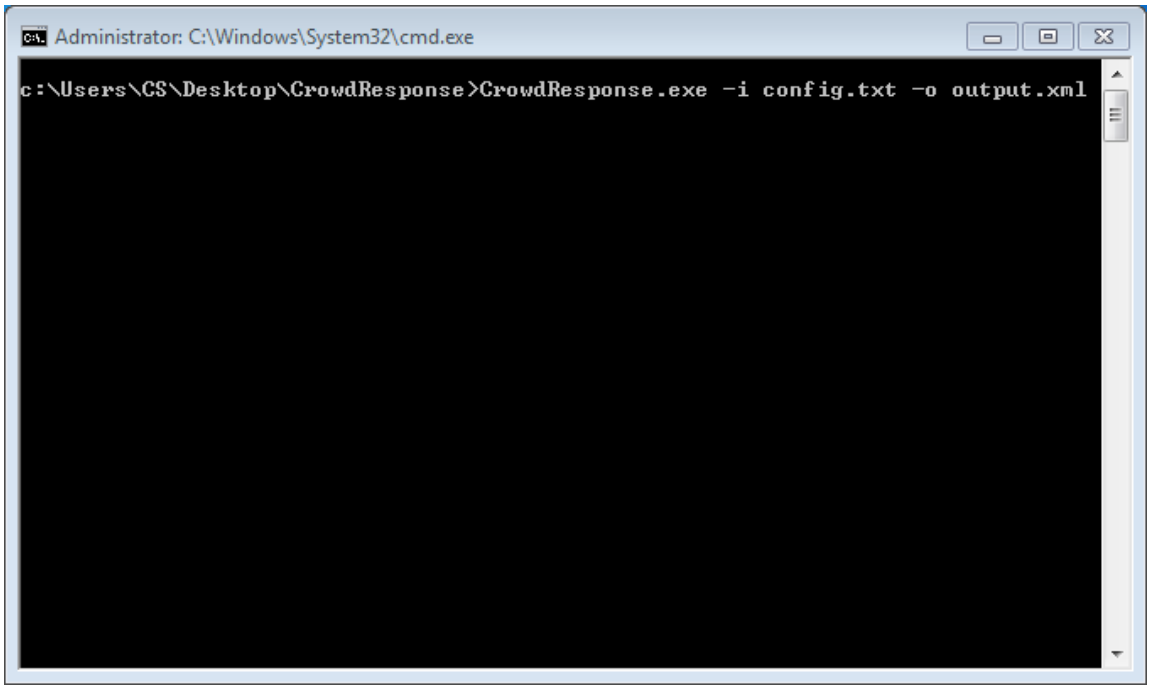


Figure 2: Example Command Line Input

This command will execute the main “CrowdResponse.exe” file which calls the associated processes defined in the configuration file “config.txt”. The data associated with the scan is then output to a file called “output.xml”.

In addition to the simple syntax utilized above, the following table lists the full variables that may be called from the command line during execution of the scan, which were detailed previously under the Main Tool Options section:

The output file (listed after “-o”) also understands the following syntax, outside of these system environment variables:

Each of the sub-tools also has additional variables associated with its functionality, which can be adapted in the configuration file itself. This information is provided in the Sub-Tools section of this document.

Example Output (Command Line)

For the purpose of providing example screenshots, the simple command noted previously (`CrowdResponse.exe –i config.txt –o output.xml`) was executed on a virtual image utilizing the following config file information:

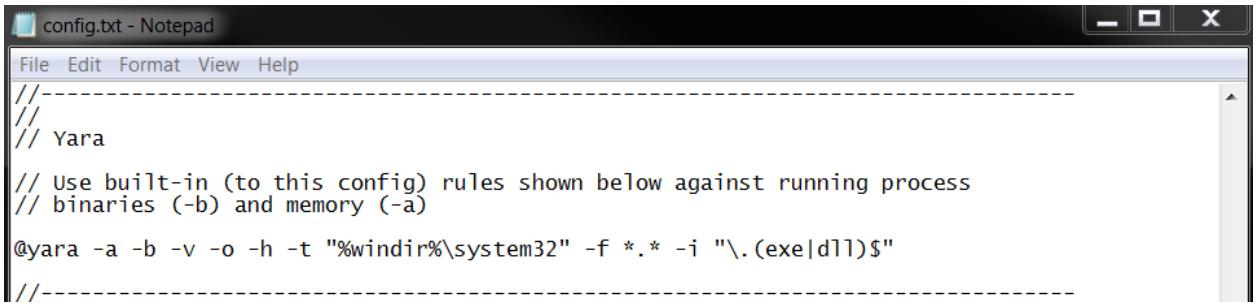


Figure 3: Example Config File with YARA Specified Variables

As you can see in the screenshot above, the YARA sub-tool includes several variables including the options to scan active processes in memory (-a), scan active process executable files (-b), enable verbose informational output (-v), scan all loaded module files of active processes (-o), only show positive hits (-h), start target file directory (-t), apply a target file name mask (-f), and apply a target file path inclusion regex filter (-i).

The command line execution of the config file results in the following output:

