Meet PyXie: A Nefarious New Python RAT

threatvector.cylance.com/en_us/home/meet-pyxie-a-nefarious-new-python-rat.html

Ryan Tracey

Introduction

BlackBerry Cylance researchers have recently discovered a previously unnamed Python RAT we're calling PyXie. PyXie has been observed in the wild since at least 2018 without much attention from the cybersecurity industry.

PyXie has been deployed in an ongoing campaign that targets a wide range of industries. It has been seen in conjunction with Cobalt Strike beacons as well as a downloader that has similarities to the *Shifu* banking Trojan. Analysts have observed evidence of the threat actors attempting to deliver ransomware to the healthcare and education industries with PyXie.

BlackBerry Cylance has conducted multiple incident response (IR) engagements in which PyXie was identified on hosts in the victim environment.

Key highlights of the PyXie campaign include:

- Legitimate LogMeIn and Google binaries used to sideload payloads.
- A Trojanized Tetris app to load and execute Cobalt Strike stagers from internal network shares.
- Use of a downloader with similarities to *Shifu* named "Cobalt Mode".
- Use of *Sharphound* to collect active directory information from victims.
- A custom compiled Python interpreter that uses scrambled opcodes to hinder analysis.
- Use of a modified RC4 algorithm to encrypt payloads with a unique key per infected host.

The main focus of this blog will be on PyXie RAT, but IOCs for other parts of the campaign will be listed in the Appendix.

Attack Overview

Figure 1: Loader overview

First Stage - Loader

This campaign uses a sideloading technique leveraging legitimate applications to load the first stage components of the malware.

We have observed two different variants of malicious loaders targeting popular applications which are likely to be found on most computers:

- LMIGuardianDll.dll side-loaded by a signed binary (LmiGuardianSvc.exe) from LogMeIn.
- Goopdate.dll side-loaded by a signed binary (GoogleUpdate.exe) from Google.

The list of the side-loaded DLLs analyzed, in chronological order by compilation timestamp, is presented in Table 1:

Filename	SHA256	Timestamp

N/A	e0f22863c84ee634b2650b322e6def6e5bb74460952f72556715272c6c18fe8e	3/17/17 12:23
GOOPDATE.dll	c9400b2fff71c401fe752aba967fa8e7009b64114c9c431e9e91ac39e8f79497	12/15/17 18:46
GOOPDATE.dll	814357417aa8a57e43d50cb3347c9d287b99955b0b8aee4e53e12b463f7441a0	1/11/18 22:30
GOOPDATE.dll	7330fa1ca4e40cdfea9492134636ef06cd999efb71f510074d185840ac16675d	6/4/18 20:44
GOOPDATE.dll	a765df03fffa343aa7a420a0a57d4b5c64366392ab6162c3561ff9f7b0ad5623	7/16/18 16:51
GOOPDATE.dll	de44656b4a3dde6e0acdc6f59f73114ce6bb6342bec0dcd45da8676d78b0042e	8/23/18 17:11
GOOPDATE.dll	5937746fc1a511d9a8404294b0caa2aedae2f86b5b5be8159385b6c7a4d6fb40	11/20/18 20:35
GOOPDATE.dll	56e96ce15ebd90c197a1638a91e8634dbc5b0b4d8ef28891dcf470ca28d08078	12/2/18 13:47
GOOPDATE.dll	1d970f2e7af9962ae6786c35fcd6bc48bb860e2c8ca74d3b81899c0d3a978b2b	12/8/18 20:38
GOOPDATE.dll	d271569d5557087aecc340bb570179b73265b29bed2e774d9a2403546c7dd5ff	12/10/18 14:26
GOOPDATE.dll	3a47e59c37dce42304b345a16ba6a3d78fc44b21c4d0e3a0332eee21f1d13845	1/9/19 16:58
GOOPDATE.dll	3aa746bb94acee94c86a34cb0b355317de8404c91de3f00b40e8257b80c64741	1/11/19 16:51
GOOPDATE.dll	f9290cd938d134a480b41d99ac2c5513a964de001602ed34c6383dfeb577b8f7	1/27/19 15:36
LMIGUARDIANDLL.DLL	c3b3f46a5c850971e1269d09870db755391dcbe575dc7976f90ccb1f3812d5ea	4/10/19 14:22
N/A	ea27862bd01ee8882817067f19df1e61edca7364ce649ae4d09e1a1cae14f7cc	4/10/19 14:22
GOOPDATE.dll	edd1480fe3d83dc4dc59992fc8436bc1f33bc065504dccf4b14670e9e2c57a89	4/11/19 13:08
LMIGUARDIANDLL.DLL	92a8b74cafa5eda3851cc494f26db70e5ef0259bc7926133902013e5d73fd285	6/19/19 14:29
LMIGUARDIANDLL.DLL	78471db16d7bd484932c8eb72f7001db510f4643b3449d71d637567911ca363b	8/14/19 16:29

Table 1: Malicious side-loaded DLLs

Once loaded by the LogMeIn or Google binary, the malicious DLL will locate its corresponding encrypted payload. It does this by taking the full path it was loaded from and appending a *.dat* extension to it.

For example, if the malicious DLL is named *LmiGuardianDLL.dll*, the payload filename would be *LmiGuardianDLL.dll.dat*.

The encrypted payload is read from disk and then *AES-128* decrypted in *CBC* mode by the loader. The 16-byte initialization vector (IV) and symmetric key are hardcoded into the DLL and can vary from sample to sample. An example of a loader using hardcoded IV and Key is shown in Figure 2:

Figure 2: Payload hardcoded decryption details

The result of the decryption is the second stage payload, which is mapped into the loader process address space and executed.

Second Stage – Installation and Persistence

The second stage malware is primarily responsible for installing itself, setting up persistence, and spawning a new process to inject the third stage payload. The second stage component fingerprints the victim machine by generating a Hardware ID hash. This hash is later used as a seed for various functions as well as an encryption key in the subsequent stages.

Hardware ID

The Hardware ID is generated by calculating the MD5 hash of an 80-byte buffer containing information collected from the system, including:

- A 64-byte zero-padded Process Brand string obtained through a sequence of calls to the *CPUID* instruction (with EAX set to *0x80000002*, *0x80000003* and *0x80000004* values).
- The output values of *dwNumberOfProcessors* and *dwProcessorType* returned by a call to *GetSystemInfo.*
- The volume serial number of the Operating System logical drive obtained through a call to *GetVolumeInformationA*.
- The CRC32 hash of the current user obtained through a call to *GetUserNameA* using the ZLIB-CRC32 implementation with a seed of -1 (0xFFFFFFFF).

Figure 3: Buffer used to compute the Hardware ID

Given the buffer in Figure 3, the Hardware ID would be:
66aafbed7c63b1a1d02899969d97e06f.

Mutexes

Two mutexes are created to prevent multiple payload instances from running at the same time.

The first mutex name is generated using the following algorithm:

- The process infected with the second stage payload is obtained by calling the *GetModuleFileNameA* API, and the CRC32 of the corresponding binary is computed.
- The computed hash is XORed with the CRC32 of the currently logged-on user.
- The hex string representation of the last hash represents the mutex name.

The second mutex is the last 28 characters of the hex representation of Hardware ID MD5 hash:

Figure 4: Mutex generated from the Hardware ID

Privilege Escalation

If the process infected with the second stage payload is running with administrator privileges, the malware will attempt to escalate its own privileges. It does so by creating and starting a temporary service, thus respawning and running as a LOCAL SYSTEM process. To remain stealthy, the malware deletes the temporary service from the Service Control Manager.

