Threat Intelligence

Coruna: The Mysterious Journey of a Powerful iOS Exploit Kit

March 3, 2026

Google Threat Intelligence Group

Introduction

Google Threat Intelligence Group (GTIG) has identified a new and powerful exploit kit targeting Apple iPhone models running iOS version 13.0 (released in September 2019) up to version 17.2.1 (released in December 2023). The exploit kit, named “Coruna” by its developers, contained five full iOS exploit chains and a total of 23 exploits. The core technical value of this exploit kit lies in its comprehensive collection of iOS exploits, with the most advanced ones using non-public exploitation techniques and mitigation bypasses.

The Coruna exploit kit provides another example of how sophisticated capabilities proliferate. Over the course of 2025, GTIG tracked its use in highly targeted operations initially conducted by a customer of a surveillance vendor, then observed its deployment in watering hole attacks targeting Ukrainian users by UNC6353, a suspected Russian espionage group. We then retrieved the complete exploit kit when it was later used in broad-scale campaigns by UNC6691, a financially motivated threat actor operating from China. How this proliferation occurred is unclear, but suggests an active market for "second hand" zero-day exploits. Beyond these identified exploits, multiple threat actors have now acquired advanced exploitation techniques that can be re-used and modified with newly identified vulnerabilities.

Following our disclosure policy, we are sharing our research to raise awareness and advance security across the industry. We have also added all identified websites and domains to Safe Browsing to safeguard users from further exploitation. The Coruna exploit kit is not effective against the latest version of iOS, and iPhone users are strongly urged to update their devices to the latest version of iOS. In instances where an update is not possible, it is recommended that Lockdown Mode be enabled for enhanced security.