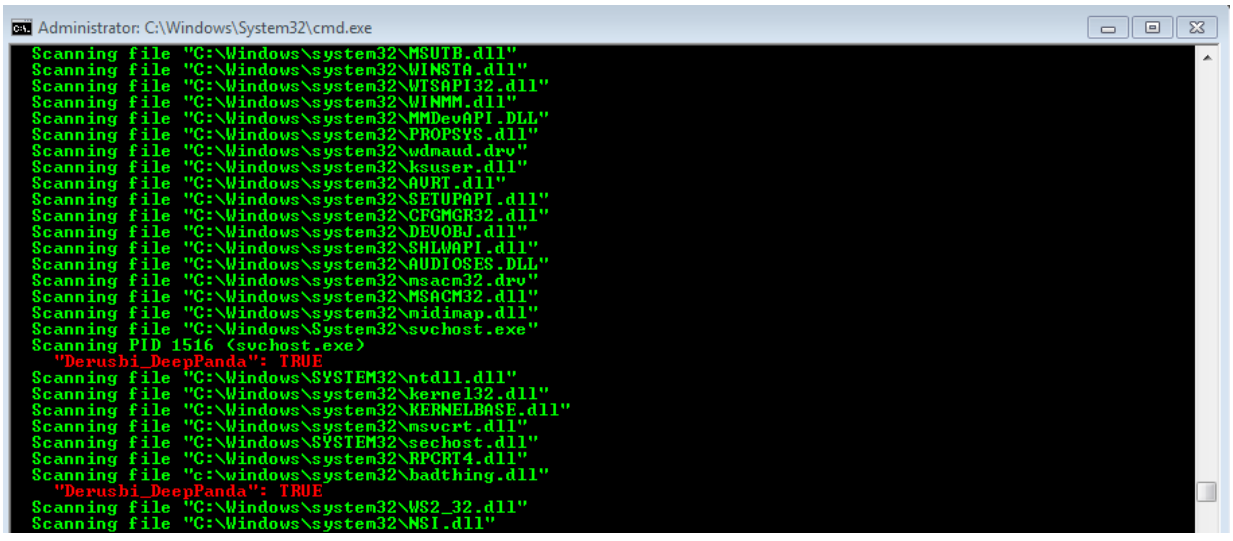


Figure 4: Example Command Line Output Identifying Positive Hits

Output Key Observations:

- (-h): The output identifies a TRUE hit on the “Derusbi_DeepPanda” YARA rule provided in an earlier screenshot. The scan first identifies the parent process (svchost.exe) and then subsequently identifies the underlying bad file associated with the malware (badthing.dll). The scan utilizes the –o variable to load the module files of the active processes, otherwise only the parent processes would show.
- (-i): The only results that are returned are for files and processes that end in with a .exe or .dll extension. This is the result of the target file path inclusion regex filter in the YARA sub-tool section of the config file.
- (-h): Only the positive hits are reported.
- In addition to the data that is passed back in the command line, the output is also exported to an .xml file that is located in the default folder where the CrowdResponse.exe file is located, unless otherwise specified.

OUTPUT CONVERSION

CRconvert

CrowdResponse results may be viewed in a variety of ways, particularly when leveraging CrowdStrike's CRconvert. By default, output from CrowdResponse is provided in an XML file. CRconvert will flatten this XML to CSV, TSV, HTML or plain Text, if desired. The HTML output may be viewed in any browser. The CSV and TSV output may be processed via a spreadsheet application of your choice, such as Microsoft Excel or OpenOffice Calc, or via a data analysis platform such as Splunk. The various format options were created to support the different needs and analysis preferences of the end user.

Please see the accompanying *readme-CRConvert.html* file for comprehensive details on syntax and options.

CRconvert Overview

```
CRconvert -f <path/mask>
          [-b <table>]
          [-d <database>]
          [-e <prefix>]
          [-m <rows>]
          [-n <name>]
          [-o <dir>]
          [-achjpqrstvx]

  -a              Append output files (default is overwrite)
  -b <table>      Database table name for option -d. Default is "Hashes"
  -c              Output in CSV format
  -d <database>   Path to SQLite3 database of whitelisted SHA256 hashes
  -e <prefix>     Output file name prefix. Default is "CrowdResponse_"
  -f <path/mask>  File name path/mask. Default mask is "*.xml"
  -h              Output in HTML format
  -j              Delete all output files on error
  -m <rows>       Limit rows per CSV output file creating new ones when reached
  -n <name>       Database SHA256 column name for option -d. Default is "SHA256"
  -o <dir>        Output directory for files. Default is current
  -p              Use low CPU priority (idle) to lessen load on the system
  -q              Quit the application immediately after decryption
  -r              Recursively search input directory when looking for files
  -s              Do not use XML file name as first field ("system")
  -t              Use tabs to separate CSV output fields instead of commas
  -v              Verbose output
  -x              Output in text format
  -z              Ignore zero byte sized file when using whitelist (option -d)
```

Splunk

Depending on your environment, Splunk may be a very powerful way to leverage CrowdResponse output. The CSV output that CRconvert can create is Splunk-friendly and can be indexed with ease. By default, CSV column headers will be used as field names in Splunk when setting the sourcetype to CSV, allowing for easy labeling of the data.

Once ingested, analysis of the data is as powerful as your queries. Within the context of CrowdResponse, Splunk reports, dashboards and alerts can simplify analysis of output files by utilizing key events. For instance, you can enable a Splunk alert if “result=TRUE” in the YARA output, indicating a match of a YARA. Thus, rather than searching through the output for the positive hits, Splunk will easily pull these records to your attention.

Modules may contain multiple timestamps (e.g. created, modified, and accessed). In order to specify which timestamp you’d like Splunk to use at index time as the primary (for quickly filtering based on timeframe), a new sourcetype may be defined per module in your props.conf including the parameters below:

- INDEXED_EXTRACTIONS = csv
- TIME_FORMAT = %Y-%m-%dT%H:%M:%S%Z
- TIME_PREFIX = ^([^,]*,){6}

Note: TIME_PREFIX is used to provide a regular expression to represent where Splunk should begin looking for the appropriate timestamp, based on field.

