Analysis of LilithBot Malware and Eternity Threat Group | Zscaler

By Shatak Jain, Aditya Sharma

Published: 2022-10-05 · Archived: 2026-04-05 13:54:42 UTC

Introduction

ThreatLabz recently discovered a sample of the multi-function malware LilithBot in our database. Further research revealed that this was associated with the Eternity group (a.k.a. EternityTeam; Eternity Project), a threat group linked to the Russian "Jester Group," that has been active since at least January 2022. Eternity uses an as-a-service subscription model to distribute different Eternity-branded malware modules in underground forums, including a stealer, miner, botnet, ransomware, worm+dropper, and DDoS bot.

The LilithBot we discovered was being distributed through a dedicated Telegram group and a Tor link that provided one-stop-shopping for these various payloads. In addition to its primary botnet functionality, it also had built-in stealer, clipper, and miner capabilities. In this blog, we'll provide a deep analysis of the LilithBot campaign, including a look at several variants.

Key Features of this Attack

- Threat groups have been enhancing their capabilities and selling them as Malware-as-a-Service (MaaS) in exchange for a membership fee. One such cyber criminal group, dubbed "Eternity," has been found selling the malware "LilithBot"
- "LilithBot" is distributed by Eternity via a dedicated Telegram channel from which we can purchase it via Tor. It has advanced capabilities to be used as a miner, stealer, and a clipper along with its persistence mechanisms.
- The group has been continuously enhancing the malware, adding improvements such as anti-debug and anti-VM checks.
- The malware registers itself on the system and decrypts itself step by step, dropping its configuration file.
- LilithBot uses various types of fields such as license key, encoding key, and GUID which is encrypted via AES and decrypts itself at runtime.
- It steals all the information and uploads itself as a zip file to its Command and Control.

Summary

In July 2022, Zscaler's ThreatLabz threat research team identified a multifunctional malware bot known as LilithBot, sold on a subscription basis by the Eternity group. In this campaign, the threat actor registers the user on its botnet and steals files and user information by uploading it to a command-and-control (C2) server using the Tor network. In this campaign, the malware uses fake certificates to bypass detections; it acts as a stealer, miner, clipper, and botnet.

In this blog, ThreatLabz will explain various aspects of the LilithBot threat campaign.

About Eternity

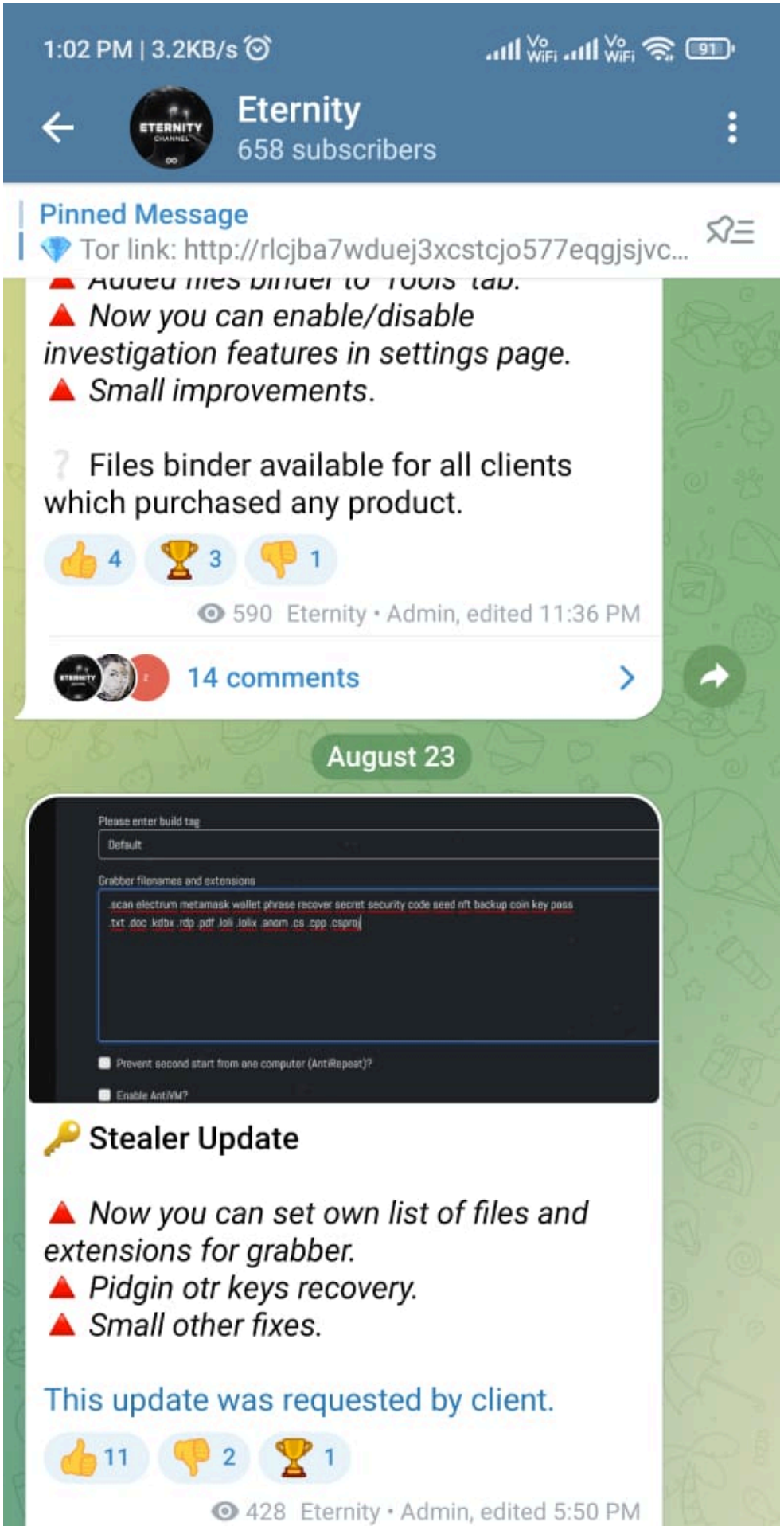
Eternity Project is a malware toolkit which is sold as a malware-as-a-service (MaaS). These malware are distributed via the Tor proxy. Eternity advertises via a dedicated Telegram channel named @EternityDeveloper and has an email address of eternity@onionmail[.]org. They have different types of services:

- Stealer
- Miner
- Clipper
- Ransomware
- Worm+Dropper
- DDoS Bot

Eternity usually operates via Telegram and accepts payments through popular cryptocurrencies including BTC, ETH, XMR, USDT, LTC, DASH, ZEC and DOGE.

They provide customized viruses and will create viruses with add-on features if the customer desires.The price of the malware ranges from $90-$470 USD.

The below screenshot of the Eternity Telegram channel illustrates the regular updates and enhancements the group makes to their products.