Installation

The loader and its corresponding payload are copied to a subdirectory within the **%APPDATA%** folder. The directory is chosen from the list of strings in Figure 5. Selection is made by using the CRC32 number of the Hardware ID string modulo 21(0x15) result as the index in the list:

Wireshark	WinRAR	VisualAssist
UltraVNC	TortoiseSVN	TeamViewer
Subversion	RoboForm	Notepad++
Mozilla	Macromedia	KeePass
JGsoft	Identities	Apple Computer

AnyDesk	Microsoft FxCop	Microsoft Corporation
Microsoft Visual Studio		

Figure 5: Installation targeted APPDATA directories

Persistence

Persistence is achieved by creating a registry value whose name is the hex string representation of a dynamically generated DWORD under *HKCU\Software\Microsoft\Windows\CurrentVersion\Run* registry key. The registry value is set to point to the path of the loader executable.

Injection of The Next Stage Payload

The third stage payload is decompressed with a call to *RtlDecompressBuffer* API and subsequently injected into a newly spawned process.

The executable targeted for injection is chosen by enumerating over the executables in the **%SYSTEMROOT%**\System32 directory and searching for the first one that meets the following criteria:

- It is not running
- It has a GUI subsystem
- It has “Microsoft” in the version information
- It is signed

The new process is spawned by using the *CreateProcessA* API and passing the path to the executable that loaded the seconded stage as the *lpCommandLine* parameter.

Third Stage - “Cobalt Mode” Downloader

The third stage is a downloader that has been designated *Cobalt Mode* based on the debug information left in some of the analyzed samples. An example of the PDB path is shown in Figure 6:

Z:\coding\pyproject\compiled\cobalt_mode\cobalt_mode.pdb

Figure 6: Third stage debug information

The primary function of Cobalt Mode is:

- Connecting to a command and control (C&C) server
- Downloading an encrypted payload
- Decrypting the payload
- Mapping and executing the payload in the address space of the current process
- Spawning a new process for code injection

Cobalt Mode malware can carry out a series of environmental checks. It can determine if it is being run from a sandbox or virtual machine (VM), if a smart card reader is attached, and if requests are being intercepted with a man-in-the-middle (MitM) attack.

Examining the functions and values used to make these checks revealed code overlap with the *Shifu* banking Trojan.

The configuration for *Cobalt Mode* is stored within the malware as a compressed JSON blob. A decompressed copy of the config extracted from our sample is shown below:

```
{
  "user_agent": "Mozilla/4.0 (compatible; MSIE 2.1; Windows NT 5.0; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729;
.NET CLR 3.0.30729; Media Center PC 6.0)",
"check_extra_locale": 1,
  "mitm_host2": "twitter[.]com",
  "check_mitm": 0,
  "mitm_host1": "google[.]com",
  "referer": "hxxps://www.google[.]com",
  "download_url": "hxxps://tedxns[.]com/api/get_file/c2b469a7-d628-4804-8cca-5b734c5c6b42?hw=",
  "check_vm": 0,
  "extra_locale": "Ukraine"
}
```


Figure 7: CobaltMode configuration JSON

A request to the *download_url* specified in the JSON config is made to retrieve the next stage payload. The same Hardware ID discussed in the second stage analysis is passed in the request as the *hw* parameter. The server uses this value as a key to encrypt a 7zip archive it returns with a modified RC4 algorithm. The archive contains a custom Python interpreter and a zip file containing the PyXie RAT bytecode that will be discussed later in our analysis.

Modified RC4 Algorithm

The modified RC4 algorithm used by the malware differs from regular RC4 in the way it generates its Substitution Box (S-BOX). After the S-BOX is initialized, the key is XORed against the S-BOX rather than using it to swap values.

Injection

The same algorithm as in the previous stage is used to select an executable in the ***%SYSTEMROOT%\system32*** directory, then spawn a new process for injection. This time ***%SYSTEMROOT%\system32\worker.exe*** is used as the *lpCommandLine* parameter for the call to *CreateProcessA* API:

Figure 8: Third stage spawned process

PyXie RAT

The final stage payload is a full-featured Python RAT compiled into an executable. Rather than using Py2Exe or PyInstaller to create an executable, the malware authors compiled their own Python interpreter that loads an archive containing the PyXie RAT bytecode from memory.

PyXie RAT functionality includes:

Functionality
Man-in-the-middle (MITM) Interception
Web-injects
Keylogging
Credential harvesting
Network Scanning
Cookie theft
Clearing logs
Recording video
Running arbitrary payloads

Monitoring USB drives and exfiltrating data
WebDav server
Socks5 proxy
Virtual Network Connection (VNC)
Certificate theft
Inventorying software
Enumerating the domain with Sharphound

Table 2: PyXie RAT functionality

The custom interpreter is bundled with an obscure library named memzipimport which imports a zip file containing the compiled RAT bytecode directly from memory. The zip contains over 1,500 files of Python bytecode, many of which are third party libraries.

The core RAT consists of approximately 79 bytecode files. A list of these core files can be found in Table 3:

```
/core/__init__.pyx
/core/active_host.pyx
/core/backdoor.pyx
/core/beacon.pyx
/core/commands.pyx
/core/conf/__init__.pyx
/core/conf/config.pyx
/core/debug.pyx
/core/destruct.pyx
/core/entry_point.pyx
/core/initialize.pyx
/core/install.pyx
/core/ipc/__init__.pyx
/core/ipc/exclude.pyx
/core/ipc/ipc.pyx
/core/ipc/mimikatz.pyx
/core/keylog.pyx
/core/mitm/__init__.pyx
/core/mitm/cert_gen.pyx
/core/mitm/proxy.pyx
/core/mitm/web_dump.pyx
/core/mitm/web_fakes.pyx
/core/mitm/web_injects.pyx
/core/mitm/web_screens.pyx
/core/modules/__init__.pyx
/core/modules/aes_cfc.pyx
/core/modules/bot_lib.pyx
/core/modules/cookies.pyx
/core/modules/crc64.pyx
/core/modules/decorators.pyx
/core/modules/description.pyx
/core/modules/ffmpeg_inst.pyx
/core/modules/ffmpeg_rec.pyx
/core/modules/find_files.pyx
/core/modules/keepass.pyx
/core/modules/lnk_file.pyx
/core/modules/logmein.pyx
/core/modules/multipart.pyx
/core/modules/os_ver.pyx
/core/modules/rdp.pyx
/core/modules/rdp_creds.pyx
/core/modules/recent_files.pyx
/core/modules/research_domain.pyx
/core/modules/sharphound.pyx
/core/modules/smb_scan.pyx
/core/modules/socks5.pyx
```

/core/modules/sysinfo.pyx
/core/modules/tools.pyx
/core/modules/webdav.pyx
/core/modules/winapi_stubs.pyx
/core/modules/windnsquery.pyx
/core/modules/winfiletime.pyx
/core/nmc.pyx
/core/obfuscate/__init__.pyx
/core/obfuscate/boolean_obfuscator.pyx
/core/obfuscate/number_obfuscator.pyx
/core/obfuscate/obfuscate.pyx
/core/obfuscate/string_obfuscator.pyx
/core/passwords.pyx
/core/protect.pyx
/core/pwnage.pyx
/core/software.pyx
/core/systems.pyx
/core/transport/__init__.pyx
/core/transport/dns.pyx
/core/transport/github.pyx
/core/transport/google.pyx
/core/transport/i2p.pyx
/core/transport/slack.pyx
/core/transport/tcp.pyx
/core/transport/tor.pyx
/core/transport/twitter.pyx
/core/transport/udp.pyx
/core/transport/xmpp.pyx
/core/tun/__init__.pyx
/core/tun/client.pyx
/core/tun/util.pyx
/core/usbmon.pyx
/core/zip_logs.pyx

Table 3: RAT bytecode files

Analysis of .pyx files

At first glance, the *.pyx* files appear to be renamed *.pyc* files. However, after attempting to decompile them with uncompyle2 researchers recognized that this was not the case:

Figure 9: Uncompyle2 error

Upon closer examination, we discovered these *.pyx* files have had the first 8 bytes stripped. These bytes typically would contain a 4-byte magic number, which denotes the version of the Python that generated them along with a 4-byte timestamp. The removal of these bytes is a well-known copy-protection trick aimed at preventing some tools from decompiling the files and getting access to the code.