Discovery Timeline

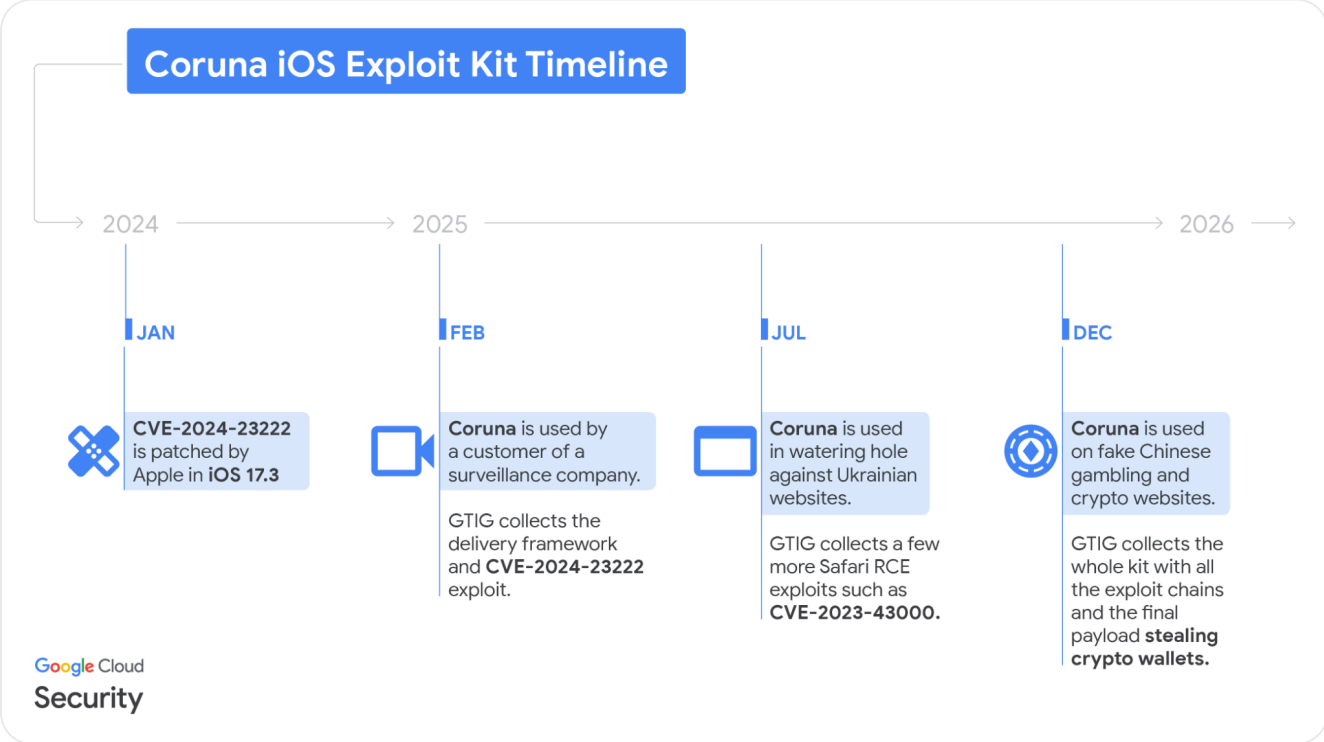

Figure 1: Coruna iOS exploit kit timeline

Initial Discovery: The Commercial Surveillance Vendor Role

In February 2025, we captured parts of an iOS exploit chain used by a customer of a surveillance company. The exploits were integrated into a previously unseen JavaScript framework that used simple but unique JavaScript obfuscation techniques.

```
[16, 22, 0, 69, 22, 17, 23, 12, 6, 17].map(x => {return String.fromC
```

```
i.p1=(1111970405 ^ 1111966034);
```

The JavaScript framework used these constructs to encode strings and integers

The framework starts a fingerprinting module collecting a variety of data points to determine if the device is real and what specific iPhone model and iOS software version it is running. Based on the collected data, it loads the appropriate WebKit remote code execution (RCE) exploit, followed by a pointer authentication code (PAC) bypass as seen in Figure 2 from the deobfuscated JavaScript.

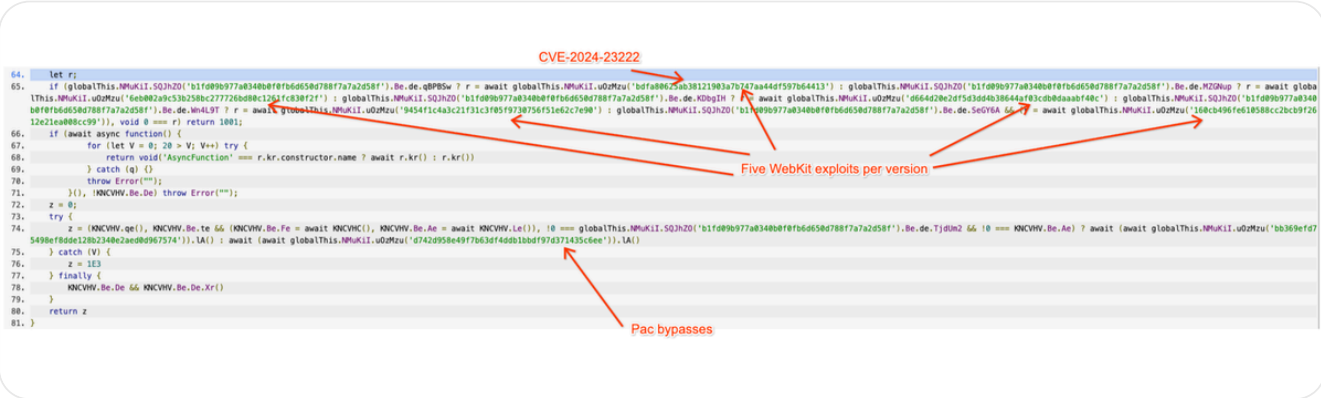

Figure 2: Deobfuscated JavaScript of the Coruna exploit kit

At that time, we recovered the WebKit RCE delivered to a device running iOS 17.2 and determined it was CVE-2024-23222, a vulnerability previously identified as a zero-day that was addressed by Apple on Jan. 22, 2024 in iOS 17.3 without crediting any external researchers. Figure 3 shows the beginning of the RCE exploit exactly how it was delivered in-the-wild with our annotations.