1:02 PM | 3.2KB/s
Eternity
658 subscribers
Pinned Message
Tor link: http://rlcjba7wduej3xcstcjo577eqgjsjvc...
Now you can enable/disable investigation features in settings page.
Small improvements.
Files binder available for all clients which purchased any product.
4
3
1
590 Eternity • Admin, edited 11:36 PM
14 comments
August 23
Please enter build tag
Default
Grabber filenames and extensions
.scan electrum metamask wallet phrase recover secret security code seed nft backup coin key pass
.txt .doc .kdbx .rdp .pdf .loli .lolix .enom .cs .cpp .csproj
Prevent second start from one computer (AntiRepeat)?
Enable Anti-VM?
Stealer Update
Now you can set own list of files and extensions for grabber.
Pidgin otr keys recovery.
Small other fixes.
This update was requested by client.
11
2
1
428 Eternity • Admin, edited 5:50 PM

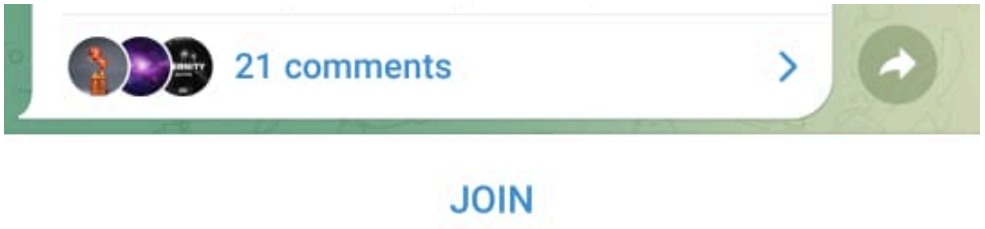
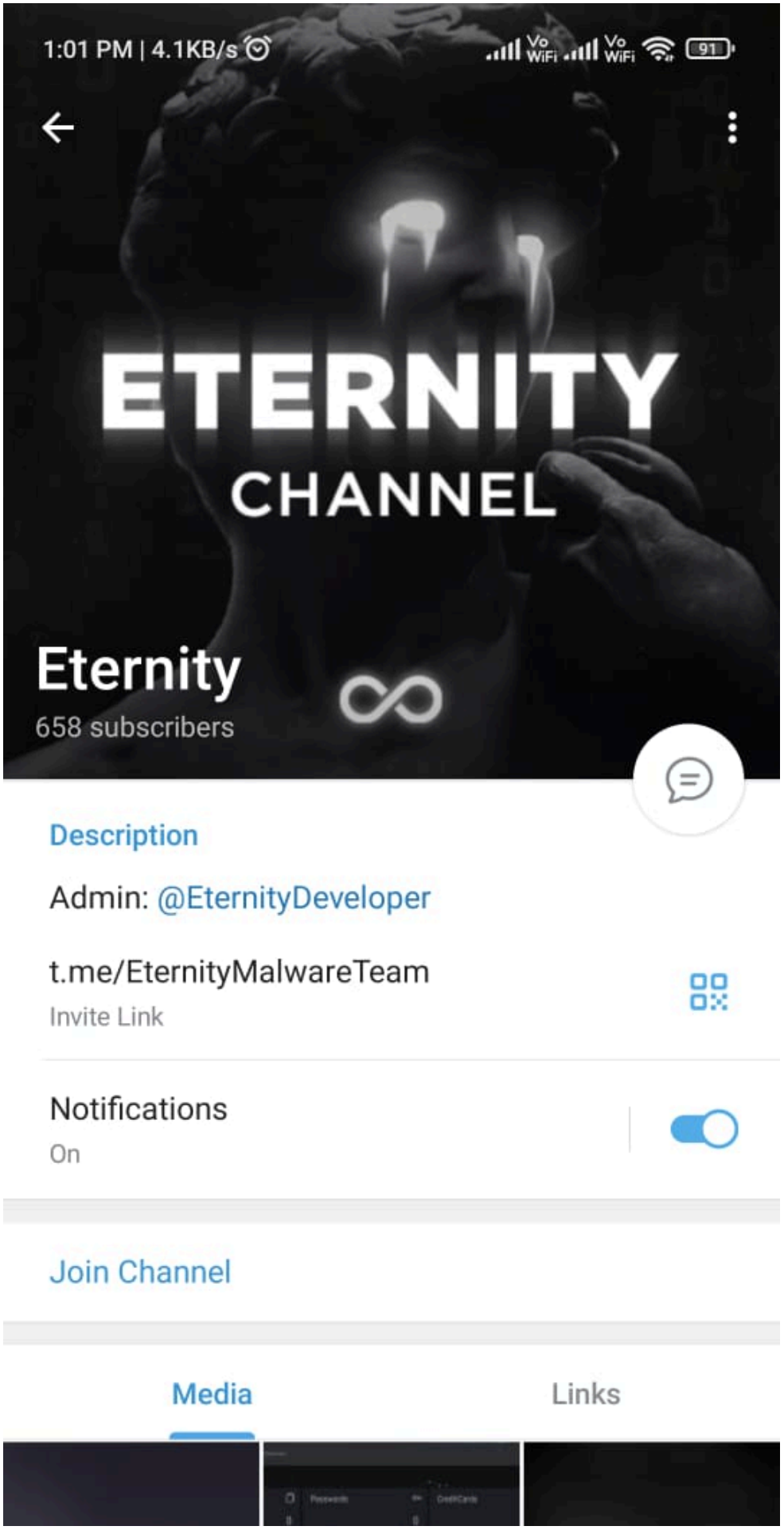


Fig 1. Eternity Telegram Channel

The Telegram channel is dubbed “Eternity Channel.” Basic account details are shown below.


1:01 PM | 4.1KB/s
ETERNITY
CHANNEL
Eternity
658 subscribers
Description
Admin: @EternityDeveloper
t.me/EternityMalwareTeam
Invite Link
Notifications
On
Join Channel
Media
Links

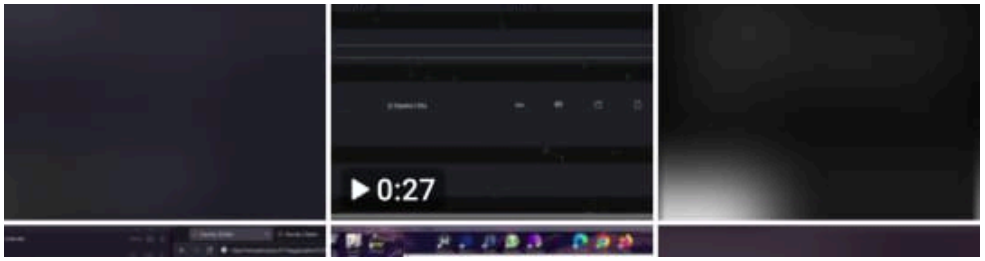


Fig 2. Telegram Home Page

The Eternity group regularly directs clients to their dedicated Tor link, in which their various malware and their features are laid out in detail.