The workaround for this is simple. All that is needed is to add the appropriate magic number and a timestamp. A quick look at the strings from the malicious interpreter indicates that it may be based on Python 2.7.15:

```
z:\coding\pyproject\python_static_2.7.15\pyuv-1.x\src\common.c
z:\coding\pyproject\python_static_2.7.15\pyuv-1.x\deps\libuv\src\win\handle-inl.h
z:\coding\pyproject\python_static_2.7.15\pyuv-1.x\deps\libuv\src\win\req-inl.h
```

Figure 10: Strings identifying the Python version

Once the appropriate header for Python 2.7.15 was added along with an arbitrary timestamp, researchers attempted to decompile the *.pyx* bytecode files again. This worked better than our first attempt, but something still wasn't right. Disassembly still failed when using a clean install of Python 2.7.15:

Figure 11: Disassembler error

Our exercise led researchers to believe that the interpreter may have remapped opcodes. This would prevent other interpreters from correctly parsing the bytecode generated by it.

Remapping opcodes is not new. It is a technique often used by commercial applications such as Dropbox. The workaround for this is well known and documented in '*pyREtic´ – In memory reverse engineering for obfuscated Python bytecode* and *Looking inside the (Drop) box*. The basic premise covers compiling the same Python script in both a standard and modified interpreter. The generated bytecode can then be compared to derive the remapped opcodes. This is typically straightforward if arbitrary code can be executed in the modified interpreter.

Unfortunately, PyXie is not a friendly Python interpreter and does not provide a mechanism to trivially run arbitrary code. Our solution was to analyze the modified interpreter in IDA Pro and compare it to the Python 2.7.15 source code. By using strings for reference, it was possible to locate the native *PyRun_SimpleStringFlags()* function. This function can be used to run Python code from an in-memory string.

Once the location of *PyRun_SimpleStringFlags()* was found, a debugger was used to attach to the process running the PyXie interpreter.

Prior to redirecting the execution to a code cave, the script we wanted compiled is dropped in the working directory. For this controlled execution we used all.py from DeDrop. Then:

- Write the “import all.py” string into the memory.
- Push the address of the “import all.py” string to the stack.
- Call *PyRun_SimpleStringFlags()* which caused the interpreter to execute *all.py* and in turn emit compiled bytecode to *all.pyx.*

Figure 12: Code cave importing all.py

The *all.py* file was then executed within the standard Python 2.7.15 interpreter to output compiled bytecode as *all.pyc*. The bytecode in *all.pyx* was compared to bytecode in *all.pyc* to determine the remapped opcodes. This comparison was used to generate a modified version of *opcode.py* which could be used to disassemble and decompile the PyXie RAT bytecode:

Mnemonic	Python 2.7 Opcode	PyXie RAT Opcode
PRINT_ITEM	0x47	0x58
PRINT_NEWLINE	0x48	0x3e
POP_TOP	0x1	0xd
RETURN_VALUE	0x53	0x19
ROT_TWO	0x2	0x43
ROT_THREE	0x3	0x50
STORE_MAP	0x36	0x3c
INPLACE_ADD	0x37	0x3d
ROT_FOUR	0x5	0x4e
UNARY_POSITIVE	0xa	0x39
UNARY_NEGATIVE	0xb	0x52
UNARY_NOT	0xc	0x41
UNARY_CONVERT	0xd	0x4a
UNARY_INVERT	0xf	0x3a

GET_ITER	0x44	0x1d
BINARY_MULTIPLY	0x14	0x53
BINARY_POWER	0x13	0x4c
BINARY_DIVIDE	0x15	0x16
BINARY_MODULO	0x16	0xa
BINARY_ADD	0x17	0x4
BINARY_SUBTRACT	0x18	0x46
BINARY_SUBSCR	0x19	0x1b
BINARY_FLOOR_DIVIDE	0x1a	0x14
INPLACE_FLOOR_DIVIDE	0x1c	0x59
INPLACE_DIVIDE	0x3a	0x47
INPLACE_SUBTRACT	0x38	0xf
INPLACE_MULTIPLY	0x39	0x3
STORE_SUBSCR	0x3c	0xc
DELETE_SUBSCR	0x3d	0x44
BINARY_LSHIFT	0x3e	0x13
BINARY_RSHIFT	0x3f	0x54
BINARY_AND	0x40	0x17
BINARY_XOR	0x41	0x1c
BINARY_OR	0x42	0x37
INPLACE_POWER	0x43	0x57
POP_BLOCK	0x57	0x48
DUP_TOP	0x4	0x4f
PRINT_ITEM_TO	0x49	0x42
PRINT_NEWLINE_TO	0x4a	0x1
INPLACE_LSHIFT	0x4b	0x1a

INPLACE_RSHIFT	0x4c	0x4b
INPLACE_AND	0x4d	0x56
INPLACE_XOR	0x4e	0x38
INPLACE_OR	0x4f	0xb
BREAK_LOOP	0x50	0x15
WITH_CLEANUP	0x51	0x40
END_FINALLY	0x58	0x55
BUILD_CLASS	0x59	0x2
EXEC_STMT	0x55	0x36
LOAD_LOCALS	0x52	0x3f
IMPORT_STAR	0x54	0x4d
YIELD_VALUE	0x56	0x51

Table 4: Remapped Opcodes

Once decompiled, one last layer of obfuscation remained. A majority of the significant strings were encoded with the ZLIB codec:

Figure 13: ZLIB encoded strings

C&C Communication

The version of PyXie RAT we analyzed can communicate with the command and control (C&C) server over HTTP/HTTPS as well as via comments left in GitHub Gist. Indicators in the code lead researchers to believe that the malware authors may also be planning to add the following C&C channels:

Figure 14: C&C communication channels

C&C domains can be resolved internally via DNS servers specified in the config or by utilizing the servers from the OpenNIC Project.

Recent samples we collected are configured to connect back to a subset of the following domains:

```
tedxns[.]com
benreat[.]com
planlamaison[.]com
sarymar[.]com
teamchuan[.]com
c1oudflare[.]com
```

Figure 15: C&C domains

Earlier samples utilize Namecoin ".bit" domains:

```
athery[.]bit
babloom[.]bit
floppys[.]bit
```

Figure 16: Namecoin .bit C&C domains

An example of a C&C request is shown in Figure 17:

Figure 17: C&C sample request

Commands

PyXie RAT currently supports the following commands:

Command	Description
!load	Download and run an executable
!get_config	Retrieve current config
!set_config	Set config
!update	Update
!update2	Update
!update3	Update
!get_keylog	Retrieve keylog
!get_cookies	Retrieve cookies
!get_sysinfo	Retrieve system info
!scan_lan	SMB scan local network
!scan_lan_ex	SMB scan specified IP ranges
!webdav	Start WebDAV server
!webdav_stop	Stop WebDAV server
!active_sk	Start SOCKS5 server
!deactive_sk	Stop SOCKS5 server
!active_bc	Start HVNC module
!deactive_bc	Stop HVNC module
!eval	Download and execute Python code
!self_destruct	Uninstall RAT
!get_screens	Retrieve Screenshots