ANALYSIS

System Information

Upon conversion of the XML data via CRconvert, you'll have a CSV, TSV or HTML file for each sub-tool module selected, in addition to one for overall main system information collected during execution. The system information provides an inventory of the systems that have been scanned along with associated data point. The output will be similar to what is seen below:

Splunk

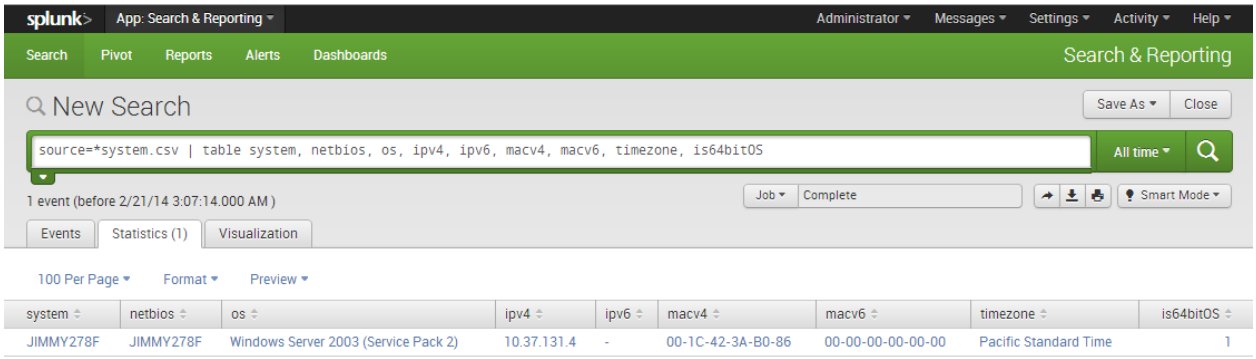


Figure 5: Example System Information - Splunk

Sample Query:

```
source=*system.csv | table system, netbios, os, ipv4, ipv6, macv4,
macv6, timezone, is64bitOS
```

Excel

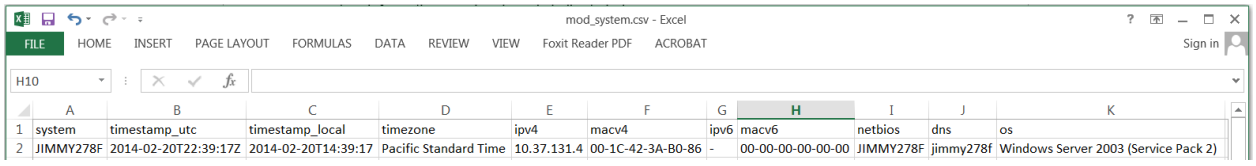


Figure 6: Example System Information - Excel

Web Browser

CROWDSTRIKE

www.crowdstrike.com

Produced by the free CrowdStrike tool *CrowdResponse*

Module: system

system	timestamp_utc	timestamp_local	timezone	ipv4	macv4	ipv6	macv6	netbios	dns	os
JIMMY278F	2014-02-20T22:39:17Z	2014-02-20T14:39:17	Pacific Standard Time	10.37.131.4	00-1C-42-3A-B0-86	-	00-00-00-00-00-00	JIMMY278F	jimmy278f	Windows Server 2003 (Service Pack 2)

Figure 7: Example System Information – Web

YARA Module

Following the System information that shows inventory information, it is important to review any YARA matches. The steps required to investigate a YARA match will vary based on method of viewing the data, but the methodology will remain the same. Examples of the YARA match information output are shown in the following sections and screenshots.

The first step in analysis of the YARA module is to identify any YARA results that returned "TRUE." The YARA module output will yield the name of the affected system, the YARA rule that matched, the name and path of the matching file, as well as the process ID in use by the operating system (if applicable).

In the output below, "svchost.exe" and "badthing.dll" both match the YARA rule named "Derusbi_DeepPanda." The Process ID ("pid") reported for svchost.exe (2940) indicates that it is the parent process, while the identified file "badthing.dll" does not have a process ID value, indicating it is not a process. Instead, this result is the offending module loaded by process ID 2940. This malicious file was identified by using the "-o" option for @YARA to scan all modules loaded by an active process, which leads the analyst directly to the issue.

Splunk

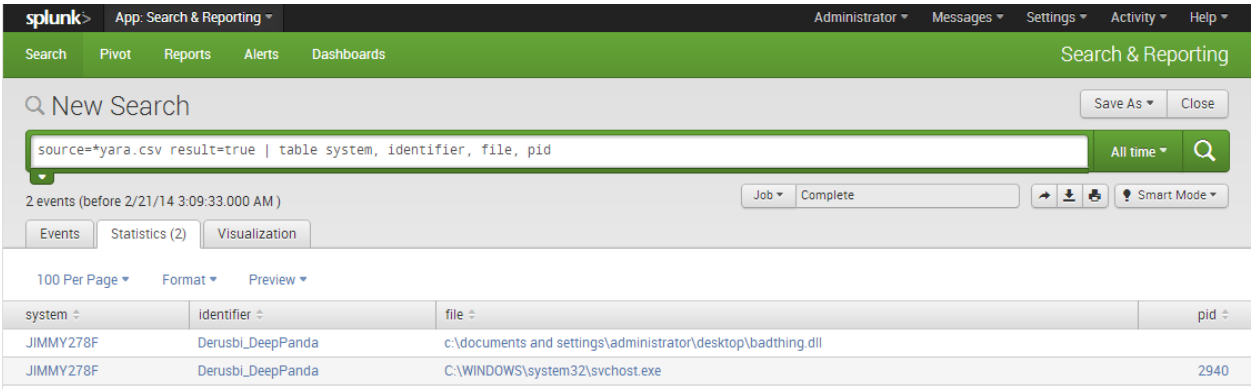


Figure 8: Example YARA Module Output – Splunk

Sample Query:

```
source=*YARA.csv result=true | table system, identifier, file, pid
```

Excel

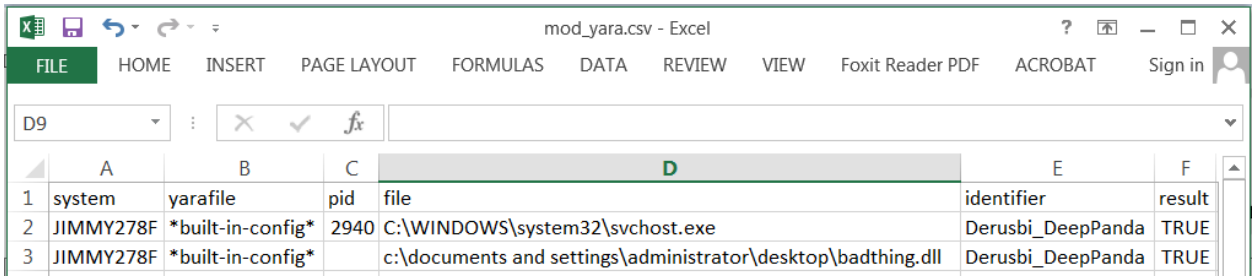


Figure 9: Example YARA Module Output – Excel

Web Browser

Figure 10: Example YARA Module Output – Web Browser

Note: Depending on signature quality, it's possible that false positive YARA hits on memory will result with some processes like Anti-Virus.

