← Blog

Sharmine Low

Malware Analyst, APAC

APT Lazarus: Eager Crypto Beavers, Video calls and Games

Explore the growing threats posed by the Lazarus Group's financially-driven campaign against developers. We will examine their recent Python scripts, including the CivetQ and BeaverTail malware variants, along with their updated versions in Windows and Python releases. Additionally, we will analyze their tactics, techniques, and indicators of compromise.

September 4, 2024 · min to read · Advanced Persistent Threats

Lazarus APT Malware analysis Python Scripts Threat Intelligence

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.

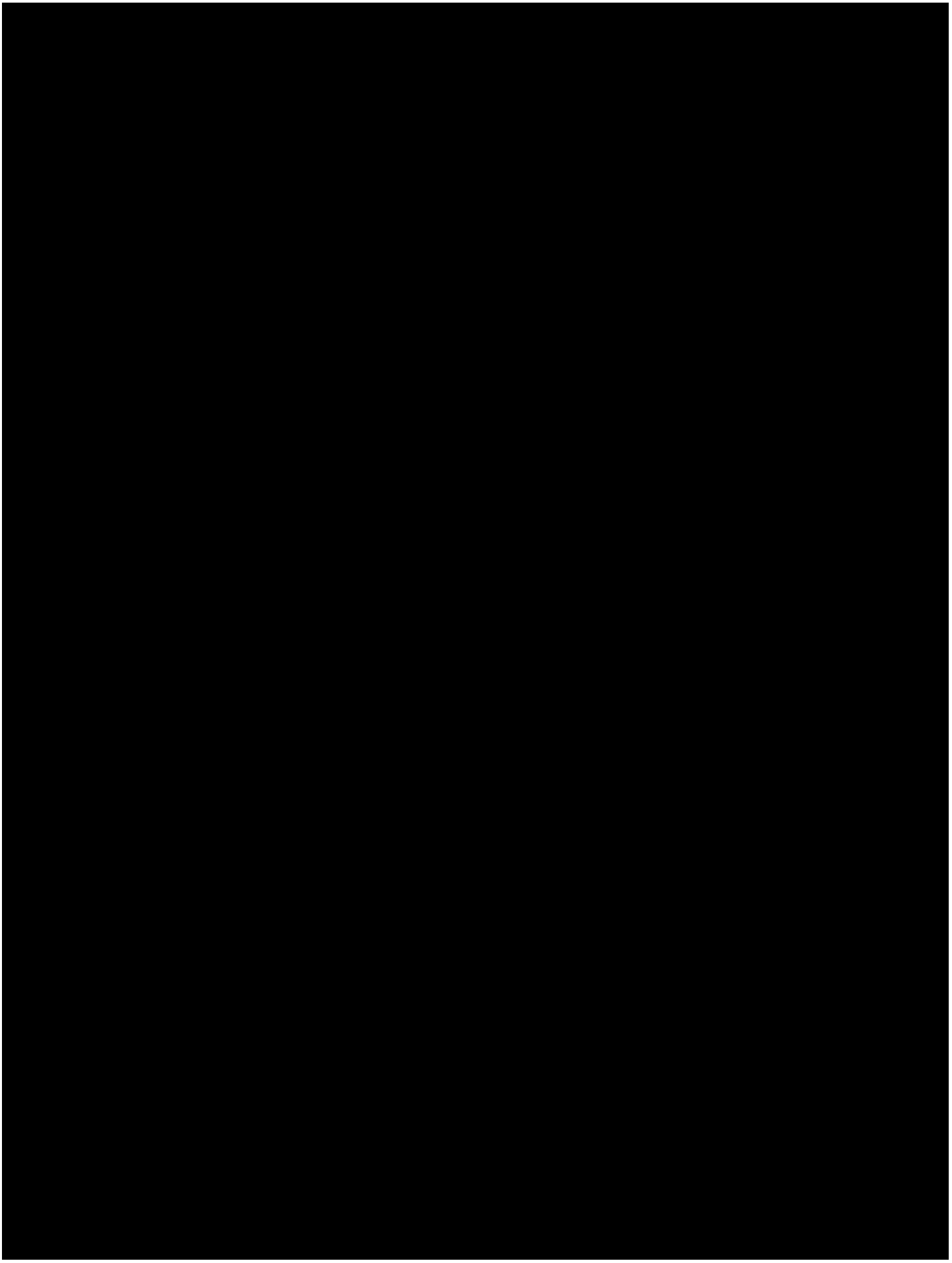
Telegram was added as an additional data exfiltration method.