!mem_load	Download and execute DLL in memory
!shellcode	Download and execute shellcode
!get_passwords	Dump passwords with LaZagne
!docfind	Retrieve file
!filefind	Find files matching certain criteria
!del_cookies	Clear cookies
!export_certs	Retrieve certificates from certificate store
!del_keylog	Clear keylog
!reboot	Reboot system
!check_soft	Check for installed software
!install_ffmpeg	Download ffmpeg binaries
!record_video	Record video with ffmpeg
!shell	Run command and capture output
!kill_lgmn_tokens	Retrieve LogMeIn credentials
!get_lgmn_tokens	Clear LogMeIn credentials
!sharphound	Enumerate domain with Sharphound
!bot_hashes	Retrieves hashes of loader and DLL
!mimi_32	Download Mimikatz
!mimi_64	Download Mimikatz
!mimi_grab	Execute Mimikatz
!get_kdbx	Retrieve keepass databases
!research_domain	SMB scan of computers identified by Sharphound
!research_full	SMB scan and port scan of computers identified by Sharphound
!wipe_rdp_creds	Clear RDP creds

Table 5: RAT commands

Configuration

The configuration for PyXie RAT is stored as an encrypted JSON blob. Fernet algorithm is used for the encryption and the symmetric key is base64 encoded and hardcoded in *config.pyx* as the *internal_key* variable. A full sample of the decrypted configuration JSON can be found in the Appendix.

For the majority of samples analyzed, the following *internal_key* value was used:

eJxLsnDOzElOLyrOsvTKrrQsNUspNoxISfMxDE1y043IydGtMAhyNMkNMy/OtgUAVu0OyQ==

Figure 18: Configuration snippet

Tool.exe – Trojanized Tetris Game

Our researchers have also observed PyXie RAT being deployed by and in conjunction with Cobalt Strike and a custom shellcode loader. We've also seen evidence of this loader being used in several Ransomware incidents.

The loader is a Trojanized open source Tetris game. It has been modified to load an encrypted shellcode payload named "*settings.dat*" from an internal network share and inject it into a new process:

Figure 19: Trojanized Tetris Game

The payload is decrypted using a single-byte XOR with the key 0xFA, and it is injected into a new process using the same injection routine previously described in our analysis of the Second Stage – Installation and Persistence and Third Stage - "Cobalt Mode" Downloader components:

Figure 20: Payload being loaded from a network share and decrypted

In the cases that we have observed, the encrypted payloads have been Cobalt Strike stagers that connect back to one of the servers listed below:

fearlesslyhuman[.]org
185[.]82[.]202[.]109
foods-pro[.]com
dopearos[.]com

Figure 21: Cobalt Strike servers

Conclusion

PyXie RAT has been quietly lurking in systems since at least 2018. It has been observed targeting a wide range of industries while successfully maintaining a low profile.

The PyXie RAT developers take a number of steps to obfuscate key components. In our analysis, we covered the different stages of the malware and shared the techniques we used to analyze the custom Python interpreter with remapped opcodes. We also provided a number of indicators that can help identify an infection.

BlackBerry Cylance uses artificial intelligence-based agents trained for threat detection on millions of both safe and unsafe files. Our automated security agents block PyXie RAT based on countless file attributes and malicious behaviors instead of relying on a specific file signature. BlackBerry Cylance, which offers a predictive advantage over zero-day threats, is trained on and effective against both new and legacy cyberattacks. For more information, visit https://www.cylance.com.

Appendix

Indicators of Compromise (IOCs)

Indicator	Type	Description

1d970f2e7af9962ae6786c35fcd6bc48bb860e2c8ca74d3b81899c0d3a978b2b	SHA256	Loader DLL
3a47e59c37dce42304b345a16ba6a3d78fc44b21c4d0e3a0332eee21f1d13845	SHA256	Loader DLL
3aa746bb94acee94c86a34cb0b355317de8404c91de3f00b40e8257b80c64741	SHA256	Loader DLL
56e96ce15ebd90c197a1638a91e8634dbc5b0b4d8ef28891dcf470ca28d08078	SHA256	Loader DLL
5937746fc1a511d9a8404294b0caa2aedae2f86b5b5be8159385b6c7a4d6fb40	SHA256	Loader DLL
7330fa1ca4e40cdfea9492134636ef06cd999efb71f510074d185840ac16675d	SHA256	Loader DLL
78471db16d7bd484932c8eb72f7001db510f4643b3449d71d637567911ca363b	SHA256	Loader DLL
814357417aa8a57e43d50cb3347c9d287b99955b0b8aee4e53e12b463f7441a0	SHA256	Loader DLL
92a8b74cafa5eda3851cc494f26db70e5ef0259bc7926133902013e5d73fd285	SHA256	Loader DLL
a765df03fffa343aa7a420a0a57d4b5c64366392ab6162c3561ff9f7b0ad5623	SHA256	Loader DLL
c3b3f46a5c850971e1269d09870db755391dcbe575dc7976f90ccb1f3812d5ea	SHA256	Loader DLL
c9400b2fff71c401fe752aba967fa8e7009b64114c9c431e9e91ac39e8f79497	SHA256	Loader DLL
d271569d5557087aecc340bb570179b73265b29bed2e774d9a2403546c7dd5ff	SHA256	Loader DLL
de44656b4a3dde6e0acdc6f59f73114ce6bb6342bec0dcd45da8676d78b0042e	SHA256	Loader DLL
e0f22863c84ee634b2650b322e6def6e5bb74460952f72556715272c6c18fe8e	SHA256	Loader DLL
ea27862bd01ee8882817067f19df1e61edca7364ce649ae4d09e1a1cae14f7cc	SHA256	Loader DLL
edd1480fe3d83dc4dc59992fc8436bc1f33bc065504dccf4b14670e9e2c57a89	SHA256	Loader DLL
f9290cd938d134a480b41d99ac2c5513a964de001602ed34c6383dfeb577b8f7	SHA256	Loader DLL
366d47b95e216863ee64e0024e2bbf0bf1b66420986fe0a5b3e805ce795dcf9f	SHA256	Encrypted Payload
d031081b8c211994b5406bf3f2544c0d6ebcbab384f23e393f084b49563e1d12	SHA256	Encrypted Payload
f466bc20544bf203155142cf14456e55b0e756aa93ecfb5edc74ba7ed60f9573	SHA256	Encrypted Payload
ca68f02bd01650383af68f0c129482faf283329dd1e6a18821ad26fc2c3d00b2	SHA256	Encrypted Payload
d776235e628422ada7f1e976a3cf771049286edf2219583028fbbd6229af72b9	SHA256	Encrypted Payload
50a4b19b38caea4eea042704314f5ae1acf2162c7353fb92bc896dcada14b86a	SHA256	Encrypted Payload
610c3536ceafc0e4ad0d60c683052ee7272e29049ceac909b1d1e55ac1206f49	SHA256	Encrypted Payload
7ee6235f0e653a36a818a12531657f6dac5f3fb41efa1e1c63f6761ba3faeb90	SHA256	Encrypted Payload