PSList Module

Using the process ID (2940) and system name (JIMMY278F) identified previously, the PSList Module output can be used to cross-reference details on that specific process. In this example note that the svchost.exe process itself is not malicious, as it is only being used to load the offending DLL as a service.

PSList provides numerous fields of interest related to a specific process, such as the system name, file path and name, size, process ID, creation date, MD5 or SHA256 hash, command line parameters used to execute the process and detection of thread injection. These fields are a small sampling of what's available with PSList, the rest of which may be explored by the analyst as desired and are included in Appendix A.

Looking at the command line details for the svchost.exe process shows the parameter used to reference and execute the malicious DLL as "Badservice." Also of value is to verify the MD5 or SHA256 hash of "svchost.exe" with repositories of known hashes. In this case, the hash identifies svchost.exe as the standard, non-malicious Microsoft version.

Splunk

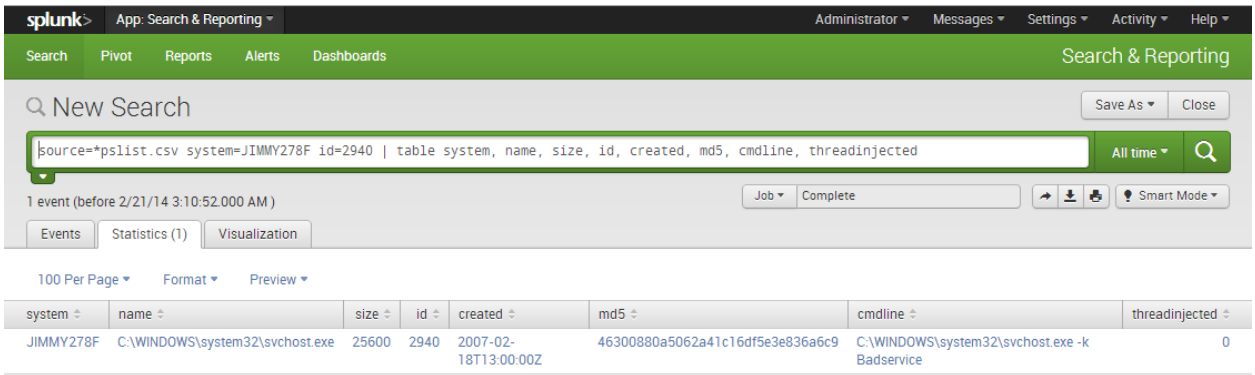
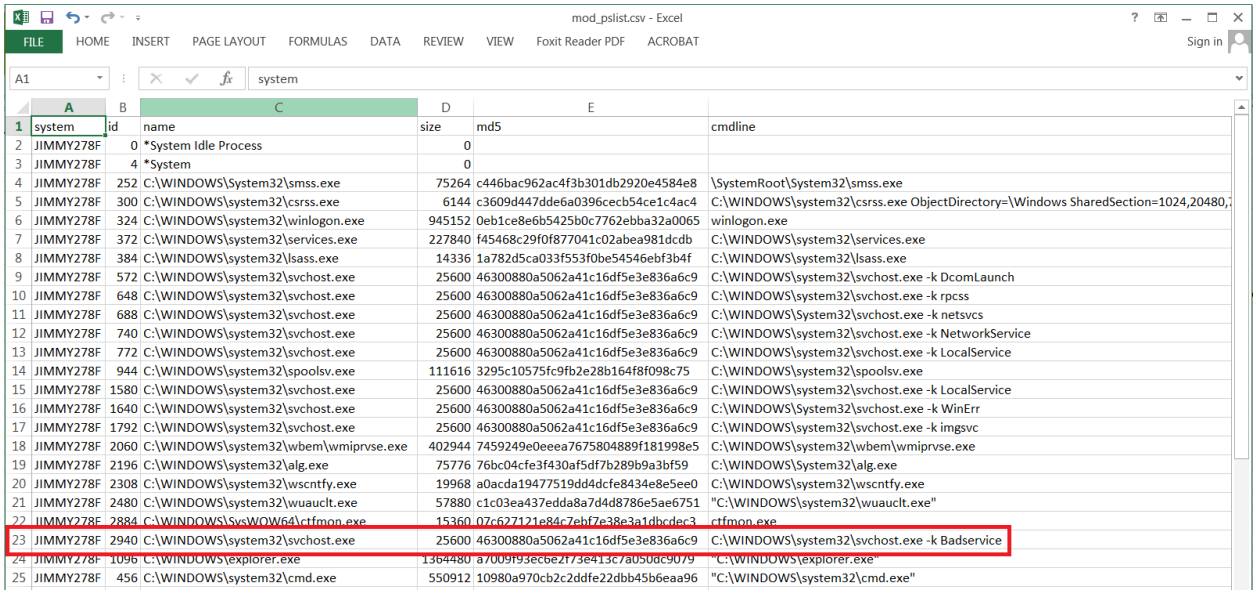


Figure 11: Example PSList Module Output – Splunk

Sample Query:

```
source=*pslist.csv system=JIMMY278F id=2940 | table system, name,
size, id, created, md5, cmdline, threadinjected
```

Excel



mod_pslist.csv - Excel

FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW Foxit Reader PDF ACROBAT Sign in