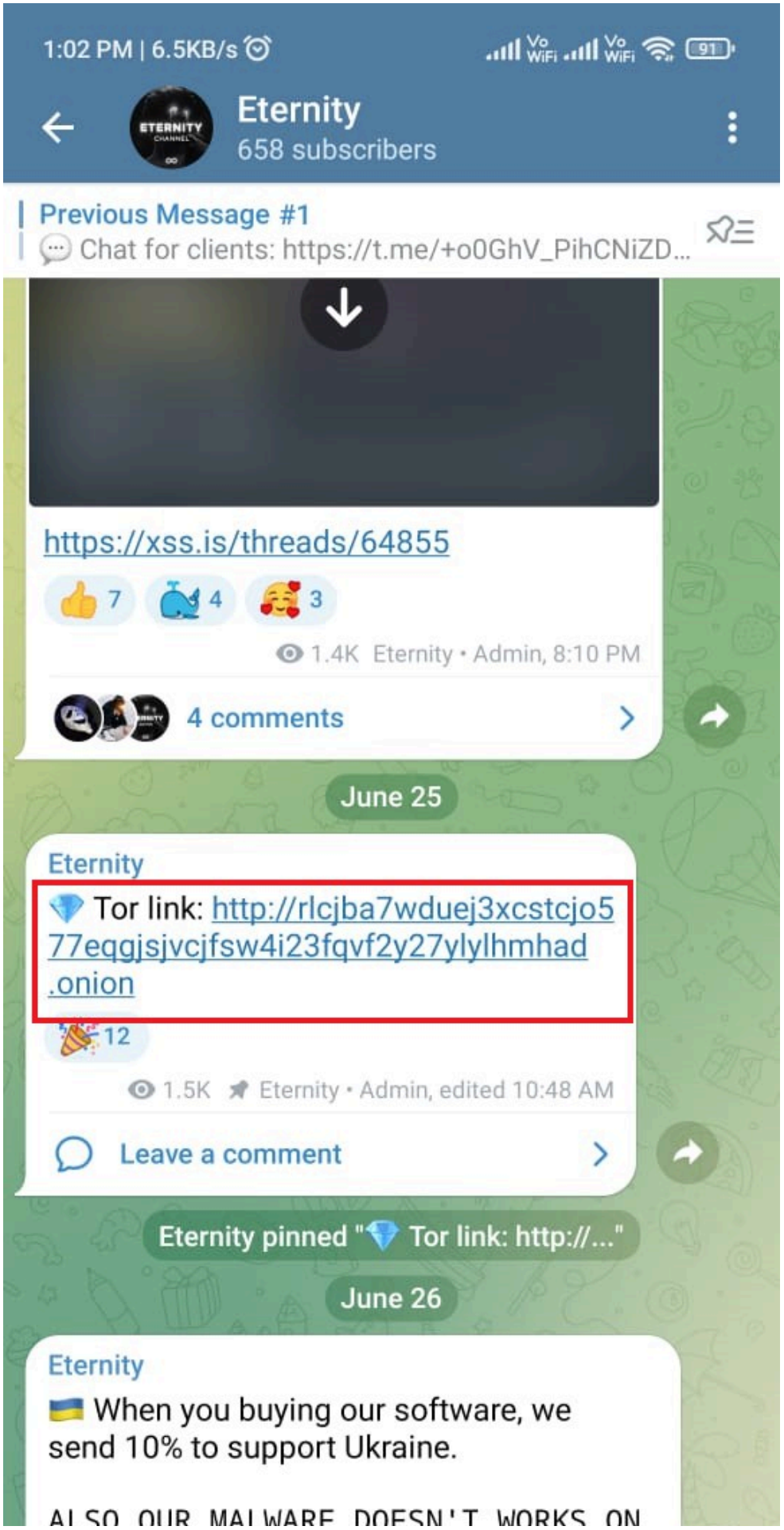

1:02 PM | 6.5KB/s
Eternity
658 subscribers
Previous Message #1
Chat for clients: https://t.me/+o0GhV_PihCNiZD...
https://xss.is/threads/64855
7
4
3
1.4K Eternity • Admin, 8:10 PM
4 comments
June 25
Eternity
Tor link: http://rlcjba7wduej3xcstcjo577eqgjsjvcjfsw4i23fqvf2y27ylylhmhad.onion
12
1.5K Eternity • Admin, edited 10:48 AM
Leave a comment
Eternity pinned "Tor link: http://..."
June 26
Eternity
When you buying our software, we send 10% to support Ukraine.
ALSO OUR MALWARE DOESN'T WORKS ON

Fig 3. *Tor link mentioned in Telegram*

The Tor link leads to the below homepage, which explains the various products and modules available for purchase.

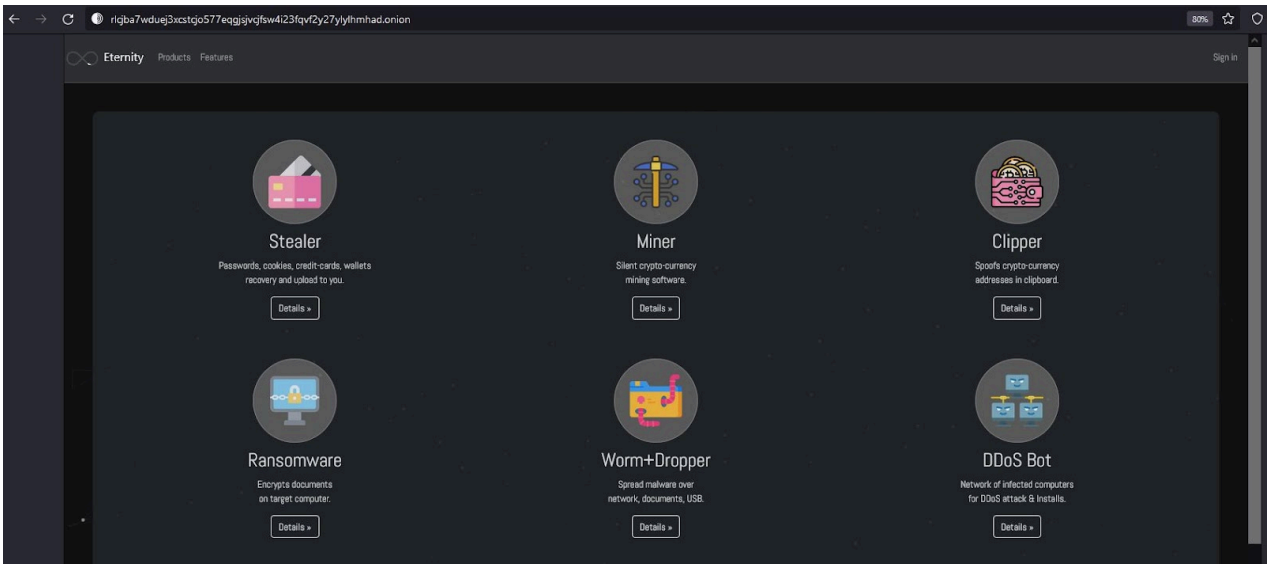


Fig 4. Tor site for Eternity group

The highest priced product for sale is their Ransomware, described in the below screenshot. The ransomware encrypts documents and files of the targeted user. The Tor page includes a dedicated video on how to generate the ransomware payload.