265e5e1389b3145bf2ac1a017b67a54d84bc361dc3795120656dcabc1212c34a	SHA256	Encrypted Payload
8d2b3b0cbb32618b86ec362acd142177f5890917ae384cb58bd64f61255e9c7f	SHA256	PyXie RAT interpreter
d1429f54baaad423a8596140a3f70f7d9f762373ad625bda730051929463847d	SHA256	PyXie RAT bytecode
ade8f07bf7918343bf307ec35837327efc7a85a0edac5ab5b2cd037134af8d57	SHA256	Cobalt Mode
fd93858f4e7356bebe30dd0dfe07367e3ddf6164bb78725e1c543b093558cf64	SHA256	Cobalt Strike Loader
a50b58e24eb261157c4f85d02412d80911abe8501b011493c7b393c1905fc234	SHA256	Cobalt Strike Loader
0d14a1b5574dc12f6286d37d0a624232fb63079416b98c2e1cb5c61f8c2b66ff	SHA256	Cobalt Strike Loader
625c22b21277c8a7e1b701da9c1c21b64bfa02baef5d7a530a38f6d70a7a16d0	SHA256	Cobalt Strike Loader
bd7da341a28a19618b53e649a27740dfeac13444ce0e0d505704b56335cc55bd	SHA256	Cobalt Strike Loader
d612144c1f6d4a063530ba5bfae7ef4e4ae134bc55dcf067439471934b841b00	SHA256	Cobalt Strike Loader
ce0936366976f07ea24e86733888e97e421393829ecfd0fde66bd943d4b992ab	SHA256	Cobalt Strike Loader
3259dd0efed1d28a149d4e8c4f980a19199d9bead951ee1231e3a26521185f2f	SHA256	Cobalt Strike Loader
e5fede5eb43732c7f098acf7b68b1350c6524962215b476de571819b6e5a71fc	SHA256	Cobalt Strike Loader
f6ff873e1bd3d0e6b6182792aebd781f4f60be39d49085ba3d64658456260402	SHA256	Cobalt Strike Loader
608f34a79e5566593b284ef0d24f48ea89bc007e5654ae0969e6d9f92ec87d32	SHA256	Cobalt Strike Loader
b1f54b88c9b7680877981f6bebde6aea9effbc38a0a8b27a565fb35331094680	SHA256	Cobalt Strike Loader
d50f28cf5012e1ffde1cd28655e07519dadcf94218b15c701c526ab0f6acb915	SHA256	Cobalt Strike Loader
56934547dcf0d7ecf61868ae2f620f60e94c094dbd5c3b5aaf3d3a904d20a693	SHA256	Cobalt Strike Loader
73609f8ebd14c6970d9162ec8d7786f5264e910573dff73881f85b03163bd40e	SHA256	Cobalt Strike Loader
2ceb5de547ad250140c7eb3c3d73e4331c94cf5a472e2806f93bf0d9df09d886	SHA256	Cobalt Strike Loader
840985b782648d57de302936257ba3d537d21616cb81f9dce000eaf1f76a56c8	SHA256	Cobalt Strike Loader
e48e88542ec4cd6f1aa794abc846f336822b1104557c0dfe67cff63e5231c367	SHA256	Cobalt Strike Loader
cb2619b7aab52d612012386d88a0d983c270d9346169b75d2a55010564efc55c	SHA256	Cobalt Strike Loader
88565b4c707230eac34d4528205056264cd70d797b6b4eb7d891821b00187a69	SHA256	Cobalt Strike Loader
91c62841844bde653e0357193a881a42c0bc9fcc798a69f451511c6e4c46fd18	SHA256	Cobalt Strike Loader
ddf83c02effea8ae9ec2c833bf40187bed23ec33c6b828af49632ef98004ea82	SHA256	Cobalt Strike Loader

edecfdd2a26b4579ecacf453b9dff073233fb66d53c498632464bca8b3084dc5	SHA256	Cobalt Strike Loader
sarymar[.]com	Network	PyXie RAT C&C
benreat[.]com	Network	PyXie RAT C&C
planlamaison[.]com	Network	PyXie RAT C&C
teamchuan[.]com	Network	PyXie RAT C&C
tedxns[.]com	Network	PyXie RAT / Cobalt Mode C&C
athery[.]bit	Network	PyXie RAT C&C
babloom[.]bit	Network	PyXie RAT C&C
floppys.bit	Network	PyXie RAT C&C
104[.]200[.]67[.]173	Network	PyXie RAT C&C
hwartless.bit	Network	Cobalt Mode C&C
c1oudflare[.]com	Network	Cobalt Mode C&C
foods-pro[.]com	Network	Cobalt Strike C&C
dopearos[.]com	Network	Cobalt Strike C&C
fearlesslyhuman[.]org	Network	Cobalt Strike C&C
185.82.202.109	Network	Cobalt Strike C&C
192[.]52[.]167[.]241	Network	Seen hosting malicious Loader DLL
ololo[.]space	Network	Seen hosting malicious Loader DLL
%Appdata%\Wireshark\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\WinRAR\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\VisualAssist\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\UltraVNC\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\TortoiseSVN\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\TeamViewer\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory

%Appdata%\Subversion\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\RoboForm\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\Notepad++\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\Mozilla\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\Macromedia\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\KeePass\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\JGsoft\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\Identities\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\Apple Computer\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\AnyDesk\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\Microsoft FxCop\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\Microsoft Corporation\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
%Appdata%\Microsoft Visual Studio\	File	Presence of Goopdate.dll or LmiGuardianDLL.dll in this directory
\SystemRoot\system32\worker.exe	Process	Command line argument

Yara

```
rule PyXie_RAT
{
    meta:
        description = "Detects PyXie RAT"

    strings:
        $mz = "MZ"
        $op = {C6 06 68 89 46 01 C7 46 05 9C 81 74 24 C6 46 09 04 89 4E 0A 66 C7 46 0E 9D C3}

    condition:
        ($mz at 0) and $op
}
```