A1 system

	A	B	C	D	E	
1	system	id	name	size	md5	cmdline
2	JIMMY278F	0	*System Idle Process	0		
3	JIMMY278F	4	*System	0		
4	JIMMY278F	252	C:\WINDOWS\System32\smss.exe	75264	c446bac962ac4f3b301db2920e4584e8	\SystemRoot\System32\smss.exe
5	JIMMY278F	300	C:\WINDOWS\system32\csrss.exe	6144	c3609d447dde6a0396cecb54ce1c4ac4	C:\WINDOWS\system32\csrss.exe ObjectDirectory=\Windows SharedSection=1024,20480,7
6	JIMMY278F	324	C:\WINDOWS\system32\winlogon.exe	945152	0eb1ce8e6b5425b0c7762ebba32a0065	winlogon.exe
7	JIMMY278F	372	C:\WINDOWS\system32\services.exe	227840	f45468c29f0f877041c02abea981dcdb	C:\WINDOWS\system32\services.exe
8	JIMMY278F	384	C:\WINDOWS\system32\lsass.exe	14336	1a782d5ca033f553f0be54546ebf3b4f	C:\WINDOWS\system32\lsass.exe
9	JIMMY278F	572	C:\WINDOWS\system32\svchost.exe	25600	46300880a5062a41c16df5e3e836a6c9	C:\WINDOWS\system32\svchost.exe -k DcomLaunch
10	JIMMY278F	648	C:\WINDOWS\system32\svchost.exe	25600	46300880a5062a41c16df5e3e836a6c9	C:\WINDOWS\system32\svchost.exe -k rpcss
11	JIMMY278F	688	C:\WINDOWS\system32\svchost.exe	25600	46300880a5062a41c16df5e3e836a6c9	C:\WINDOWS\System32\svchost.exe -k netsvcs
12	JIMMY278F	740	C:\WINDOWS\system32\svchost.exe	25600	46300880a5062a41c16df5e3e836a6c9	C:\WINDOWS\system32\svchost.exe -k NetworkService
13	JIMMY278F	772	C:\WINDOWS\system32\svchost.exe	25600	46300880a5062a41c16df5e3e836a6c9	C:\WINDOWS\system32\svchost.exe -k LocalService
14	JIMMY278F	944	C:\WINDOWS\system32\spoolsv.exe	111616	3295c10575fc9fb2e28b164f8f098c75	C:\WINDOWS\system32\spoolsv.exe
15	JIMMY278F	1580	C:\WINDOWS\system32\svchost.exe	25600	46300880a5062a41c16df5e3e836a6c9	C:\WINDOWS\system32\svchost.exe -k LocalService
16	JIMMY278F	1640	C:\WINDOWS\system32\svchost.exe	25600	46300880a5062a41c16df5e3e836a6c9	C:\WINDOWS\System32\svchost.exe -k WinErr
17	JIMMY278F	1792	C:\WINDOWS\system32\svchost.exe	25600	46300880a5062a41c16df5e3e836a6c9	C:\WINDOWS\system32\svchost.exe -k imgsvc
18	JIMMY278F	2060	C:\WINDOWS\system32\wbem\wmiprvse.exe	402944	7459249e0eeea7675804889f181998e5	C:\WINDOWS\system32\wbem\wmiprvse.exe
19	JIMMY278F	2196	C:\WINDOWS\system32\alg.exe	75776	76bc04cfe3f430af5df7b289b9a3bf59	C:\WINDOWS\System32\alg.exe
20	JIMMY278F	2308	C:\WINDOWS\system32\wscntfy.exe	19968	a0acda19477519dd4dcfe8434e8e5ee0	C:\WINDOWS\system32\wscntfy.exe
21	JIMMY278F	2480	C:\WINDOWS\system32\wuauclt.exe	57880	c1c03ea437edda8a7d4d8786e5ae6751	"C:\WINDOWS\system32\wuauclt.exe"
22	JIMMY278F	2884	C:\WINDOWS\SysWOW64\ctfmon.exe	15360	07c627121e84c7ebf7e38e3a1dbcdec3	ctfmon.exe
23	JIMMY278F	2940	C:\WINDOWS\system32\svchost.exe	25600	46300880a5062a41c16df5e3e836a6c9	C:\WINDOWS\system32\svchost.exe -k Badservice
24	JIMMY278F	1096	C:\WINDOWS\explorer.exe	1364480	a7009f93ecbe2f73e413c7a050dc9079	"C:\WINDOWS\explorer.exe"
25	JIMMY278F	456	C:\WINDOWS\system32\cmd.exe	550912	10980a970cb2c2ddfe22dbb45b6eaa96	"C:\WINDOWS\system32\cmd.exe"