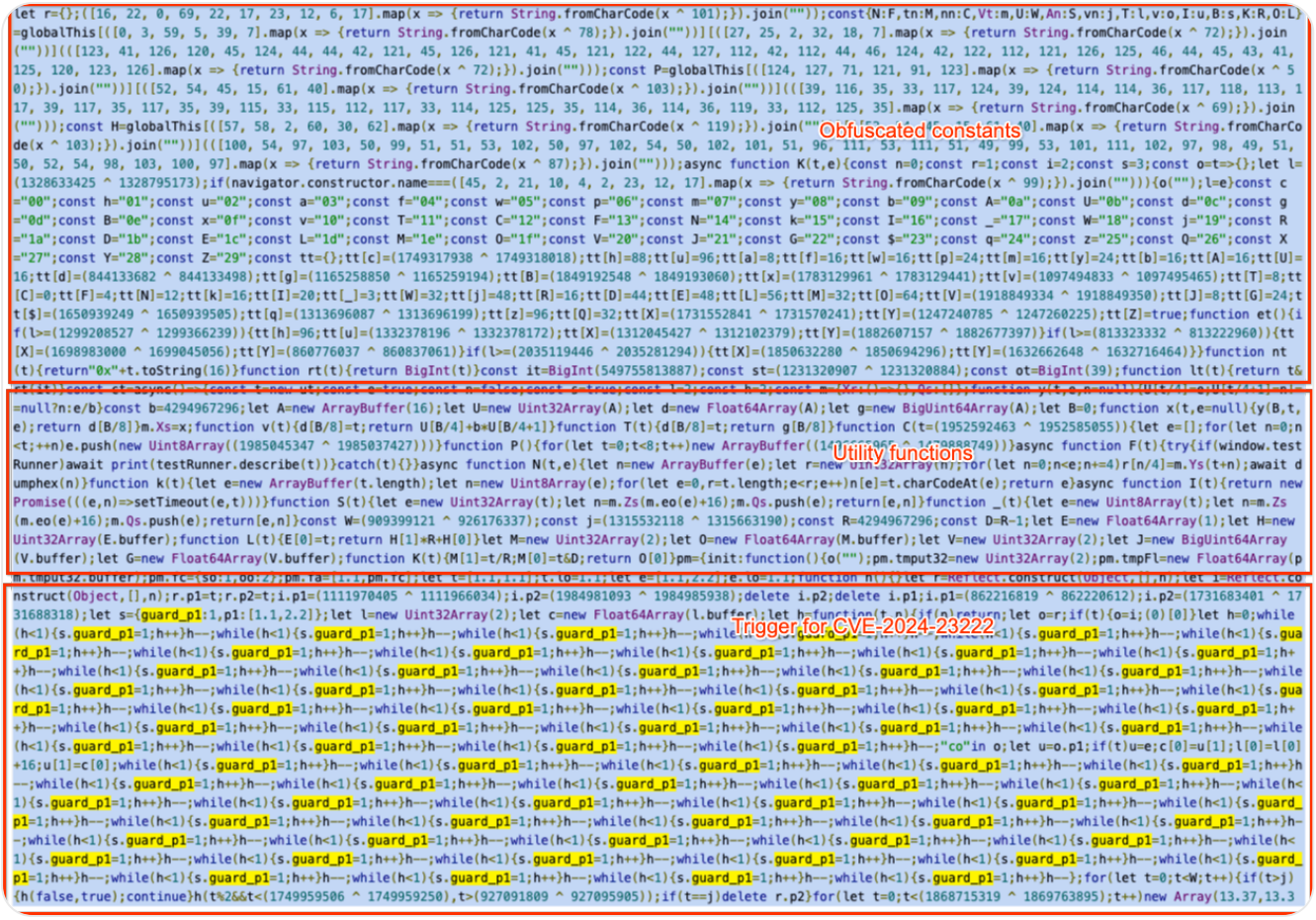

Figure 3: How the RCE exploit leveraging CVE-2024-23222 was delivered in the wild

Government-Backed Attacker Usage

In summer 2025, we noticed the same JavaScript framework hosted on cdn.uacounter[.]com, a website loaded as a hidden iFrame on many compromised Ukrainian websites, ranging from industrial equipment and retail tools to local services and ecommerce websites. The framework was only delivered to selected iPhone users from a specific geolocation.

The framework was identical and delivered the same set of exploits. We collected WebKit RCEs, which included CVE-2024-23222, CVE-2022-48503, and CVE-2023-43000, before the server was shut down. We alerted and worked with CERT-UA to clean up all compromised websites.

Full Exploit Chain Collection From Chinese Scam Websites

At the end of the year, we identified the JavaScript framework on a very large set of fake Chinese websites mostly related to finance, dropping the exact same iOS exploit kit. The websites tried to convince users to visit the websites with iOS devices, as seen in Figure 4, taken from a fake WEEX crypto exchange website.