Sample PyXie RAT Config

```
{
   "ap": {
      "cmd_ok_fmt": "_id=%s&res_code=%s&res_text=%s",
      "cmd_resp_uri": "/api/profile",
      "cn": "5hsts",
      "debug_active": 1,
      "debug_timeout": 60,
      "domain_key": "-----BEGIN PUBLIC KEY-----
\nMEkwEwYHKoZIzj0CAQYIKoZIzj0DAQEDMgAEIZ4IqMRf8jX1Out08jed9oYIT9hQ\nqDMN3qCaGp43ITWdu780MJ2rwgrYAB1StyhZ\n-
----END PUBLIC KEY-----\n",
      "domain_key_uri": "/api/users",
      "extra_beacon_port": 50105,
      "hosts": "tedxns[.]com,benreat[.]com,planlamaison[.]com,sarymar[.]com,teamchuan[.]com",
      "knock_jitter": 20,
      "knock_timeout": 300,
      "knock_uri": "/api/userlogin",
      "logs_uri": "/api/imageupload",
      "post_log_fmt": "type=post&name=%s&url=%s&user_agent=%s&process=%s&referer=%s&keylog=%s&data=%s",
      "referer": "hxxps://www[.]google[.]com",
      "secure": 1,
      "user_agent": "Mozilla/4.0 (compatible; MSIE 2.1; Windows NT 5.0; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR
3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0)",
      "zips_uri": "/api/userdata"
   },
   "backdoor": {
      "enabled": 0
   },
   "dirs": [
      "%APPDATA%\\Agama",
      "%APPDATA%\\Armory",
      "%APPDATA%\\B3-CoinV2",
      "%APPDATA%\\BeerMoney",
      "%APPDATA%\\Bitcloud",
      "%APPDATA%\\Bitcoin",
      "%APPDATA%\\BitcoinZ",
      "%APPDATA%\\bitconnect",
      "%APPDATA%\\Bither",
      "%APPDATA%\\bitmonero",
      "%APPDATA%\\BlocknetDX",
      "%APPDATA%\\Cybroscoin",
      "%APPDATA%\\Daedalus",
      "%APPDATA%\\DashCore",
      "%APPDATA%\\DeepOnion",
      "%APPDATA%\\DigiByte",
      "%APPDATA%\\Dogecoin",
      "%APPDATA%\\ElectronCash",
      "%APPDATA%\\Electrum",
      "%APPDATA%\\Electrum-LTC",
      "%APPDATA%\\Ember",
      "%APPDATA%\\EmeraldWallet",
      "%APPDATA%\\Ethereum Wallet",
      "%APPDATA%\\Exodus",
      "%APPDATA%\\FairCoin",
      "%APPDATA%\\faircoin2",
      "%APPDATA%\\Florincoin",
      "%APPDATA%\\FORT",
      "%APPDATA%\\GambitCoin",
      "%APPDATA%\\GeyserCoin",
      "%APPDATA%\\GreenCoinV2",
      "%APPDATA%\\GridcoinResearch",
      "%APPDATA%\\Gulden",
```

```
        "%APPDATA%\\Hush",
        "%APPDATA%\\IOTA Wallet",
        "%APPDATA%\\Komodo",
        "%APPDATA%\\Learncoin",
        "%APPDATA%\\lisk-nano",
        "%APPDATA%\\Litecoin",
        "%APPDATA%\\Minexcoin",
        "%APPDATA%\\mSIGNA_Bitcoin",
        "%APPDATA%\\MultiBitHD",
        "%APPDATA%\\MultiDoge",
        "%APPDATA%\\Neon",
        "%APPDATA%\\NXT",
        "%APPDATA%\\Parity",
        "%APPDATA%\\Particl",
        "%APPDATA%\\Peercoin",
        "%APPDATA%\\pink2",
        "%APPDATA%\\PPCoin",
        "%APPDATA%\\Qtum",
        "%APPDATA%\\RainbowGoldCoin",
        "%APPDATA%\\RoboForm",
        "%APPDATA%\\StartCOIN-v2",
        "%APPDATA%\\straks",
        "%APPDATA%\\Stratis",
        "%APPDATA%\\TREZOR Bridge",
        "%APPDATA%\\TrumpCoinV2",
        "%APPDATA%\\VeriCoin",
        "%APPDATA%\\Verium",
        "%APPDATA%\\Viacoin",
        "%APPDATA%\\VivoCore",
        "%APPDATA%\\Xeth",
        "%APPDATA%\\Zcash",
        "%APPDATA%\\ZcashParams",
        "%APPDATA%\\Zetacoin",
        "%APPDATA%\\StratisNode",
        "%PROGRAMDATA%\\electroneum",
        "%PROGRAMDATA%\\bitmonero",
        "%LOCALAPPDATA%\\bisq",
        "%LOCALAPPDATA%\\copay",
        "%LOCALAPPDATA%\\programs\\zap-desktop",
        "%LOCALAPPDATA%\\RippleAdminConsole",
        "%LOCALAPPDATA%\\StellarWallet",
        "%ALLDRIVESROOTS%\\Alliance"
    ],
    "dirs_keys": [
        "coin",
        "wallet",
        "diebold",
        "altaro",
        "unitrends",
        "wincor",
        "magtek",
        "payment",
        "ncr",
        "replication",
        "bitmessage",
        "veeam",
        "backup",
        "filemaker",
        "back-up",
        "swift",
        "screenconnect",
```

```
    "aldelo",
    "bank",
    "passw",
    "avamar"
],
"ffmpeg": {
    "command": "ffmpeg.exe -f gdigrab -i desktop -pix_fmt yuv420p -threads 2 -c:v libvpx-vp9 -crf 40 -b:v 0 -speed 5",
    "timeout": 60
},
"keylog": {
    "date_format": "%H:%M:%S-%d:%b:%Y",
    "format": "\n\n[%s (%s) - %s]\n"
},
"mitm": {
    "enabled": 1,
    "exclusion": [
        "cc0141[.]bizsol[.]anser[.]ne[.]jp",
        "wupos[.]westernunion",
        "xpressmoney[.]biz",
        "webpos[.]epayworldwide[.]com",
        "cc[.]b-direct[.]saitamaresona[.]co[.]jp",
        "cc0181[.]eb[.]shinwabank[.]co[.]jp",
        "cc0001[.]b-web[.]mizuhobank[.]co[.]jp",
        "maza[.]cc",
        "light[.]webmoney[.]ru",
        "light[.]wmtransfer[.]com",
        "business24[.]cz",
        "certificate[.]us[.]army[.]mil"
    ],
    "get_as_post_marker": "&extra_flag_51783=",
    "hr_marker": "/automation_13111949/",
    "post_log_limit": 102400,
    "screens": [
        "blvlva[.]secure[.]fundsxpress[.]com|passcode",
        "cibng[.]ibanking-services[.]com/EamWeb/Remote/RemoteLoginAPI[.]aspx|_textBoxCompanyId,_textBoxUserId",
        "cityntl[.]webcashmgmt[.]com/wcmfd/wcmpw/CustomerLogin|organizationid,userid,password",
        "client[.]schwab[.]com/Login/SignOn/SignOn[.]ashx|txtPassword",
        "connect[.]secure[.]wellsfargo[.]com/auth/login/do|j_password",
        "express[.]53[.]com/portal/auth/login/Login|username,password",
        "login[.]morganstanleyclientserv[.]com/msologin/handler/proxy/auth/authenticate|Userid,Password",
        "[.]chase[.]com/auth/fcc/login|auth_userId,auth_passwd",
        "onepass[.]regions[.]com/oaam_server/loginAuth[.]do|userid,pass",
        "personal[.]vanguard[.]com/us/AuthenticationServiceServlet|USER,PASSWORD",
        "secure[.]bankofamerica[.]com|onlineId,passcode",
        "sellercentral-europe[.]amazon|email,password",
        "sellercentral[.]amazon[.]com/ap/signin|email,password",
        "www[.]security[.]us[.]hsbc[.]com/gsa/passwordAuth|username,password",
        "www2[.]secure[.]hsbcnet[.]com/uims/portal/IDV_OTP_CHALLENGE|idv_OtpCredential",
        "onlinebanking[.]mtb[.]com|UserId,Passcode",
        "accounts[.]logme[.]in/login[.]aspx|email,password",
        "www[.]gotomypc[.]com/users/login|UserId,Password",
        "authentication[.]logmeininc[.]com/login|emailAddress,password",
        "www[.]bitfinex[.]com/sessions|login,password",
        "poloniex[.]com/login|username,password",
        "www[.]coinbase[.]com/sessions|email,password",
        "[.]fiservse[.]net|PrincipalID,PrincipalPWD",
        "exchange[.]gemini[.]com/signin|email,password",
        "www[.]binance[.]com/user/login[.]html|email,password",
        "www[.]cryptopia[.]co[.]nz/Login|EmailAddress,Password",
        "www[.]bittrex[.]com/Account/Login|UserName,Password",
        "fxpayments[.]americanexpress[.]com/fxipfo/IPLogin[.]do|userName",
```

```
        "cm[.]netteller[.]com/login2008/Authentication/Views/Login[.]aspx|IdTextBox",
        "access[.]jpmorgan[.]com/prelogin|userID",
        "my[.]electroneum[.]com/authenticate|my_pin",
        "chsec[.]wellsfargo[.]com/login/login[.]fcc|PASSWORD",
        "wexhealthcard[.]com/LoginPage[.]aspx|TextBoxUsername,TextBoxPassword",
        "/Login|ctl00$Main$userNameBox,ctl00$Main$passwordBox",
        "/ebc_ebc1961/|PWD=",
        "signatureny[.]ebanking-services[.]com/EamWeb/account/login[.]aspx|textBoxCompanyId,textBoxUserId",
        "businessbankingbdc[.]tdcommercialbanking[.]com|ConnectID,password",
        "securentrycorp[.]nsbank[.]com|publicCred1"
    ],
    "sni_invalid_doman": "cloudflare[.]com"
},
"nmc_api_url": "hxxps://api[.]opennicproject[.]org/geoip/?json&ipv=4",
"nmc_dns": [
    "167[.]160[.]36[.]72",
    "172[.]106[.]170[.]81",
    "185[.]141[.]62[.]5",
    "192[.]250[.]230[.]196"
],
"registry": [
    "SOFTWARE\\S.W.I.F.T.",
    "SOFTWARE\\LogMeIn Ignition",
    "SOFTWARE\\PyBitmessage",
    "SOFTWARE\\Hex-Rays",
    "SOFTWARE\\Whole Tomato",
    "SOFTWARE\\WinLicense",
    "SOFTWARE\\LogMeIn",
    "SOFTWARE\\HexaD",
    "SOFTWARE\\GitForWindows",
    "SOFTWARE\\Cppcheck",
    "SOFTWARE\\TortoiseSVN",
    "SOFTWARE\\VisualSVN",
    "SOFTWARE\\DASH"
],
"screens": {
    "count": 10,
    "interval": 4
},
"software": [
    "Alliance Workstation",
    "Alliance WebStation",
    "Microsoft Dynamics RMS Store Operations",
    "Cisco",
    "Citrix",
    "Dashlane",
    "Fund",
    "FortiClient",
    "VPN",
    "LexisNexis",
    " OPOS",
    "Card Processing",
    "Boot Camp",
    "SII RP-D10",
    "Protect",
    "Withdraw",
    "Cloud",
    "Private",
    "SWIFT",
    "Wallet",
    "Bank",
```