Figure 12: Example PSList Module Output – Excel

Web Browser

Figure 13: Example PSList Module Output – Web Browser

www.crowdstrike.com

Produced by the free CrowdStrike tool *CrowdResponse*

Module: pslist

system	pid	name	size	companyname	filedescription	fileversion
JIMMY278F	0	*System Idle Process	0			
JIMMY278F	4	*System	0			
JIMMY278F	252	C:\WINDOWS\System32\smss.exe	75264	Microsoft Corporation	Windows NT Session Manager	5.2.3790.3959 (srv03_sp2_rtm.070216-1710)
JIMMY278F	300	C:\WINDOWS\system32\csrss.exe	6144	Microsoft Corporation	Client Server Runtime Process	5.2.3790.1830 (srv03_sp1_rtm.050324-1447)
JIMMY278F	324	C:\WINDOWS\system32\winlogon.exe	945152	Microsoft Corporation	Windows NT Logon Application	5.2.3790.4516 (srv03_sp2_qfe.090518-1415)
JIMMY278F	372	C:\WINDOWS\system32\services.exe	227840	Microsoft Corporation	Services and Controller app	5.2.3790.4550 (srv03_sp2_qfe.090713-1210)
JIMMY278F	384	C:\WINDOWS\system32\lsass.exe	14336	Microsoft Corporation	LSA Shell	5.2.3790.1830 (srv03_sp1_rtm.050324-1447)
JIMMY278F	572	C:\WINDOWS\system32\svchost.exe	25600	Microsoft Corporation	Generic Host Process for Win32 Services	5.2.3790.3959 (srv03_sp2_rtm.070216-1710)
JIMMY278F	648	C:\WINDOWS\system32\svchost.exe	25600	Microsoft Corporation	Generic Host Process for Win32 Services	5.2.3790.3959 (srv03_sp2_rtm.070216-1710)
JIMMY278F	688	C:\WINDOWS\system32\svchost.exe	25600	Microsoft Corporation	Generic Host Process for Win32 Services	5.2.3790.3959 (srv03_sp2_rtm.070216-1710)
JIMMY278F	944	C:\WINDOWS\system32\spoolsv.exe	111616	Microsoft Corporation	Spooler SubSystem App	5.2.3790.4804 (srv03_sp2_qfe.101210-0234)
JIMMY278F	1792	C:\WINDOWS\system32\svchost.exe	25600	Microsoft Corporation	Generic Host Process for Win32 Services	5.2.3790.3959 (srv03_sp2_rtm.070216-1710)
JIMMY278F	2060	C:\WINDOWS\system32\wbem\wmiprvse.exe	402944	Microsoft Corporation	WMI	5.2.3790.4455 (srv03_sp2_qfe.090203-1205)
JIMMY278F	2196	C:\WINDOWS\system32\alg.exe	75776	Microsoft Corporation	Application Layer Gateway Service	5.2.3790.4076 (srv03_sp2_qfe.070507-2336)
JIMMY278F	2308	C:\WINDOWS\system32\wscntfy.exe	19968	Microsoft Corporation	Windows Security Center Notification App	5.2.3790.1830 (srv03_sp1_rtm.050324-1447)
JIMMY278F	2480	C:\WINDOWS\system32\wuauclt.exe	57880	Microsoft Corporation	Windows Update	7.6.7600.256 (winmain_wtr_wsus3sp2(oobla).12
JIMMY278F	2884	C:\WINDOWS\SysWOW64\ctfmon.exe	15360	Microsoft Corporation	CTF Loader	5.2.3790.1830 (srv03_sp1_rtm.050324-1447)
JIMMY278F	2940	C:\WINDOWS\system32\svchost.exe	25600	Microsoft Corporation	Generic Host Process for Win32 Services	5.2.3790.3959 (srv03_sp2_rtm.070216-1710)
JIMMY278F	1096	C:\WINDOWS\explorer.exe	1364480	Microsoft Corporation	Windows Explorer	6.00.3790.4093 (srv03_sp2_qfe.070603-2353)

DirList Module

The next logical analytical step is to look more closely at “badthing.dll”. As the file itself is not a process, PSList does not provide details about it. Instead, the analyst should reference the DirList module output. In the output below, we see a variety of details surrounding the file including the system(s) it’s present on, the full path, size, MD5 hash, created date, modified date, file description and internal name. Similar to PSList, the output shown here is a small sampling of the information available to the analyst, but the full list is provided in Appendix B. Armed with these details, the investigator can immediately analyze the malware’s metadata and establish indicators of compromise to determine the scope of infection in our environment.

Splunk

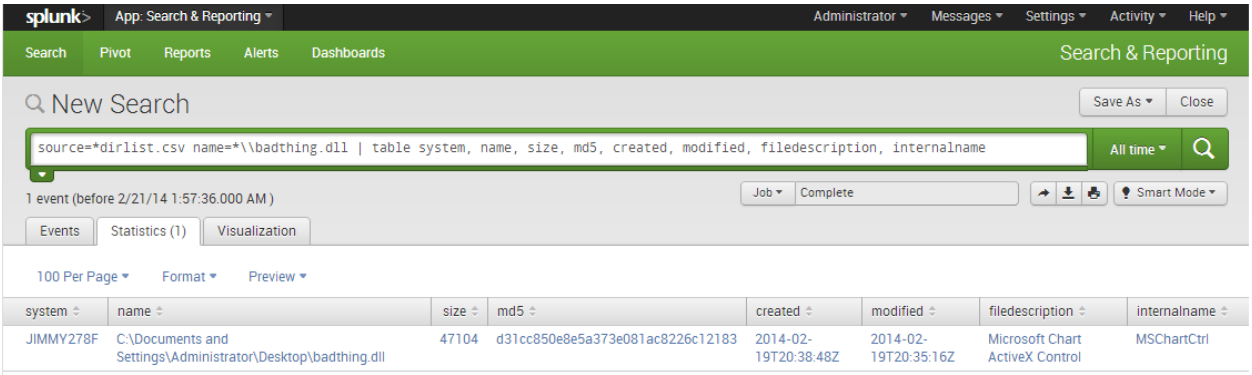
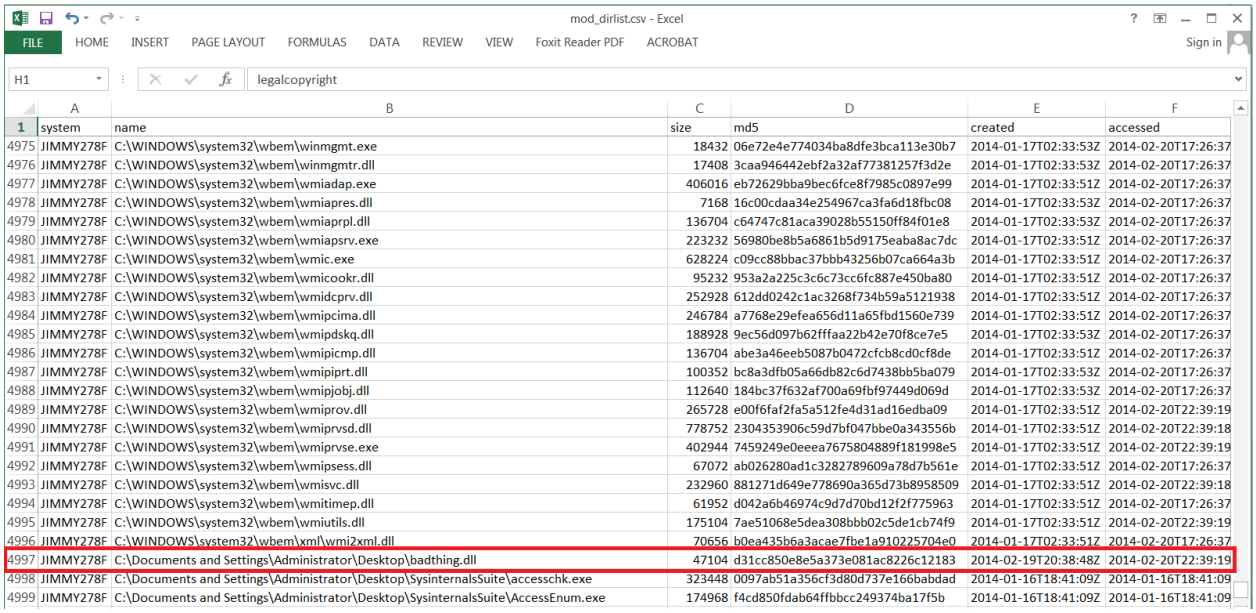