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.**



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.

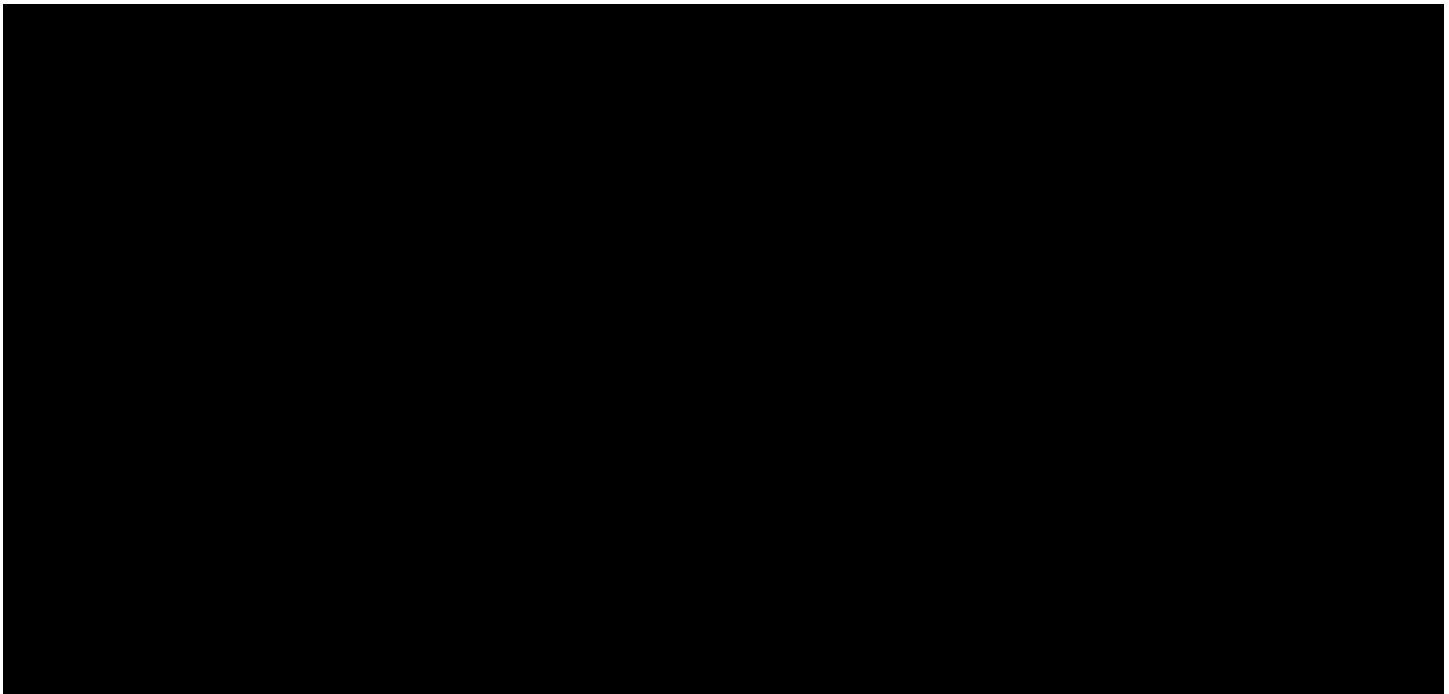


Figure 1: Chain of events leading to compromise.

The FCCCall file is a video conferencing call application installer—possibly downloaded from **hxxp://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.