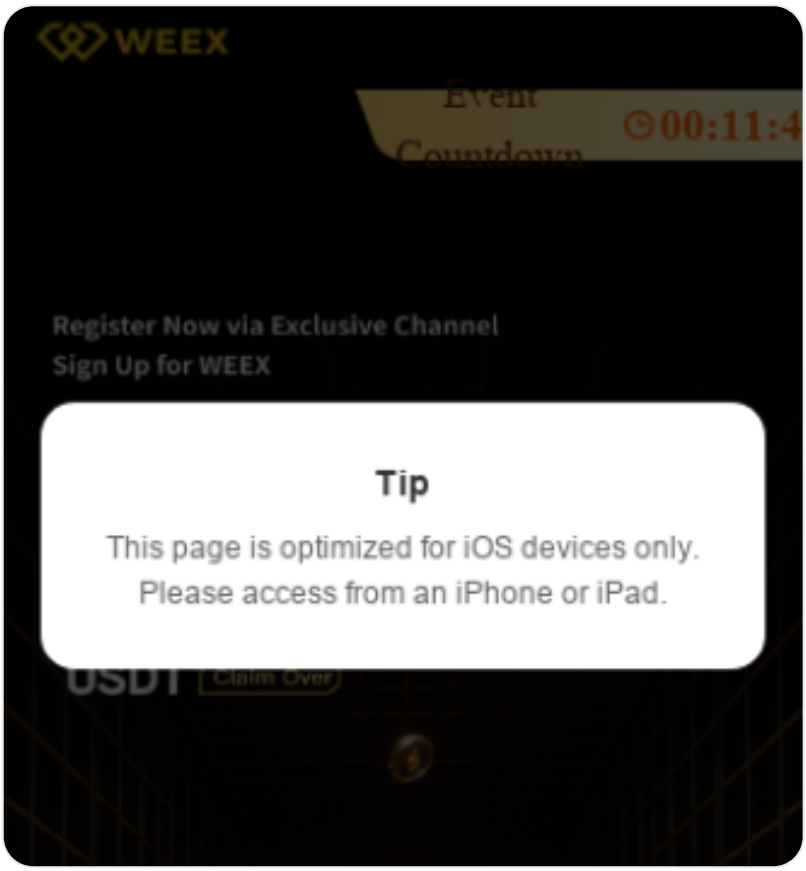

Figure 4: Pop-up on a fake cryptocurrency exchange website trying to drive users to the exploits

Upon accessing these websites via an iOS device and regardless of their geolocation, a hidden iFrame is injected, delivering the exploit kit. As an example, Figure 5 shows the same CVE-2024-23222 exploit as it was found on 3v5w1km5gv[.]xyz.

Figure 5: Screenshot of CVE-2024-23222 exploit recovered from a scam site

We retrieved all the obfuscated exploits, including ending payloads. Upon further analysis, we noticed an instance where the actor deployed the debug version of the exploit kit, leaving in the clear all of the exploits, including their internal code names. That's when we learned that the exploit kit was likely named Coruna internally. In total, we collected a few hundred samples covering a total of five full iOS exploit chains. The exploit kit is able to target various iPhone models running iOS version 13.0 (released in September 2019) up to version 17.2.1 (released in December 2023).

In the subsequent sections, we will provide a quick description of the framework, a breakdown of the exploit chains, and the associated implants we have captured. Our analysis of the collected data is ongoing, and we anticipate publishing additional technical specifications via new blog entries or root cause analyses (RCAs).

The Coruna Exploit Kit

The framework surrounding the exploit kit is extremely well engineered; the exploit pieces are all connected naturally and combined together using common utility and exploitation frameworks. The kit performs the following unique actions:

- Bailing out if the device is in Lockdown Mode, or the user is in private browsing.

- A unique and hard-coded cookie is used along the way to generate resource URLs.
- Resources are referred to by a hash, which needs to be derived with the unique cookie using `sha256(COOKIE + ID)[:40]` to get their URL.
- RCE and PAC bypasses are delivered unencrypted.

The kit contains a binary loader to load the appropriate exploit chain post RCE within WebKit. In this case, binary payloads:

- Have unique metadata indicating what they really are, what chips and iOS versions they support.
- Are served from URLs that end with .min.js.
- Are encrypted using ChaCha20 with a unique key per blob.
- Are packaged in a custom file format starting with 0xf00dbeef as header.
- Are compressed with the Lempel–Ziv–Welch (LZW) algorithm.

Figure 6 shows what an infection of an iPhone XR running iOS 15.8.5 looks like from a networking point of view, with our annotation of the different parts when browsing one of these fake financial websites.