Figure 14: Example DirList Module Output – Splunk

Sample Query:

```
source=*dirlist.csv name=*\\badthing.dll | table system, name, size, md5, created, modified, filedescription, internalname
```

Excel



mod_dirlist.csv - Excel

FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW Foxit Reader PDF ACROBAT Sign in

H1 legalcopyright

	A	B	C	D	E	F
1	system	name	size	md5	created	accessed
4975	JIMMY278F	C:\WINDOWS\system32\wbem\winmgmt.exe	18432	06e72e4e774034ba8dfe3bca113e30b7	2014-01-17T02:33:53Z	2014-02-20T17:26:37
4976	JIMMY278F	C:\WINDOWS\system32\wbem\winmgmtr.dll	17408	3caa946442ebf2a32af77381257f3d2e	2014-01-17T02:33:53Z	2014-02-20T17:26:37
4977	JIMMY278F	C:\WINDOWS\system32\wbem\wmiadap.exe	406016	eb72629bba9bec6fce8f7985c0897e99	2014-01-17T02:33:51Z	2014-02-20T17:26:37
4978	JIMMY278F	C:\WINDOWS\system32\wbem\wmiapres.dll	7168	16c00cdaa34e254967ca3fa6d18fbc08	2014-01-17T02:33:53Z	2014-02-20T17:26:37
4979	JIMMY278F	C:\WINDOWS\system32\wbem\wmiaprpl.dll	136704	c64747c81aca39028b55150ff84f01e8	2014-01-17T02:33:53Z	2014-02-20T17:26:37
4980	JIMMY278F	C:\WINDOWS\system32\wbem\wmiapsrv.exe	223232	56980be8b5a6861b5d9175eaba8ac7dc	2014-01-17T02:33:51Z	2014-02-20T17:26:37
4981	JIMMY278F	C:\WINDOWS\system32\wbem\wmic.exe	628224	c09cc88bbac37bbb43256b07ca664a3b	2014-01-17T02:33:51Z	2014-02-20T17:26:37
4982	JIMMY278F	C:\WINDOWS\system32\wbem\wmicookr.dll	95232	953a2a225c3c6c73cc6fc887e450ba80	2014-01-17T02:33:51Z	2014-02-20T17:26:37
4983	JIMMY278F	C:\WINDOWS\system32\wbem\wmidcprv.dll	252928	612dd0242c1ac3268f734b59a5121938	2014-01-17T02:33:51Z	2014-02-20T17:26:37
4984	JIMMY278F	C:\WINDOWS\system32\wbem\wmipcima.dll	246784	a7768e29efea656d11a65fbd1560e739	2014-01-17T02:33:51Z	2014-02-20T17:26:37
4985	JIMMY278F	C:\WINDOWS\system32\wbem\wmipdskq.dll	188928	9ec56d097b62fffaa22b42e70f8ce7e5	2014-01-17T02:33:53Z	2014-02-20T17:26:37
4986	JIMMY278F	C:\WINDOWS\system32\wbem\wmipicmp.dll	136704	abe3a46eeb5087b0472cfcb8cd0cf8de	2014-01-17T02:33:51Z	2014-02-20T17:26:37
4987	JIMMY278F	C:\WINDOWS\system32\wbem\wmipiprt.dll	100352	bc8a3dfb05a66db82c6d7438bb5ba079	2014-01-17T02:33:53Z	2014-02-20T17:26:37
4988	JIMMY278F	C:\WINDOWS\system32\wbem\wmipjobj.dll	112640	184bc37f632af700a69fbf97449d069d	2014-01-17T02:33:53Z	2014-02-20T17:26:37
4989	JIMMY278F	C:\WINDOWS\system32\wbem\wmiprov.dll	265728	e00f6faf2fa5a512fe4d31ad16edba09	2014-01-17T02:33:51Z	2014-02-20T22:39:19
4990	JIMMY278F	C:\WINDOWS\system32\wbem\wmiprvsd.dll	778752	2304353906c59d7bf047bbe0a343556b	2014-01-17T02:33:51Z	2014-02-20T22:39:18
4991	JIMMY278F	C:\WINDOWS\system32\wbem\wmiprvse.exe	402944	7459249e0eeea7675804889f181998e5	2014-01-17T02:33:51Z	2014-02-20T22:39:19
4992	JIMMY278F	C:\WINDOWS\system32\wbem\wmipsess.dll	67072	ab026280ad1c3282789609a78d7b561e	2014-01-17T02:33:51Z	2014-02-20T17:26:37
4993	JIMMY278F	C:\WINDOWS\system32\wbem\wmisvc.dll	232960	881271d649e778690a365d73b8958509	2014-01-17T02:33:51Z	2014-02-20T22:39:18
4994	JIMMY278F	C:\WINDOWS\system32\wbem\wmitimep.dll	61952	d042a6b46974c9d7d70bd12f2f775963	2014-01-17T02:33:51Z	2014-02-20T17:26:37
4995	JIMMY278F	C:\WINDOWS\system32\wbem\wmiutils.dll	175104	7ae51068e5dea308bbb02c5de1cb74f9	2014-01-17T02:33:51Z	2014-02-20T22:39:19
4996	JIMMY278F	C:\WINDOWS\system32\wbem\xml\wmi2xml.dll	70656	b0ea435b6a3acae7fbe1a910225704e0	2014-01-17T02:33:51Z	2014-02-20T17:26:37
4997	JIMMY278F	C:\Documents and Settings\Administrator\Desktop\badthing.dll	47104	d31cc850e8e5a373e081ac8226c12183	2014-02-19T20:38:48Z	2014-02-20T22:39:19
4998	JIMMY278F	C:\Documents and Settings\Administrator\Desktop\SysinternalsSuite\accesschk.exe	323448	0097ab51a356cf3d80d737e166babdad	2014-01-16T18:41:09Z	2014-01-16T18:41:09
4999	JIMMY278F	C:\Documents and Settings\Administrator\Desktop\SysinternalsSuite\AccessEnum.exe	174968	f4cd850fdab64ffbbcc249374ba17f5b	2014-01-16T18:41:09Z	2014-01-16T18:41:09