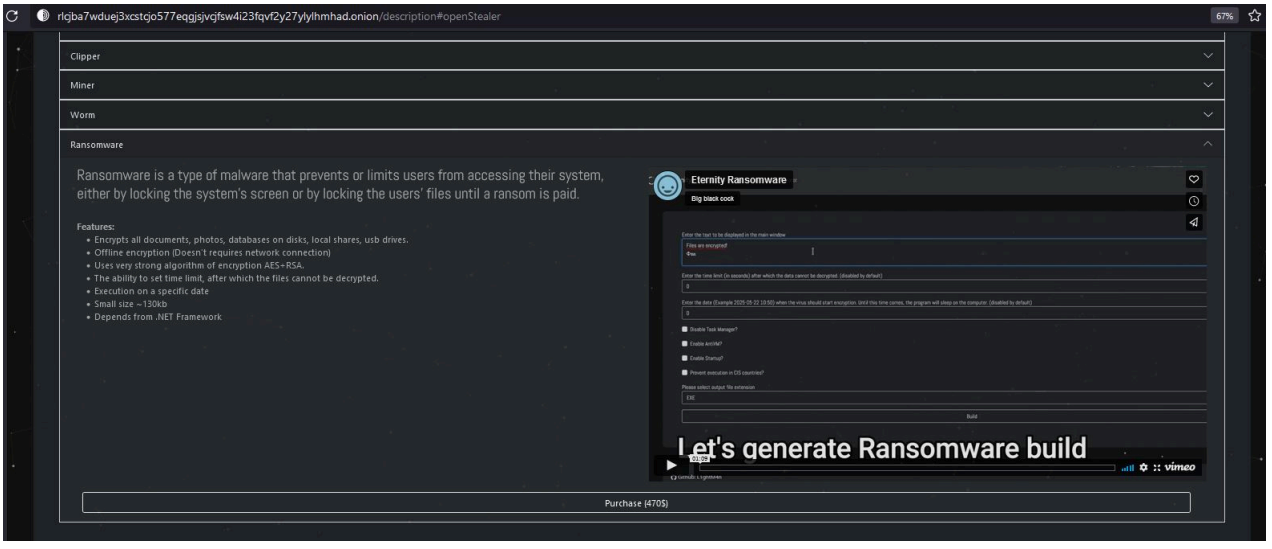


Fig 5. Features of payloads

In summary, Eternity has a very user-friendly service that is:

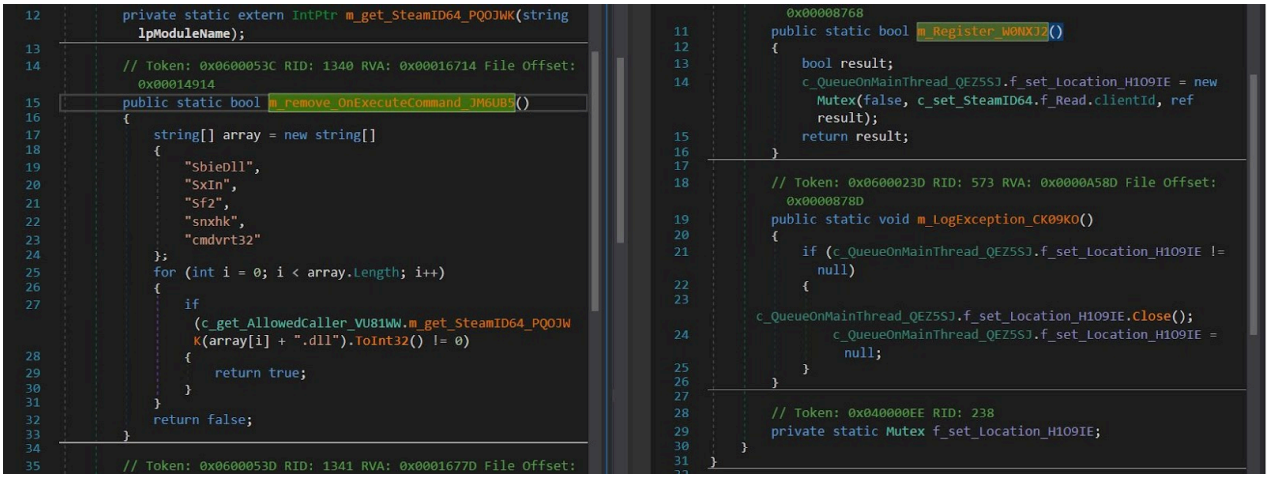
- **Easy to purchase and operate** via Tor, with a wide range of popular crypto currencies accepted for payment.
- **Customizable** to fit clients' specific needs.
- **Regularly updated** at no additional charge. They also offer many add-on discounts and referral rewards to their customers.

Comparison Between Two Variants

As the LilithBot malware has evolved, we have observed slight differences in the main function of different releases.

Several commands that were present in earlier variants are not present in the newest variant that we have received. These functions include:

- Checking for the presence of various DLLs by iterating via arraylist and returning a Boolean value.The DLLs mentioned are related to virtual software like Sandboxie, 360 Total Security, Avast, and COMODO AVs.
- Checking for Win32_PortConnector which represents physical connection ports such as DB-25 pin male, Centronics, or PS/2. This ensures that it's on a physical machine rather than a virtual machine.



```
        private static extern IntPtr m_get_SteamID64_PQOJWK(string
          lpModuleName);

        // Token: 0x0600053C RID: 1340 RVA: 0x00016714 File Offset:
          0x00014914
        public static bool m_remove_OnExecuteCommand_JM6UB5()
        {
            string[] array = new string[]
            {
                "SbieDll",
                "SxIn",
                "Sf2",
                "snxhk",
                "cmdvrt32"
            };
            for (int i = 0; i < array.Length; i++)
            {
                if
                 (c_get_AllowedCaller_VU81WW.m_get_SteamID64_PQOJW
                 K(array[i] + ".dll").ToInt32() != 0)
                {
                    return true;
                }
            }
            return false;
        }

        // Token: 0x0600053D RID: 1341 RVA: 0x0001677D File Offset:
          0x00008768
        public static bool m_Register_W0NXJ2()
        {
            bool result;
            c_QueueOnMainThread_QEZ5SJ.f_set_Location_H1O9IE = new
              Mutex(false, c_set_SteamID64.f_Read.clientId, ref
              result);
            return result;
        }

        // Token: 0x0600023D RID: 573 RVA: 0x0000A58D File Offset:
          0x0000878D
        public static void m_LogException_CK09KO()
        {
            if (c_QueueOnMainThread_QEZ5SJ.f_set_Location_H1O9IE !=
              null)
            {
                c_QueueOnMainThread_QEZ5SJ.f_set_Location_H1O9IE.Close();
                c_QueueOnMainThread_QEZ5SJ.f_set_Location_H1O9IE =
                  null;
            }
        }

        // Token: 0x040000EE RID: 238
        private static Mutex f_set_Location_H1O9IE;
    }
}
```