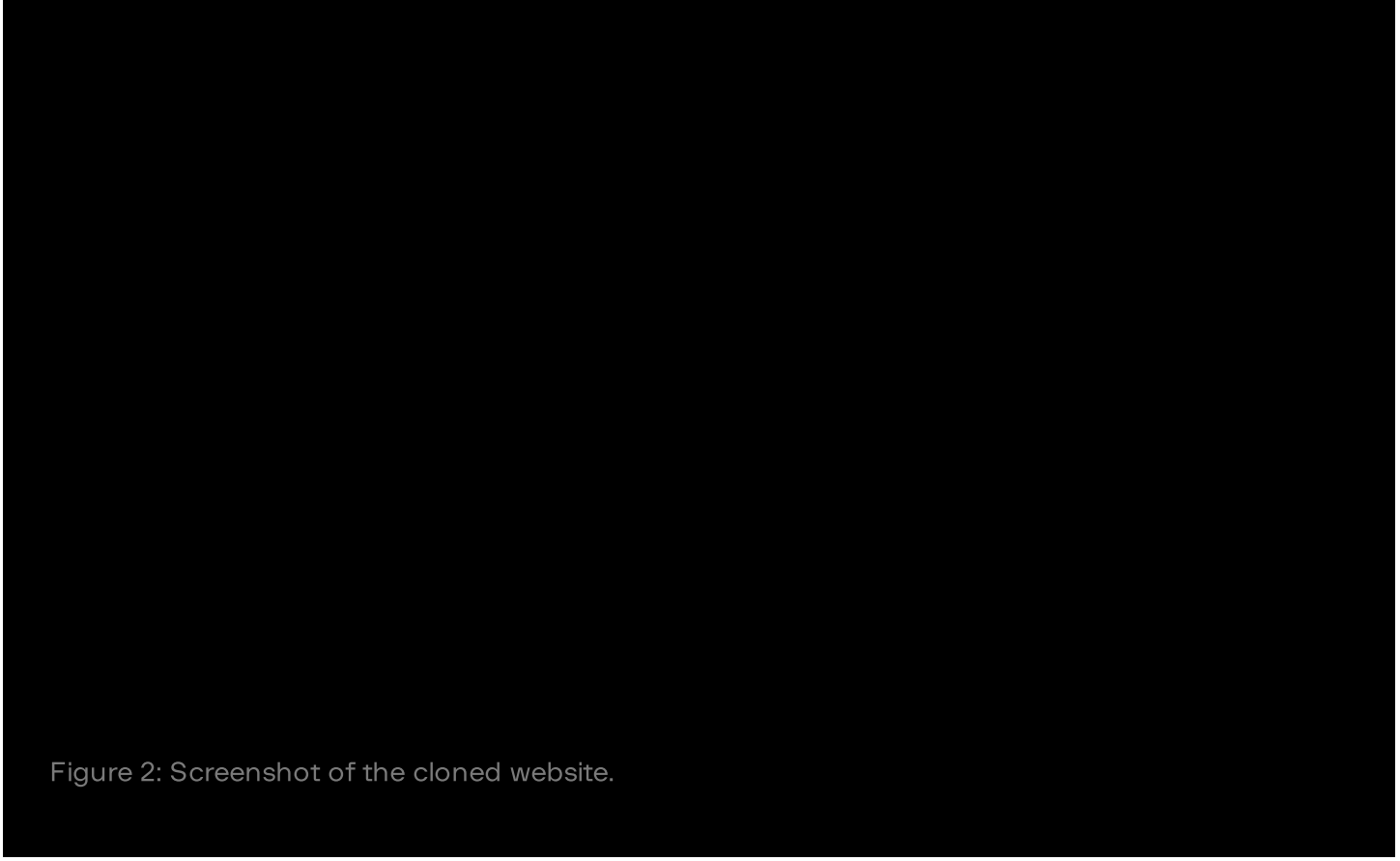


Figure 2: Screenshot of the cloned website.

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.