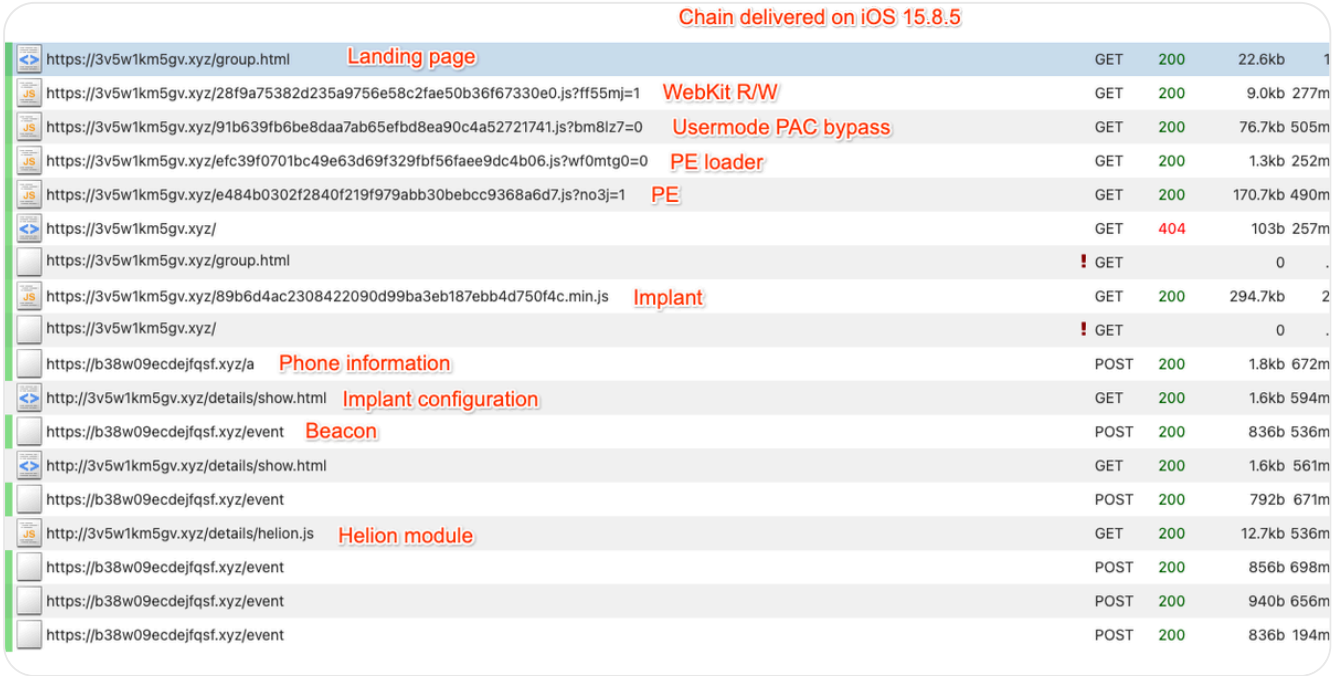

Figure 6: Coruna exploit chain delivered on iOS 15.8.5

The Exploits and Their Code Names

The core technical value of this exploit kit lies in its comprehensive collection of iOS exploits. The exploits feature extensive documentation, including docstrings and comments authored in native English. The most advanced ones are using non-public exploitation techniques and mitigation bypasses. The following table provides a summary of our ongoing analysis regarding the various exploit chains; however, as the full investigation is still in progress, certain CVE associations may be subject to revision. There are in total 23 exploits covering versions from iOS 13 to iOS 17.2.1.

Type	Codename	Targeted versions (inclusive)	Fixed version	CVE
WebContent R/W	buffout	13 → 15.1.1	15.2	CVE-2021-30952
WebContent R/W	jacurutu	15.2 → 15.5	15.6	CVE-2022-48503
WebContent R/W	bluebird	15.6 → 16.1.2	16.2	No CVE
WebContent R/W	terrorbird	16.2 → 16.5.1	16.6	CVE-2023-43000
WebContent R/W	cassowary	16.6 → 17.2.1	16.7.5, 17.3	CVE-2024-23222
WebContent PAC bypass	breezy	13 → 14.x	?	No CVE

WebContent PAC bypass	breezy15	15 → 16.2	?	No CVE
WebContent PAC bypass	seedbell	16.3 → 16.5.1	?	No CVE
WebContent PAC bypass	seedbell_16_6	16.6 → 16.7.12	?	No CVE
WebContent PAC bypass	seedbell_17	17 → 17.2.1	?	No CVE
WebContent sandbox escape	IronLoader	16.0 → 16.3.116.4.0 (<= A12)	15.7.8, 16.5	CVE-2023-32409
WebContent sandbox escape	NeuronLoader	16.4.0 → 16.6.1 (A13-A16)	17.0	No CVE
PE	Neutron	13.X	14.2	CVE-2020-27932
PE (infoleak)	Dynamo	13.X	14.2	CVE-2020-27950
PE	Pendulum	14 → 14.4.x	14.7	No CVE
PE	Photon	14.5 → 15.7.6	15.7.7, 16.5.1	CVE-2023-32434