Fig 6. Comparison between variants

It is likely that the group is still performing these functions, but doing so in more sophisticated ways: such as performing it dynamically, encrypting the functions like other regions of code, or using other advanced tactics.

Technical Analysis

The entry point starts with registration of the bot. The malware initially checks with a Mutex named "8928a2d3-173b-43cb-8837-0e2e88b6d3b1" and subsequently checks for a file in the Startup folder.

It then copies the same into the Startup folder if the file does not exist. The function StartupFilename then checks whether a file has been created which with an extension of ".exe",".com" or ".scr"; if not, it will append ".exe" to the filename and add this filename in the Startup path.

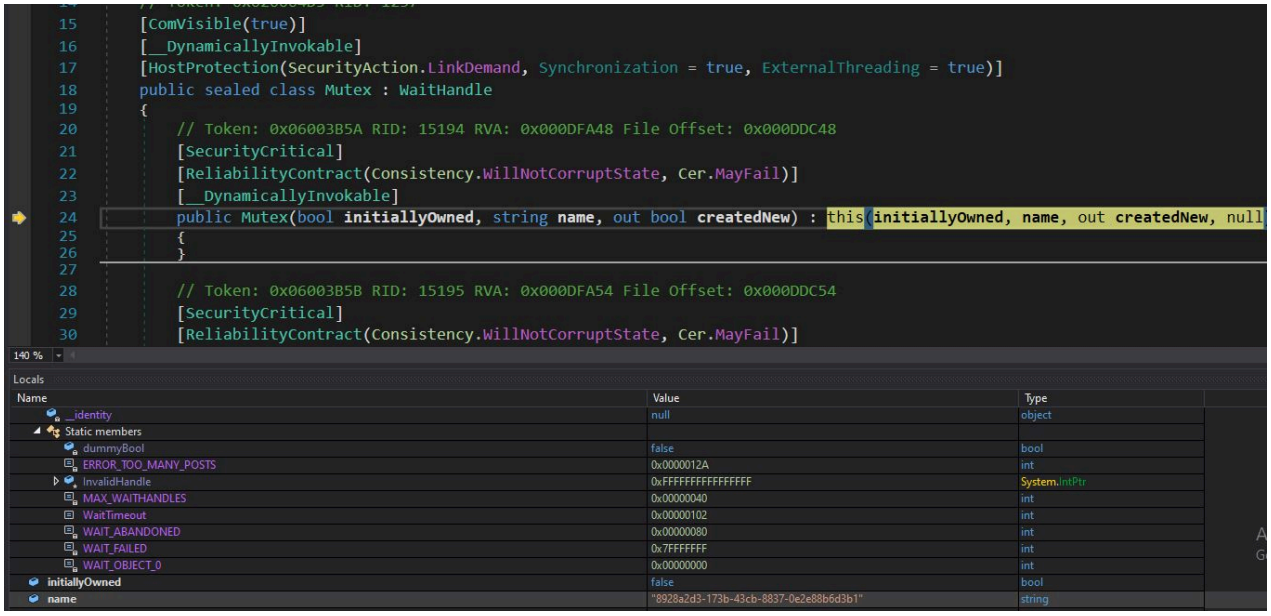


Fig 7. Mutex Creation

Fig 8. Checks Startup Files

The image below shows that the bot has successfully registered when the response to the decrypted data has the string “registered successfully” present in the register bot function, when checking the array data value.