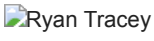
```
      "Cash",
      "Backup",
      "Replication",
      "Back-up",
      "Altaro",
      "CAM Commerce Solutions",
      "MSR",
      "iDrive",
      "Shadow",
      "Ledger",
      "VMware",
      "Trade",
      "Money",
      "Treasury",
      "Dropbox",
      "Box Sync",
      "Payment",
      "Token",
      "Coin",
      "Aldelo",
      "Microsoft POS",
      "TeamViewer",
      "Trezor",
      "PuTTY",
      "FileZilla Server",
      "M262x",
      "Vnc",
      "LogMeIn",
      "QuickBooks",
      "ScreenConnect",
      "Blockchain",
      "VIP Access",
      "mRemoteNG",
      "Storage",
      "Unitrends",
      "Password",
      "Double-Take",
      "Diskeeper"
   ],
   "systems": {
      "cmdline_name": "cmdline.txt",
      "debug_name": "debug.log",
      "keylog_name": "keylog.txt",
      "proc_data": {
         "BACKUP": [
            "1B158C51,F0741D99,B67F071D,8B5BB6B4"
         ],
         "BANK": [

            "0EF7D2D4,FFEF3E33,FEE9B9A7,A87BD8FA,C4A10E3C,D412F86A,01202A70,6B08460F"
         ],
         "CRYPT": [
            "24273C2B,689C61CA,1A528E29,71920AD0,9AAC98B8,89153906,1C2B38EA"
         ],
         "DEV": [


"FC09DD5F,2D8B32E8,0E603F68,ACD24EE9,50C046A3,4CAADC09,FB8B6597,9CAF30A4,275FA4AA,1A2456E8,D67C092F,E1E41D53"
         ],
         "ETC": [
            "36358A68,FE5AD73D,FAF3E612,6B473017,E8A0F232,E34D0957,D81477DF,BD4FB02F,D1C1A6F1,
44AAB2D8,A1B91179,B317125A,7E682F49,C7F12F81,8F4652C1,73D6E9BD,9F1F9193,06A3513B,831BB9C2,
```