Figure 15: Example DirList Module Output – Excel

Web Browser

JIMMY278F	C:\WINDOWS\system32\wbem\wmiutils.dll	175104	0x20	ARCHIVE	2014-01-17T02:33:51Z	2014-02-20T22:39:19Z	2007-02-18T13:00:00Z	Microsoft Corporation
JIMMY278F	C:\WINDOWS\system32\wbem\xml\wmi2xml.dll	70656	0x20	ARCHIVE	2014-01-17T02:33:51Z	2014-02-20T17:26:37Z	2007-02-18T13:00:00Z	Microsoft Corporation
JIMMY278F	C:\Documents and Settings\Administrator\Desktop\badthing.dll	47104	0x20	ARCHIVE	2014-02-19T20:38:48Z	2014-02-20T22:39:19Z	2014-02-19T20:35:16Z	Microsoft Corporation
JIMMY278F	C:\Documents and Settings\Administrator\Desktop\SysinternalsSuite\accesschk.exe	323448	0x20	ARCHIVE	2014-01-16T18:41:09Z	2014-01-16T18:41:09Z	2014-01-16T18:40:46Z	Sysinternals - www.sysinternals.com
JIMMY278F	C:\Documents and Settings\Administrator\Desktop\SysinternalsSuite\AccessEnum.exe	174968	0x20	ARCHIVE	2014-01-16T18:41:09Z	2014-01-16T18:41:09Z	2014-01-16T18:40:46Z	Sysinternals - www.sysinternals.com

Figure 16: Example DirList Module Output – Web Browser

Analysis - Next Steps

After completing the analysis discussed previously over the System, YARA, DirList, and PSList output, the analyst will likely want to leverage the newly discovered indicators of compromise to determine the extent of the issue and related malicious activity. Suggested next steps include:

- Lookup the MD5 and SHA256 hashes of the malware obtained from the DirList output against a repository of known hashes in an attempt to quickly identify the malware;
- Use the created and modified timestamps of the malware obtained from the DirList output to identify other suspicious files created or modified on the system (and all systems in the environment) around the same time;
- Use the MD5 and SHA256 hashes of the malware obtained from the DirList output to look for other copies, potentially with different filenames, across the enterprise;
- Using the DirList output and a defined time period of interest based off of other indicators, look for the creation of key files like NTUSER.dat or Desktop.ini that may indicate a user first appeared on a system during this time;
- Explore all available fields that were not covered here for the identified bad process from the PSList module and look for more potential indicators of compromise where values are atypical; and
- Explore all available fields that were not covered here for the identified malware from the DirList module and look for more potential indicators of compromise where values are atypical.

APPENDIX A

Complete List of PSList Values Captured by CrowdResponse		
accessed	id	peid
AddressOfEntryPoint	ImageBase	PEname
anomalies	importcount	productname
BaseOfCode	internalname	productversion
BaseOfData	legalcopyright	SectionAlignment
cert_comment	LoaderFlags	size
cert_exists	Machine	SizeOfCode
cert_result	MachineStr	SizeOfHeaders
cert_signer	MajorImageVersion	SizeOfHeapCommit
cert_verified	MajorLinkerVersion	SizeOfHeapReserve
Characteristics	MajorOperatingSystemVersion	SizeOfImage
CharacteristicsStr	MajorSubSystemVersion	SizeOfInitializedData
CheckSum	MemAddressOfEntryPoint	SizeOfOptionalHeader
cmdline	MinorImageVersion	SizeOfStackCommit
companyname	MinorLinkerVersion	SizeOfStackReserve
created	MinorOperatingSystemVersion	SizeOfUninitializedData
DllCharacteristics	MinorSubSystemVersion	Subsystem
DllCharacteristicsStr	modified	SubsystemStr
exportcount	name	system
FileAlignment	NumberOfSections	threadinjected
filedescription	NumberOfSymbols	TimeDateStamp
fileversion	originalfilename	Win32VersionValue

APPENDIX B

Complete List of DirList Values Captured by CrowdResponse		
accessed	companyname	name
attrhex	created	originalfilename
attrstr	filedescription	productname
cert_comment	fileversion	productversion
cert_exists	internalname	sha256
cert_result	legalcopyright	size
cert_signer	md5	system
cert_verified	modified	