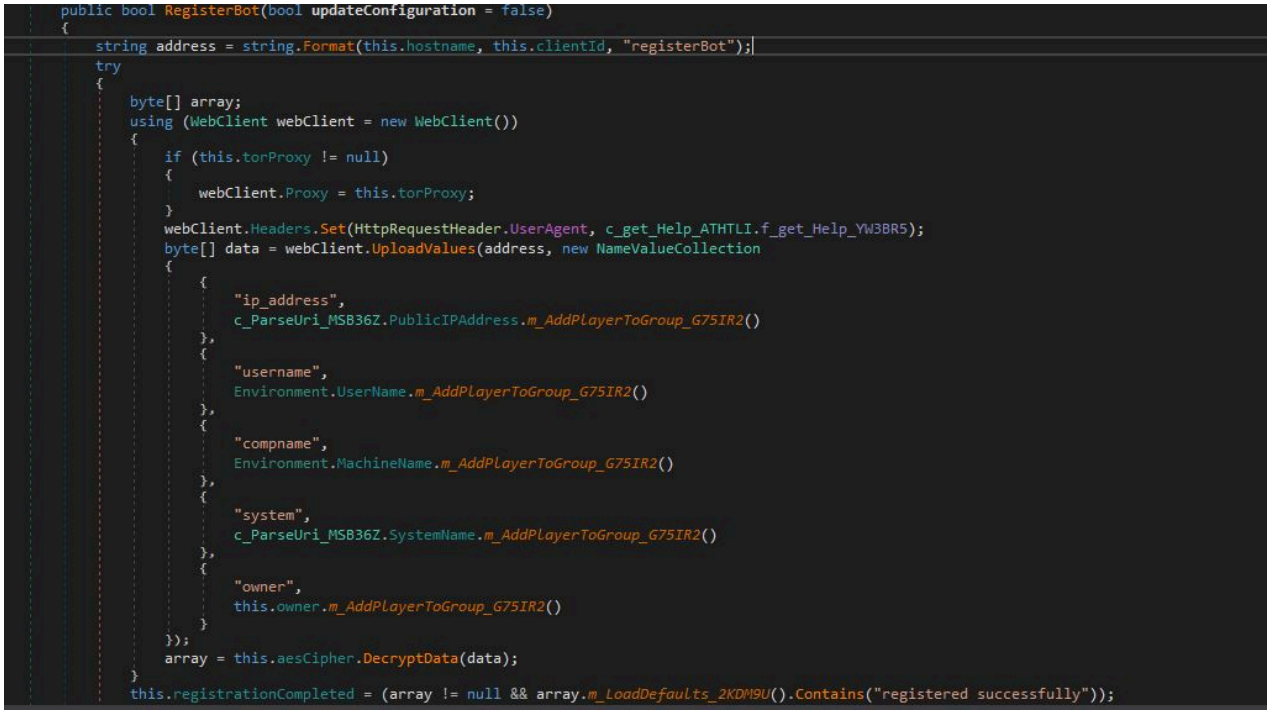


Fig 9. Steals User Information

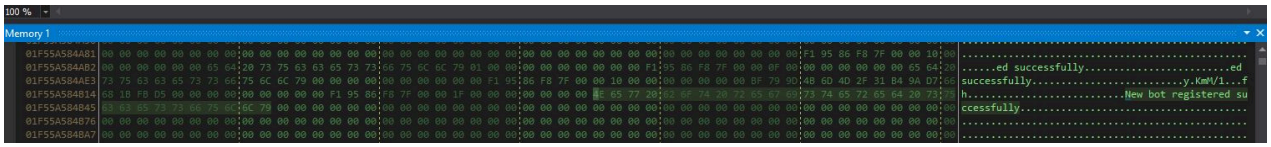


Fig 10. Registered Successfully

The Initialize function can be used to extract the value of different fields in a config file, as shown below. After decrypting the aes cipher, we can see all the important fields present in the config file. The following are the fields present inside the config file:

```
"Lilith": {
    "CommandsCheckInterval": 14
},
"BotKiller": {
    "Enabled": false
},
"Stealer": {
    "Enabled": true
},
"Clipper": {
    "Enabled": true,
    "Addresses": {
        "XMR":
"493eic71yTX23KnxuC1FimhkW5kEv1G2aMcE1spdBYot5BLo2ZdDbUcPCLdXMQPgLpgkNxLH4FWDRLjcdxmvG6ba4D8saKg
        "BTC": "bc1qd8e4maz97mv23slmgg7d4je2mydslkl5m56vdz",
        "ETH": "0xFf7f57a2c7952fD9550A5E0FE53d4F104886403A"
    }
},
"Miner": {
    "Enabled": false,
    "Pool": "pool.minexmr.com:4444",
```

"Wallet":
"493eic71yTX23KnxuC1FimhkW5kEv1G2aMcE1spdBYot5BLo2ZdDbUcPCLdXMQPgLpgkNxLH4FWDRLjcdxmvG6ba4D8saKg

"Password": "x",

"MaxCPU": "40"

}

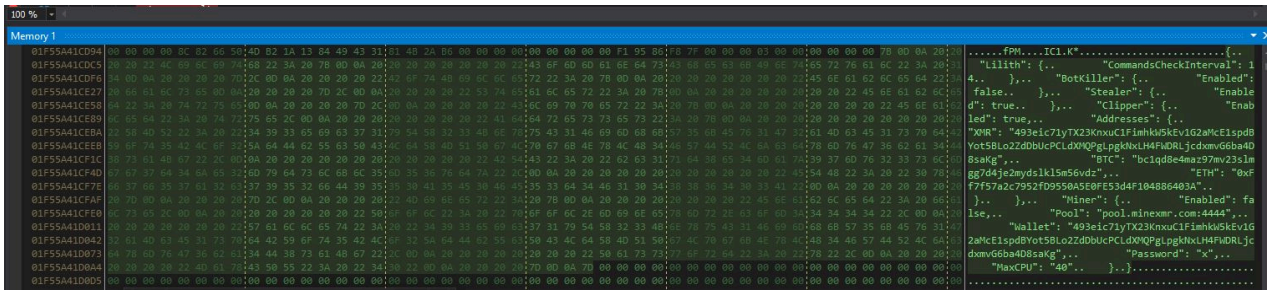


Fig 11. Decrypted Config File Found in memory

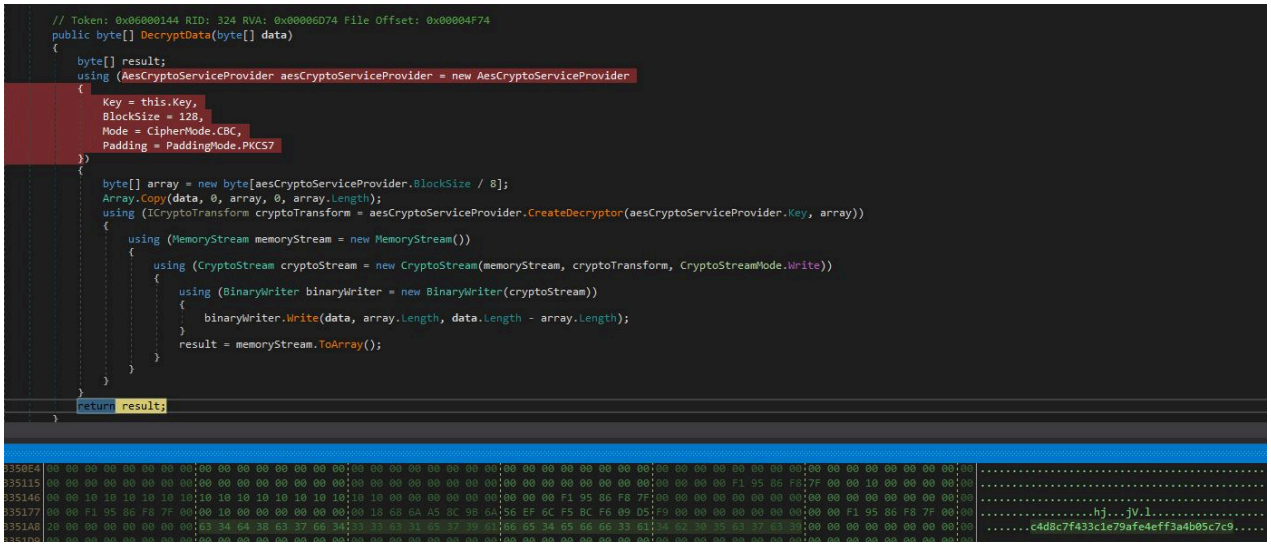
We also came across a function that confirms the malware is using its own decrypting mechanism so that it can't be decrypted manually.

All the encrypted data goes through the function "DecryptBytesToString" on which we can extend our breakpoint to know all the values of the decrypting data using dynamic analysis.

We can see that the C2 server has the IP address: 77.73.133[.]12 with the port no. 4545 with the api gate/ and which expects certain arguments for field {0} and {1}. The key and data are hidden inside the Hex array which we can see in the memory dump.

We can decrypt the encoded key which translates to the value c4d8c7f433c1e79afe4eff3a4b05c7c9.

We also observed a license key field which has the value 59BE0ABAF3BC570D8F6F88A597C64B85. This is the decrypting function; the below image shows the decrypted text for the corresponding values.