PE	Parallax	16.4 → 16.7	17.0	CVE-2023-41974
PE	Gruber	15.2 → 17.2.1	16.7.6, 17.3	No CVE
PPL Bypass	Quark	13.X	14.5	No CVE
PPL Bypass	Gallium	14.x	15.7.8, 16.6	CVE-2023-38606
PPL Bypass	Carbone	15.0 → 16.7.6	17.0	No CVE
PPL Bypass	Sparrow	17.0 → 17.3	16.7.6, 17.4	CVE-2024-23225
PPL Bypass	Rocket	17.1 → 17.4	16.7.8, 17.5	CVE-2024-23296

Table 1: Table with mapping CVE to code names

Photon and Gallium are exploiting vulnerabilities that were also used as zero-days as part of Operation Triangulation, discovered by Kaspersky in 2023. The Coruna exploit kit also embeds reusable modules to ease the exploitation of the aforementioned vulnerabilities. For example, there is a module called `rwx_allocator` using multiple techniques to bypass various mitigations preventing allocation of RWX memory pages in userland. The kernel exploits are also embedding various internal modules allowing them to bypass kernel-based mitigations such as kernel-mode PAC.

The Ending Payload

At the end of the exploitation chain, a stager binary called `PlasmaLoader` (tracked by GTIG as PLASMAGRID), using com.apple.assistd as an identifier, facilitates communication with the kernel component established by the exploit. The loader is injecting itself into powerd, a daemon running as root on iOS.

The injected payload doesn't exhibit the usual capabilities that we would expect to see from a surveillance vendor, but instead steals financial information. The payload can decode QR codes from images on disk. It also has a module to analyze blobs of text to look for BIP39 word sequences or very specific keywords like "backup phrase" or "bank account." If such text is found in Apple Memos it will be sent back to the C2.

More importantly, the payload has the ability to collect and run additional modules remotely, with the configuration retrieved from `http://<C2 URL>/details/show.html` . The configuration, as well as the additional modules, are compressed as 7-ZIP archives protected with a unique hard-coded password. The configuration is encoded in JSON and simply contains a list of module names with their respective URL, hash and size.


```
{
  "entries": [
    {
      "bundleId": "com.bitkeep.os",
      "url": "http://<C2URL>/details/f6lib.js",
      "sha256": "6eafd742f58db21fbaf5fd7636e6653446df04b4a5c9bca9104
      "size": 256832,
      "flags": {
        "do_not_close_after_run": true
      }
    }
...
  ]
}
```

As expected, most of all identified modules exhibit a uniform design; they are all placing function hooks for the purpose of exfiltrating cryptocurrency wallets or sensitive information from the following applications:

- `com.bitkeep.os`
- `com.bitpie.wallet`
- `coin98.crypto.finance.insights`
- `org.toshi.distribution`
- `exodus-movement.exodus`
- `im.token.app`
- `com.kyrd.krystal.ios`
- `io.metamask.MetaMask`
- `org.mytonwallet.app`
- `app.phantom`
- `com.skymavis.Genesis`
- `com.solflare.mobile`
- `com.global.wallet.ios`
- `com.tonhub.app`
- `com.jbig.tonkeeper`
- `com.tronlink.hdwallet`
- `com.sixdays.trust`
- `com.uniswap.mobile`

All of these modules contain proper logging with sentences written in Chinese:

```
<PlasmaLogger> %s[%d]: CorePayload 管理器初始化成功，尝试启动...
```

This log string indicates the CorePayload Manager initialized successfully

Some comments, such as the following one, also include emojis and are written in a way suggesting they might be LLM-generated.

```
<PlasmaLogger> %s[%d]: [PLCoreHeartbeatMonitor] ✅ 心跳监控已启动（端
```

Network communication is done over HTTPs with the collected data encrypted and POST'ed with AES using the SHA256 hash of a static string as key. Some of the HTTP requests contain additional HTTP headers such as `sdkv` or `x-ts,` followed by a timestamp. The implant contains a list of hard-coded C2s but has a fallback mechanism in case the servers do not respond. The implant embeds a custom domain generation algorithm (DGA) using the string "lazarus" as seed to generate a list of predictable domains. The domains will have 15 characters and use .xyz as TLD. The attackers use Google's public DNS resolver to validate if the domains are active.

