State-backed attackers and commercial surveillance vendors repeatedly use the same exploits

Clement Lecigne ⋮⋮ 8/29/2024

Today, we're sharing that Google's Threat Analysis Group (TAG) observed multiple in-the-wild exploit campaigns, between November 2023 and July 2024, delivered from a watering hole attack on Mongolian government websites. The campaigns first delivered an iOS WebKit exploit affecting iOS versions older than 16.6.1 and then later, a Chrome exploit chain against Android users running versions from m121 to m123. These campaigns delivered n-day exploits for which patches were available, but would still be effective against unpatched devices. We assess with moderate confidence the campaigns are linked to the Russian government-backed actor APT29. In each iteration of the watering hole campaigns, the attackers used exploits that were identical or strikingly similar to exploits previously used by commercial surveillance vendors (CSVs) Intellexa and NSO Group.

Although the underlying vulnerabilities had already been addressed, we notified both Apple and our partners at Android and Google Chrome about the campaigns at the time of discovery. We also notified the Mongolian CERT to remediate the infected websites.

This post will detail these campaigns, highlight the continued utility of watering hole attacks for sophisticated exploits, and demonstrate common exploit usage across government-backed actors and CSVs. This post also highlights Google Chrome protections such as Site Isolation, which requires attackers to chain more vulnerabilities together in order to steal all cookies.

Watering hole

The watering hole affected the cabinet.gov[.]mn and mfa.gov[.]mn websites, that were compromised to load a hidden iframe from the attacker-controlled website track-adv[.]com, which was replaced by ceo-adviser[.]com in later iterations.

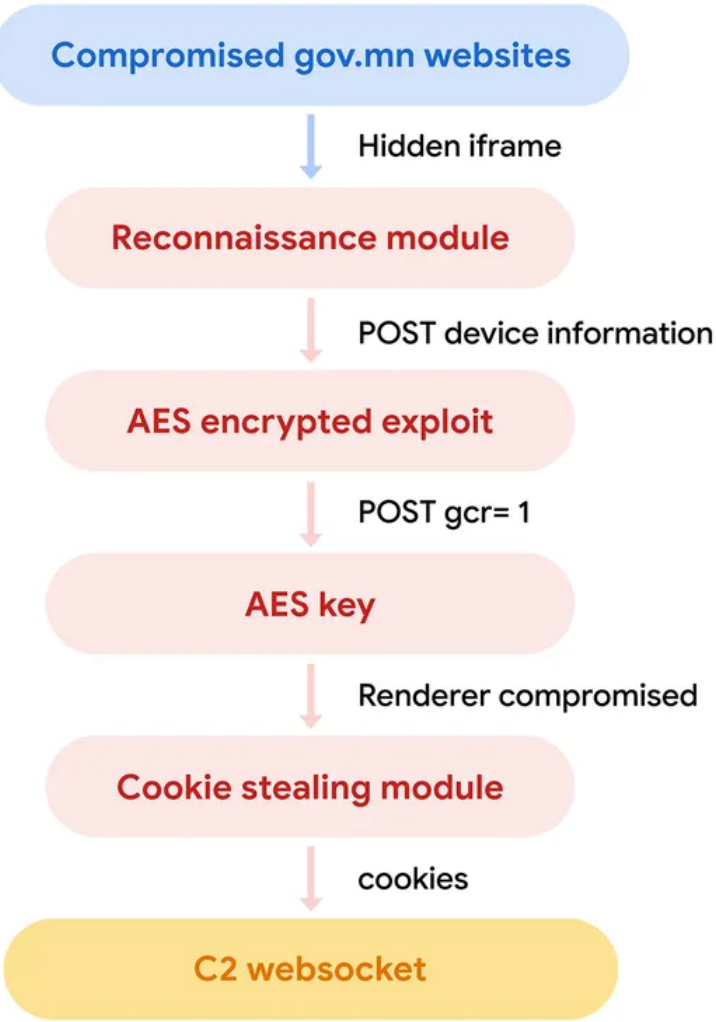
- **November 2023:** both cabinet.gov[.]mn and mfa.gov[.]mn included an iframe to https://track-adv[.]com/market-analytics.php?pc=1 that delivered the CVE-2023-41993 exploit to iPhone users running versions 16.6.1 or older. The payload was the same cookie stealer framework that TAG previously observed being used in 2021 in a suspected APT29 campaign. This is the first time it has been observed since the 2021 campaign.
- **February 2024:** mfa.gov[.]mn was compromised again to include an iframe to https://ceo-adviser[.]com/fb-connect.php?online=1 and delivered the same CVE-2023-41993 exploit to iPhone users running versions 16.6.1 or older. The cookie stealer payload remained the same, but the list of target websites had been updated. For example, cookies from webmail.mfa.gov[.]mn/owa/auth were also collected.
- **July 2024:** mfa.gov[.]mn was compromised again to include a piece of javascript redirecting Android users using Google Chrome to https://track-adv[.]com/analytics.php?personalization_id=

<random number>. The iframe delivered a Google Chrome exploit chain targeting CVE-2024-5274 and CVE-2024-4671 to deploy a Chrome information stealing payload.