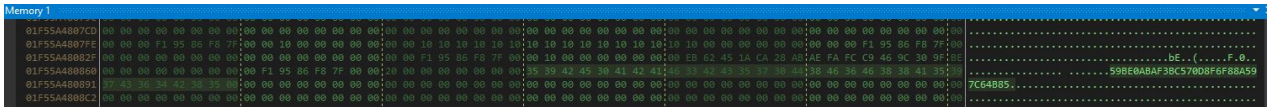


Fig 12. Decrypted License Key and Encoded Key

The sample also defines a function which gets the response of the body. If the response is not null, it then checks to make sure both the C2 server and the target's network are online. Then, it will then generate the GET request by checking a few permissions.

The malware further checks whether the hostname contains the onion domain. After checking the permissions, it downloads the Tor bundle and connects to the IP. The Upload File function combines the hostname with the client, name of the file, and directory as parameters.

```
public bool Listen()
{
    while (this.registrationCompleted)
    {
        Thread.Sleep(this.commandCheckInterval * 1000);
        string cmd = this.GetCommands();
        if (!cmd.m_Execute_AMMDV6())
        {
            if (cmd == "Lilith:ServerOffline" || cmd == "Lilith:NetworkOffline")
            {
                return false;
            }
            try
            {
                Thread thread = new Thread(delegate()
                {
                    c_checkCommandMappings_T203RB.m_get_Directory_PZC5N8(cmd);
                });
                thread.Start();
                thread.Join(2000);
            }
            catch (Exception value)
            {
                Console.WriteLine(value);
            }
        }
    }
    return false;
}
```

Fig 13. Checks if bot is online or offline

Network Artifacts

LilithBot malware shows 3 requests to the Host ip:77.73.133[.]12 with port 4545.The user agent shows the relation of the malware with LilithBot.

The first request is to register the bot with /registerBot API with the mutex name prepended.

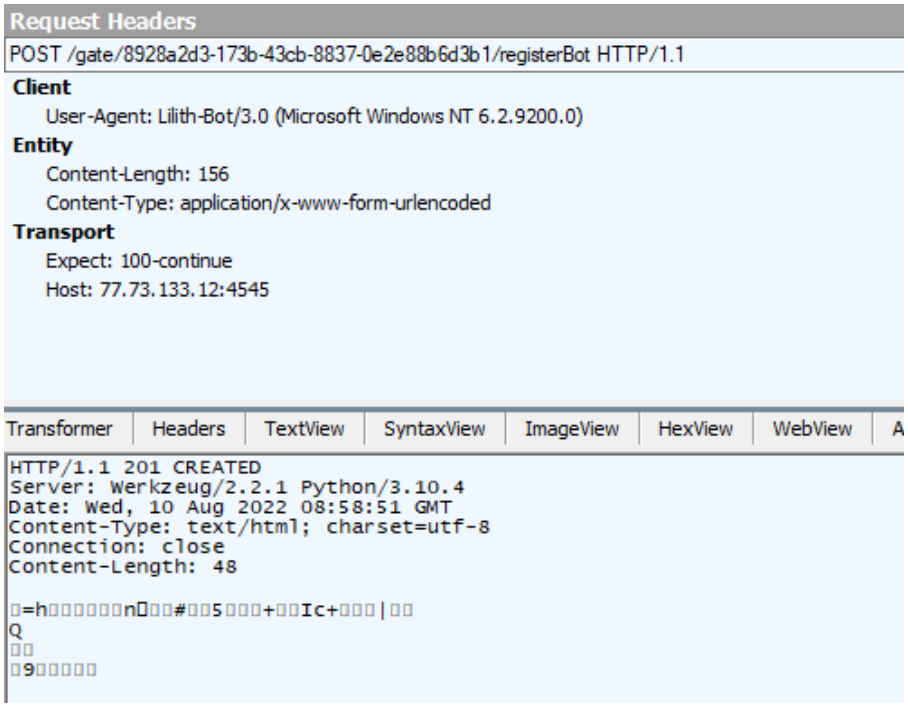


Fig 14. Sends Request to Register Bot

The second request is an API to download the file contents according to the plugin settings ‘admin_settings_plugin.json’.

Fig 15. Requests plugin settings

We see another request to upload the file in a ZIP format named as “report.zip” with dir parameter as ‘Stealer’. The zip file contains multiple directories that store information typical of a stealer, including the browser history, cookies, and personal information such as pictures stored in the C:\Users\[user]\Pictures folder, and much more.

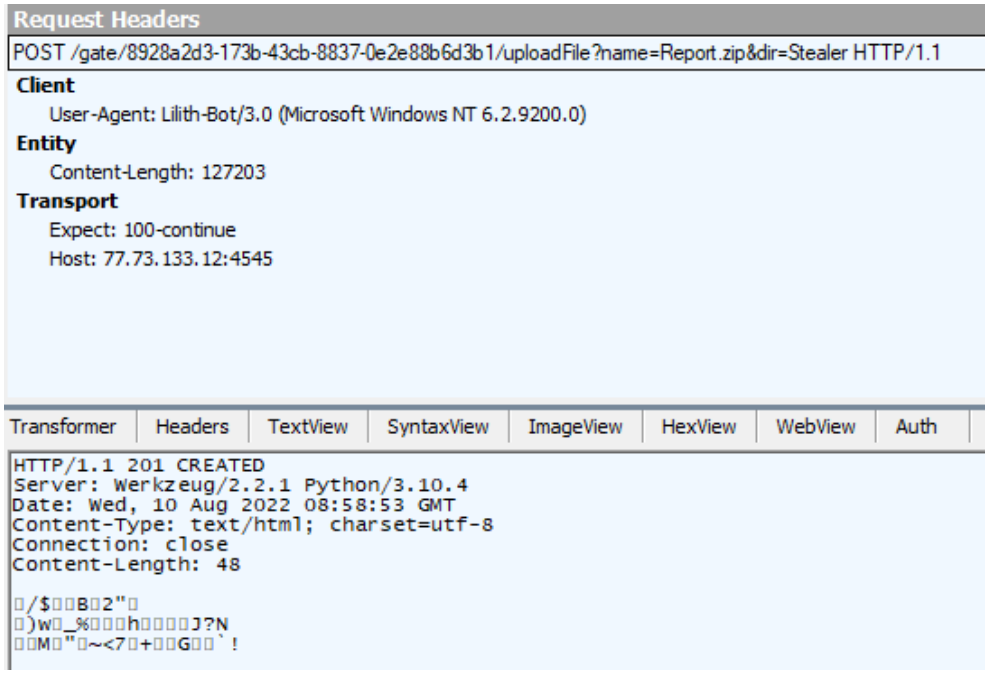


Fig 16. Uploads report file

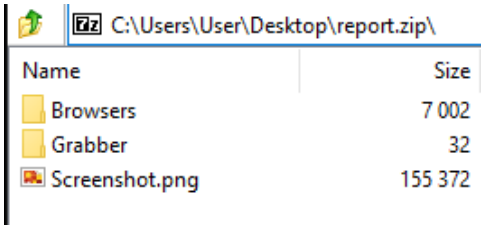


Fig 17. Contents inside Report.zip

Fake Certificates

A legitimate Microsoft-signed file is issued by the “Microsoft Code Signing PCA” certificate authority, and will also display a countersignature from Verisign. However, we have seen that the fake certificates in LilithBot have no countersignature, and appears to have been issued by “Microsoft Code Signing PCA 2011” which was not verified.