Conclusion

Google has been a committed participant in the Pall Mall Process, designed to build consensus and progress toward limiting the harms from the spyware industry. Together, we are focused on developing international norms and frameworks to limit the misuse of these powerful technologies and protect human rights around the world. These efforts are built on earlier governmental actions, including steps taken by the US Government to limit government use of spyware, and a first-of-its-kind international commitment to similar efforts.

Acknowledgements

We would like to acknowledge and thank Google Project-Zero and Apple Security Engineering & Architecture team for their partnership throughout this investigation.

Indicators of Compromise (IOCs)

To assist the wider community in hunting and identifying activity outlined in this blog post, we have included IOCs in a free GTI Collection for registered users.

File Indicators

Hashes of the implant and its modules delivered from the crypto related websites.

Implant

bundleId	SHA-256
com.apple.assistd	2a9d21ca07244932939c6c58699448f2147992c1f49c

Modules

bundleId	SHA-256
com.apple.springboard	18394fcc096344e0730e49a0098970b1
com.bitkeep.os	6eafd742f58db21fbaf5fd7636e6653
com.bitpie.wallet	42cc02cecd65f22a3658354c5a5efa6
coin98.crypto.finance.insights	0dff17e3aa12c4928273c70a2e0a6ff
org.toshi.distribution	05b5e4070b3b8a130b12ea96c5526b4
exodus-movement.exodus	10bd8f2f8bb9595664bb9160fbc4136
im.token.app	91d44c1f62fd863556aac0190cbef3b4
com.kyrd.krystal.ios	721b46b43b7084b98e51ab00606f08a

io.metamask.MetaMask	25a9b004cf61fb251c8d4024a8c7383a
org.mytonwallet.app	be28b40df919d3fa87ed49e51135a719
app.phantom	3c297829353778857edfeaed3ceeeca1
com.skymavis.Genesis	499f6b1e012d9bc947eea8e23635dfe6
com.solflare.mobile	d517c3868c5e7808202f53fa78d827a3
com.global.wallet.ios	4dfcf5a71e5a8f27f748ac7fd7760dec
com.tonhub.app	d371e3bed18ee355438b166bbf3bdaf2
com.jbig.tonkeeper	023e5fb71923cfa2088b9a48ad8566ft
com.tronlink.hdwallet	f218068ea943a511b230f2a99991f6d1
com.sixdays.trust	1fb9dedf1de81d387eff4bd5e747f730
com.uniswap.mobile	4dc255504a6c3ea8714ccdc95cc04138

Network Indicators

UNC6353 Indicators

URL delivering Coruna exploit kit
http://cdn[.]uacounter[.]com/stat[.]html

UNC6691 Indicators

URLs delivering Coruna exploit kit
https://ai-scorepredict[.]com/static/analytics[.]html
https://m[.]pc6[.]com/test/tuiliu/group[.]html
http://ddus17[.]com/tuiliu/group[.]html
https://goodcryptocurrency[.]top/details/group[.]html
http://pepeairdrop01[.]com/static/analytics[.]html
https://osec2[.]668ddf[.]cc/tuiliu/group[.]html
https://pepeairdrop01[.]com/static/analytics[.]html
https://ios[.]teegrom[.]top/tuiliu/group[.]html
https://i[.]binaner[.]com/group[.]html
https://ajskbnrs[.]xn--jor0b302fdhgwnccw8g[.]com/gogo/list[.]html
https://sj9ioz3a7y89cy7[.]xyz/list[.]html
https://65sse[.]668ddf[.]cc/tuiliu/group[.]html
https://sadjd[.]mijieqi[.]cn/group[.]html
https://mkkku[.]com/static/analytics[.]html
https://dbgopaxl[.]com/static/goindex/tuiliu/group[.]html

https://w2a315[.]tubeluck[.]com/static/goindex/tuiliu/group[.]html
https://ose[.]668ddf[.]cc/tuiliu/group[.]html
http://cryptocurrencyworld[.]top/details/group[.]html
https://iphonex[.]mjdqw[.]cn/tuiliu/group[.]html
http://goodcryptocurrency[.]top/details/group[.]html
https://share[.]4u[.]game/group[.]html
https://26a[.]online/group[.]html
https://binancealliancesintro[.]com/group[.]html
https://4u[.]game/group[.]html
http://bestcryptocurrency[.]top/details/group[.]html
https://b27[.]icu/group[.]html
https://h4k[.]icu/group[.]html
https://so5083[.]tubeluck[.]com/static/goindex/group[.]html
https://seven7[.]vip/group[.]html
https://y4w[.]icu/group[.]html
https://7ff[.]online/group[.]html
https://cy8[.]top/group[.]html