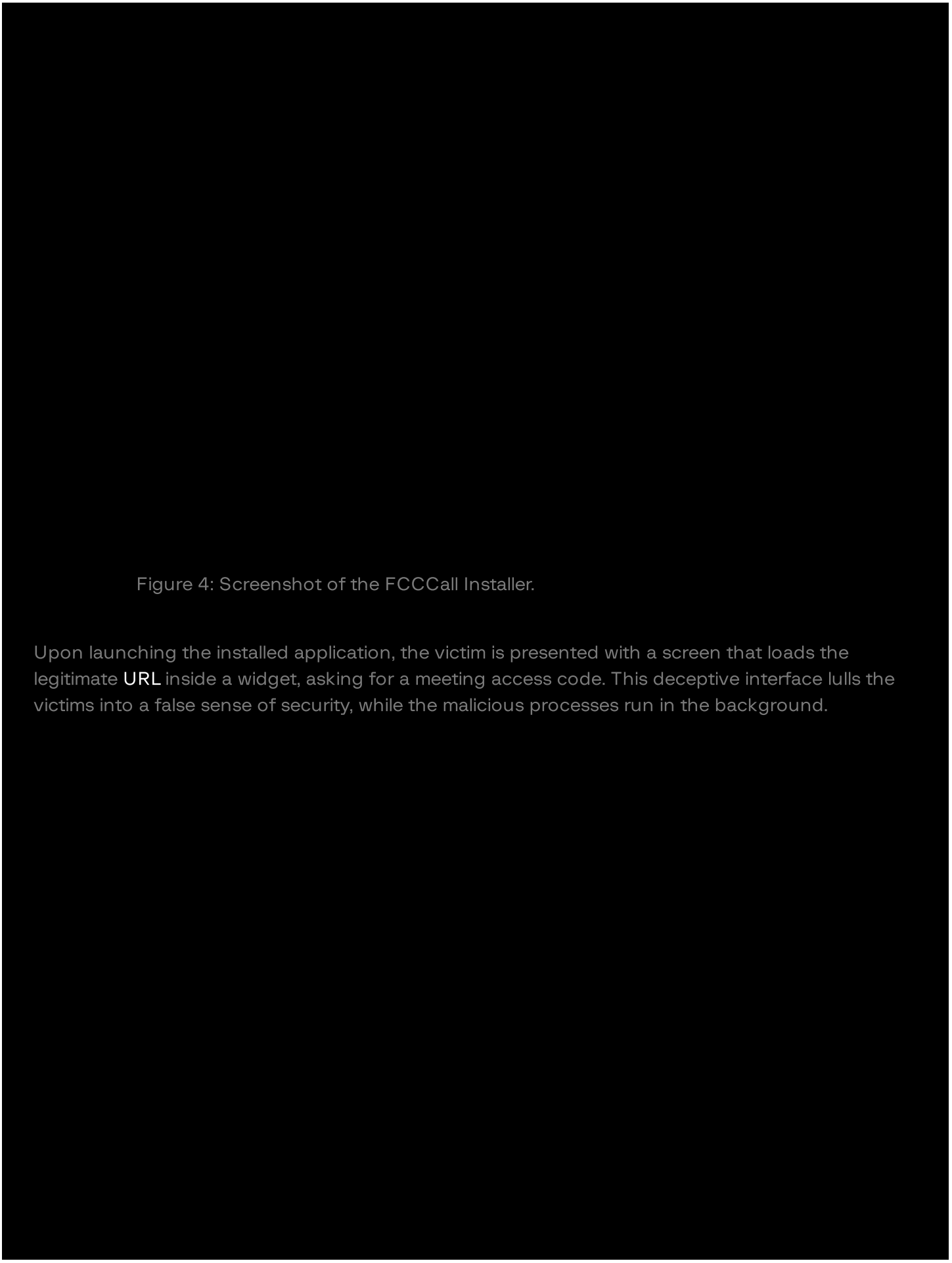


Figure 4: Screenshot of the FCCCall Installer.

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
ejbalbakoplchlghecdalmeeeajnimhm	Metamask (Edge)
fhbohimaelbohpjbbldcngcnapndodjp	BNB Chain Wallet
hnfanknocfeofbddgcijnmhnfnkdnaad	Coinbase
ibnejdfjmmkpcnlpebklmnkoeoihofec	TronLink
bfnaelmomeimhlpmgjnjophhpkkoljpa	Phantom
aeachknmefphepccionboohckonoeemg	Coin98
hifafgmccdpekplomjjkcfgodnhcellj	Crypto.com
iblndlipeogpafnldhgmapagcccfchpi	Kaia

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.

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.

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

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.

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
.q2	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 persistence 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/PropertyL
<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.sq*` 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

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.

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 http://[host]:[port]/brow/[campaign_id]/
ssh_upload	Upload directories and files specified in given command
ssh_kill	Kill Chrome and Brave browser processes
ssh_any	Download AnyDesk from http://[host]:[port]/adc/[campaign_id]
	Collect and upload folders via FTP

Table 8: Commands available for InvisibleFerret ‘pay’ script.

Figure 16: Snippet of targeted data.

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

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-

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.

MITRE ATT&CK

Indicators of Compromise

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

Consolidated Network IoCs

Annex A

Extensions

Annex B

Extensions

Application Data

Share this article

Found it interesting? Don't hesitate to share it to wow your friends or colleagues