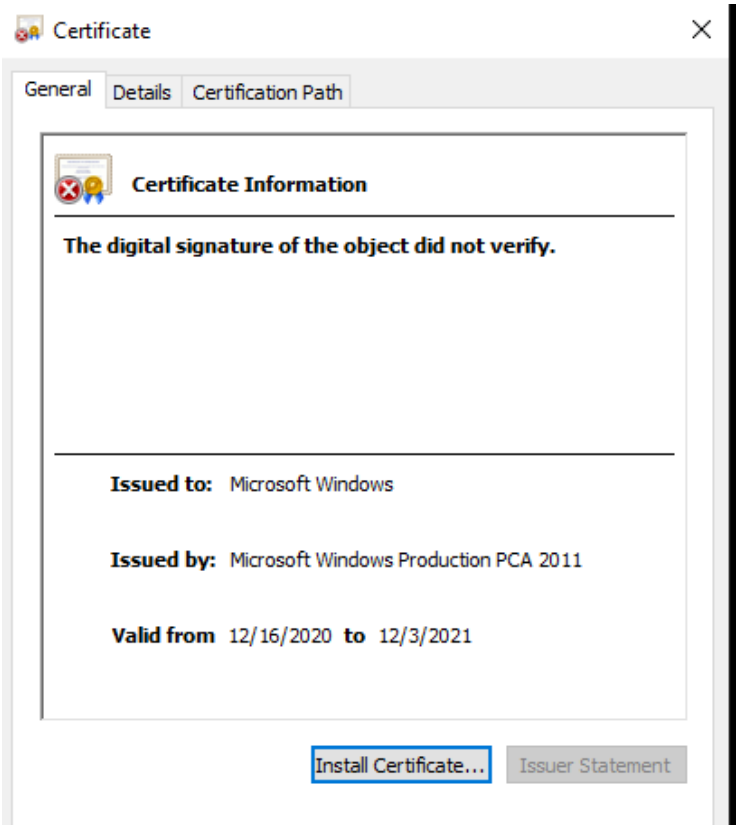


Fig 18. Fake certificate issued by Microsoft

Sandbox Report

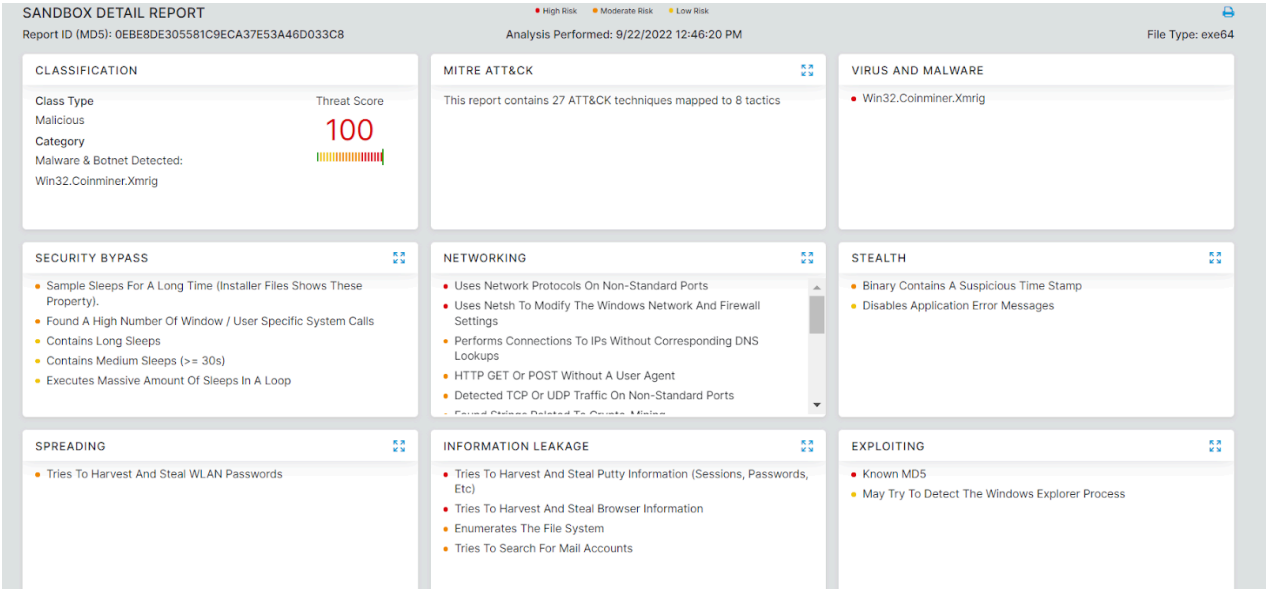


Fig 19. Zscaler Sandbox report

Zscaler's multilayered cloud security platform detects indicators, as shown below:

Win64.PWS.LilithBot

MITRE ATT&CK

ID	Tactic	Technique
T1003	Credential Access	OS Credential Dumping
T1552.002	Credential Access	Credentials in Registry
T1114.002	Collection	Remote Email Collection
T1005	Collection	Data from Local System
T1204	User Execution	User interaction
T1268	Conduct social engineering	Uses social eng to install payload
T1222	Defense Evasion	File Directory Permissions Modification
T1027	Defense Evasion	Obfuscated Files or Information
T1016	Discovery	System Network Configuration Discovery
T1012	Discovery	Query Registry
T1018	Discovery	Remote System Discovery
T1057	Discovery	Process Discovery
T1047	Execution	Windows Management Instrumentation
T1059	Execution	Command and Scripting Interpreter
T1037.005	Persistence, Privilege Escalation	Startup Items

T1071	Command and Control	Application Layer Protocol

Indicators of Compromise (IOCs)

0ebe8de305581c9eca37e53a46d033c8	Executable using microsoft signed certificate
1cae8559447370016ff20da8f717db53	Executable using microsoft signed certificate
e793fcd5e44422313ec70599078adbdc	Executable File
65c0241109562662f4398cff77499b25	Dll File using microsoft signed certificate
77.73.133.12	C&C
45.9.148.203	C&C
91.243.59.210	C&C
195.2.71.214	C&C

Explore more Zscaler blogs