173438F5,D359E656,6D38C83E,E8504062,9018EA47,EC6EDC68,9FAF2CDE,EB821910,8148A003,89961D2B,
E86FD330,D1879795,9E514577,B6453869,AB998E4A,117C8602,9CEC32CD,A649FA98,9919944A,ED096D7A,
ADE4C1F4,17F5E5AE,373DD31D,FEA3B580,6194AEDD,51FB05BB,B7F66707,A85BBCA4,21E4F6E1,5E7F7CFC,
09427565,2DE6E2CC,1F8A1A34,4DCC4E6D,34A3856D,7D1A6CF5,DFC68EA6,D1F0F78A,6164D621,81D83B96,
C93A54AD,4FDDDDBA,DF785E69,AD6C4956,AF407D3F,612836CB,BEB1E867,08ECCE46,7439D83C,5FE92CB4,
AC6D207E,66A4EFE6,56A6914D,CEB75996,71DEA7D8,831F25F0,65A59DBC,E11E8CAF,066F25CC,AB58C2B3,
8AA8ADD1,2BD2F601,C3C39C30,3DFE0297,9B60D929,D5202EDF,26382804,13FE74C3,D9CFAAFC,1EC38064,
B48B28E6,EE3F8F14,BEC4AE1B,0CEF981A,AE21F4BD,ED05A7DD,BBABEE8B,DB4FF76A,3F65FEA1,80824730,
B8132034,E4E0706E,E0872300,06A367AC,530DFC96,BFBD391A,EDB493EA,6E80D8EB,46729516,E58E420F,
9AB07D59,5FD4AF15,50F84A3B,7F383A3D,08D73940,CED3FE41,2FD368B0,941F78B7,5ACE07E5,8FBAB9AA,
2A77EE37,849A7001,F4DD7D00,ADE4FF12,B8F4EEAA,22F83602,9C5305AC,953F5973,612BE1A4,A8F17723,
27D94A1A,851F2672,CBCA91DA,8A431577,E9FC18B8,E47AA073,E1D08BD3,AC3DDCBE,8713F420,7C0A2C03,
CF18FB77,AD2581F7,F1B705CE,4BBAB8FA,DDB7DAD6,298944A0,AD1A170A,24549603,BC90AA40,48D0E729,
2CC8BE52,C8C60F25,21B4D9EC,7C934BFA,C6FA5FCE,1557F882,51C0BBAE,694203C1,2611B4E2,817EEA70,
CC52CBB0,3CD2F52B,EC155B26,02E6CF14,9FED0D8C,725E1B53,1008DDC9,EDB8EDB4,F1D6EC4C,6DAC3C4A,
9B282310,22BCF394,1FF114CE,1FED69F4,2B68955F,5C5424CD,F0965573,28E223B0,F4ABF75B,87D7BE37,
B5DC8668,A053B931,EEDAEB84,5AD6F060,679F864C,00622956,3EDDD0CC,6166CBFE,B035BEAD,7C997C1F,
16B3C140,7814A3B5,7D04697C,53619F2C,CE5D9477,AF9D47B4,39D80B01,A58F2523,4941148D,1A8549EE,
72F9117D,17238ECE,17594E4C,AA38D481,D6B2F72C,0617EB1D,B107D005,5BA8998F,6555370D,4399C284,
2A701224,307BA4C3,ED256EAD,C0507C47,DA46C5A0,EB3E9524,17028891,094CAAEF,7D0208BC,DD63FE4B,
CBABEDC0,208A6703,2DE10091,76897639,A1B089CF,00226A31,B0BEBB22,CD47F86A,EE1C4CCA,D64A0C5D,
8B34C6B4,853A92F8,EDD799AE,C76EC047,AEA032F4,8EE2D469,42F4A14F,C8D635E6,34461F8E,1F8F32DD,
613E33F3,CEFE1900,06AA08F8,40C41FE4,33DD263C,DB11366C,6863CBC4,61774F83,79A179C4,C2803376,
92C06016,B010237A,432A30F6,EAE4D38E,A28184B6,C04B85DD,535AE700,E7868E6A,20F6766D,45EFDE73,
FFA7DDB0,1177CB80,1DC8DC0A,753C1BC4,46A65141,1520C6FC,115465F8,E05FC686,48755CDD,9F44E2AD,
017C72BF,C90A4D55,3F315ED6,530EC7CF,28F258E8,4D50DB3A,5DE0A119,EBFB0CC8,7F006276,A289D938,
637301C6,6905564E,3F8C2811,15CD3ABD,F8DA60CF,B6A73D4C,A9F98641,40C1472D,C322A28E,9427291A,
3EE0249F,5025B29A,21E0C754,971C9B6E,5433B062,D4ABCE94,A5EF5232,7D292D5C,54D062A7,7EA56DD4,
6F39D5E5,637690A0,C99DF88D,D40AE36D,9EAA7E6D,E7FE982A,DB5B130C,D69E7AB7,AB5A540A,23746103,
D00D57A1,B19BE433,07F5B6AB,368506FF,61B3C004,D3174803,2E76EDDA,6DCA34D9,DAA840B0,6BAF0CCB,
A836E949,10D35165,23E559EE,FF658664,B35A1FBF,83ED7E1E,CF3FE156,C5955648,22192A6C,E53E7CF7,
E9D4AA1C,CD7007CE,E8FB9DD8,67D09CD7,C2666E5E,CE57EE1C,ACF1AFE9,9CD1C89B,5E1F3383,900F4BC6,
D4D1DABB,42E2AFE1,0254BF12,DBF32FA8,54E18969,29A16C6C,2A819D48,C63BAD20,F761556A,BFA7A700,
EC18D835,27499C3F,1883386F,C25C84B6,45B5186D,7729D6C7,867B6AD8,F49BFC32,F33B69C9,A301D995,
1E6CC84E,A4CA05F3,FB98997A,68B0A2D2,89FEE818,35121417,754DB640,D35B8E8B,8BFADABE,630380BD,
2678169A,0F17040E,B18DA826,26BE3D57,C47610CC,D5AFE347,D697181B,72C1148A,DC940E90,3E15B814,
6199DFE0,600FE875,EA5696FD,147A1679,D0393D82,7314615B,6A947CA9,5F4A22FB,7A74A927,0D9C7E1E,
FA429238,A5E4363A,07250472,D1D7BD33,38AF4595,3A2C4CF5,B9F0288B,9A0ABBDD,1112E894,F7F04B91,
F10C467C,78A3A99F,318C45D0,0FAE970A,5465EE3B,0B4B3079,6B55165F,920F9997,DD4824DF,63FB331F,
3DCF12DB,57FF7496,9DED7AE9,8589E085,28EFE770,1711D3C5,F17E2D3B,16B88982,9330B1BC,3C2AC51B,
4C3E6C40,CA8F2447,5AD194E7,DCAD848C,09B0FF98,3E2E8993,2FE5DDDB,811FE2F6,CAD75136,7C0286F7,
EB009E90,161C6399,9B5BB8F0,540539E4,DB4FEBFC"
        ],
        "KEEPASS": [
            "5FD22F11"
        ],
        "OTP": [
            "C03F4FCF,9652F18E"
        ],
        "POS": [
            "24EF6AE6,3781645F,4F6F25C8,0A8349B8,0CD13601,C605BC82,F4D9B2C9,FE115180,094CAAEF,
8B3D5DC7,F3F60AC6,C4F259E9,FC76F5FD,C3EFD902,BC983A7A,CBF115E2,3C07DE76,E1BE2720,4E576C6A,
4EE1F7A0,8B6806E8,1536649B,2E35EC33,7A0C26D5,A8F97EFE,DEDC862C,EC6824AA,9CCFC9EE,8149D125,
E1D6EADC,D4944D9B,C40FB1D0,AC0D9C0B,2BC674EA,43EC2716,10C9F247,5AFE6EFB,0DDCE00C,356111CF,
D7AC7B76,A4CAFC94,836AD74E,E3B8C0FE,8D5B8FED,CD5EBB41,FAA8EC0F,6C5A3F7C,D6F95CAE,9F3B52CA,
1769DF67,05578F82,470E840D,26C0322F,53C75648,BE044A87,AEEB6676,91633EBB,42E4F9CF,A460CB67,
60ABE024,139E2D2A,A9FC0EB6,DF9B3D89,3126A231,EA61F54D,5A4C20D9,8CCE1DB9,DC89F098,B19DAC22,
A5514494,E2B0AA31,C669C000,BC3F21A8,0A2F3C0D,2684E509,C7E45A67,9271EEF1,F5D272A7,73B89844,
86EF263E,FAD5CDD4,0CC50AC3,D560BA10,D69AB5B7,ECA0854B,DB86F715,045C1577,825E31E2,B10F2A74,
6E311050,ADFFE3F0,5A7E28CA,C6C97A0B,86C224A0,6EA77FF8,8A7800F1,9FD1B3D3,AE529627,900AB121,
F7B0C18A,2C63FE12,5FDAC8A8,AD668FCD,EF2390CC,E21B9F2A,01DB1338,547C2E92,F4E35C03,E2204FCF,

6FD903D5,E6DA53C5,AACB54D3,889CF566,A8C0FD20,89114174,ADF4E2F0,7282CFD9,3E2425A6,FEC95559,
68D404AD,DFD54B81,8AF6B819,6F4C9BE2,9300EB54,DDDA4B02,8B0A7F43,8D047993,CE75381D,2FD25572,
9F2C60A8,81DF266C,BDC933B5,0ED1A64B,C52CEFD4,3AB21009,B452D1B7,E1F2181D,F1622C8E,0712D8DC,
63A92D4D,C7356239,FF8939A1,62067A58,678F923D,50867705,9C6D97C4,38EFAC0A,7F6AD007,4CB3D796,
C7C8C5BF,699C5BD9"
            ],
            "REMOTE": [
                "039ABDEB,6DA3C888,482DD2A1,900D2C2A,BFB0AC8D,F5149492,D6492824,FB20B6FB,95D9C53A,
5A83C631,095BD149,A887B207,363ED833,F636D50B,ACF8432A,780CA03F,6EC27C57,19587E8E,096B3678,
BC439383,37CCDBF3,88E3A913,E2AEA9C6,5F023EF9,C03610DC,2C798A09,B4434710,09005D42,054D2163"
            ],
            "SWIFT": [
                "106530D5,8B587452"
            ],
            "TAX": [
                "0414785A,14091FAA,53A9657A,6EC94CCA,EC99DB1B,D1F9F2AB,5A7DE274,9CD008E4"
            ],
            "WALLET": [
                "1444C599,F3D4B706,65C9EDBA,78D00514,52DDCFE9,8BBCD13D,85E1225D,5C7AFA0A,9E099A2C,
FF641DB8,CD6B5665,72A2CDEE,165DFED9,53AC8A37,B78C2122,C90608DE,AC733BEB,969FF04A,59FF8C85,
628B247C,7909968A,670220E0,60CB5762,EA51B6DE,00695083,2DB5BB8B,E18F2D2B,7FCD7A2F,D0FF27C3,
BE468513,FFE75C6E,7A4813E4,B72A715D,598572BF,9DDDB007,9BB1BB92,46C9E01E,2E27F2BA,A71DA7CA,
FA144180,930FB7AF,6C5B82CC,D85FADEC,F703F6E9,E6A9DB28,4407FB04,4EF22574,A7E87469,0E5C39E3,
74097066,3BC1DBC4,2E3AEF7F,96020473,F7F19F1C,F2C3756E"
            ]
        },
        "screens_dir": "scrs",
        "send_attempts": 5
    },
    "usb_file_find": {
        "exts": ".doc|.docx|.xls|.pdf|.txt|.rtf|.xlsx",
        "gold_masks": "*.rdp|*.kdbx|*.vnc|*.ovpn|*.crt|*.key|*.pfx|*.pcf",
        "keys": "*passw*|*logins*|*wallet*|*private*|*confidential*|*username*|*wire*|*access*|*instruction*|*credent*|*cardholder*"
    }
}



About Ryan Tracey

Senior Threat Researcher at BlackBerry Cylance.

Ryan Tracey is a Senior Threat Researcher on the Threat Intelligence team at BlackBerry Cylance.