https://7uspin[.]us/group[.]html
https://seven7[.]to/group[.]html
https://4kgame[.]us/group[.]html
https://share[.]7p[.]game/group[.]html
https://www[.]appstoreconn[.]com/xmweb/group[.]html
https://k96[.]icu/group[.]html
https://7fun[.]icu/group[.]html
https://n49[.]top/group[.]html
https://98a[.]online/group[.]html
https://spin7[.]icu/group[.]html
https://t7c[.]icu/group[.]html
https://7p[.]game/group[.]html
https://lddx3z2d72aa8i6[.]xyz/group[.]html
https://anygg[.]liquorfight[.]com/88k4ez/group[.]html
https://goanalytics[.]xyz/88k4ez/group[.]html
http://land[.]77bingos[.]com/88k4ez/group[.]html
https://land[.]bingo777[.]now/88k4ez/group[.]html

http://land[.]bingo777[.]now/88k4ez/group[.]html
http://land[.]777bingos[.]xyz/88k4ez/group[.]html
https://btrank[.]top/tuiliu/group[.]html
https://dd9l7e6ghme8pbk[.]xyz/group[.]html
https://res54allb[.]xn--xkrsa0078bd6d[.]com/group[.]html
https://fxrhcnfwxes90q[.]xyz/group[.]html
https://kanav[.]blog/group[.]html
https://3v5w1km5gv[.]xyz/group[.]html

PLASMAGRID C2 domains
vvri8ocl4t3k8n6.xyz
rlau616jc7a7f7i.xyz
ol67el6pxg03ad7.xyz
6zvjeulzaw5c0mv.xyz
ztvnhmhm4zj95w3.xyz
v2gmupm7o4zihc3.xyz
pen0axt0u476duw.xyz

hfteigt3kt0sf3z.xyz
xfal48cf0ies7ew.xyz
yvgy29glwf72qnl.xyz
lk4x6x2ejxaw2br.xyz
2s3b3rknfqtwwpo.xyz
xjslbdt9jdijn15.xyz
hui4tbh9uv9x4yi.xyz
xittgveqaufogve.xyz
xmmfrkq9oat1daq.xyz
lsnngjyu9x6vcg0.xyz
gdvynopz3pa0tik.xyz
o08h5rhu2lu1x0q.xyz
zcjdlb5ubkhy41u.xyz
8fn4957c5g986jp.xyz
uawwydy3qas6ykv.xyz
sf2bisx5nhdkygn3l.xyz
roy2tlop2u.xyz

gqjs3ra34lyuvzb.xyz
eg2bjo5x5r8yjb5.xyz
b38w09ecdejfqsf.xyz

YARA Rules

```
rule G_Hunting_Exploit_MapJoinEncoder_1 {
        meta:
                author = "Google Threat Intelligence Group (GTIG)"
        strings:
                $s1 = /\[[^\]]+\]\.map\(\w\s*=>.{0,15}String\.from
                $fp1 = "bot|googlebot|crawler|spider|robot|crawlin
        condition:
                1 of ($s*) and not any of ($fp*)
}
```

```
rule G_Backdoor_PLASMAGRID_Strings_1 {
        meta:
                author = "Google Threat Intelligence Group (GTIG)"
        strings:
                $ = "com.plasma.appruntime.appdiscovery"
                $ = "com.plasma.appruntime.downloadmanager"
                $ = "com.plasma.appruntime.hotupdatemanager"
                $ = "com.plasma.appruntime.modulestore"
                $ = "com.plasma.appruntime.netconfig"
                $ = "com.plasma.bundlemapper"
                $ = "com.plasma.event.upload.serial"
                $ = "com.plasma.notes.monitor"
                $ = "com.plasma.photomonitor"
```

```
                $ = "com.plasma.PLProcessStateDetector"
                $ = "plasma_heartbeat_monitor"
                $ = "plasma_injection_dispatcher"
                $ = "plasma_ipc_processor"
                $ = "plasma_%@.jpg"
                $ = "/var/mobile/Library/Preferences/com.plasma.ph
                $ = "helion_ipc_handler"
                $ = "PLInjectionStateInfo"
                $ = "PLExploitationInterface"
        condition:
                1 of them
}
```