Apple Safari campaign

The November 2023 and February 2024 campaigns delivered an iOS exploit via CVE-2023-41993.

- When visited with an iPhone or iPad device, the watering hole sites used an iframe to serve a reconnaissance payload, which performed validation checks before ultimately downloading and deploying another payload with the WebKit exploit to exfiltrate browser cookies from the device.
- The WebKit exploit did not affect users running the current iOS version at the time (iOS 16.7), working only on iOS versions 16.6.1 or older. Users with lockdown mode enabled were not affected even when running a vulnerable iOS version.



Attack chain used in the November 2023-February 2024 campaigns targeting iOS

Reconnaissance payload

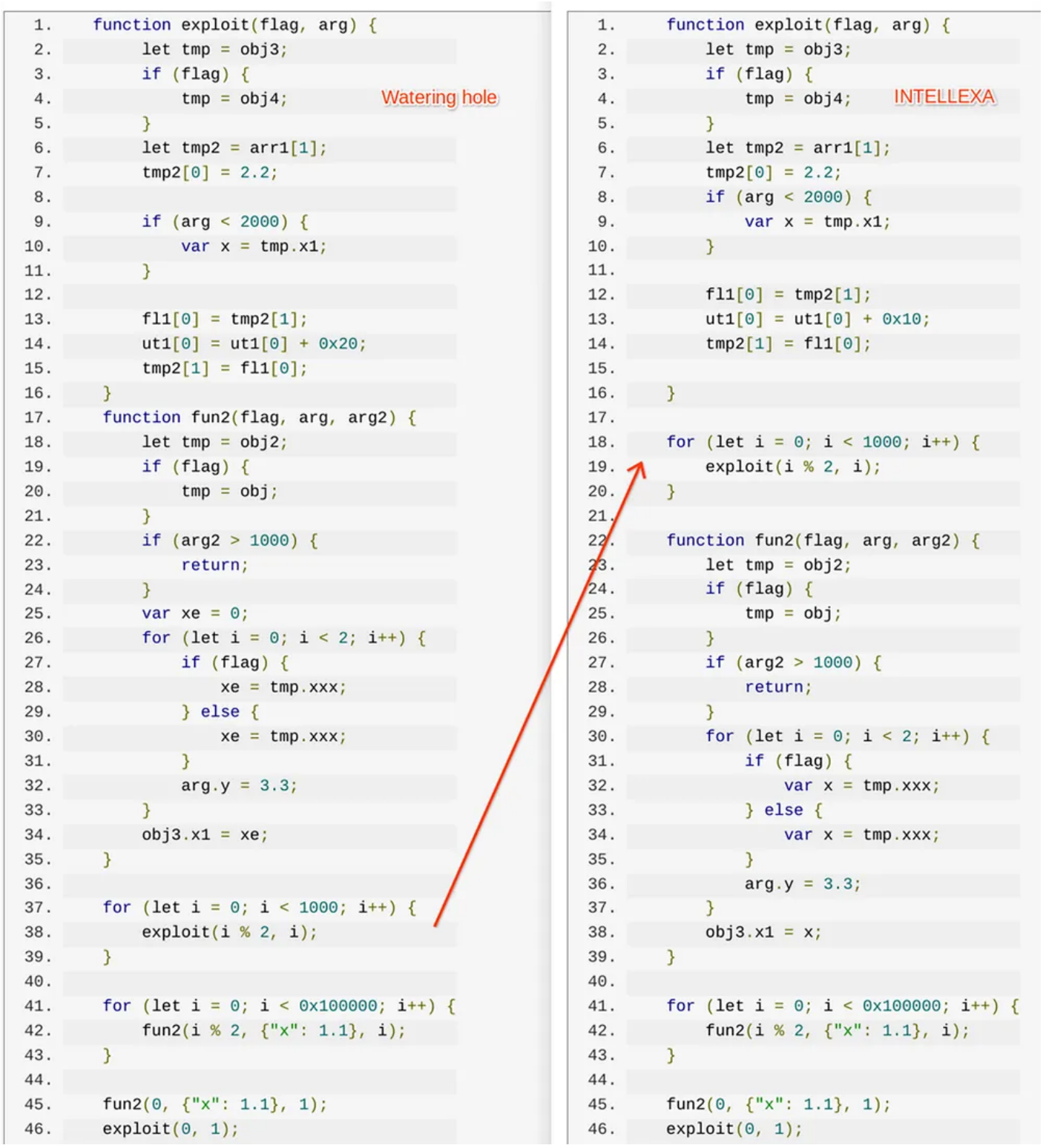
The reconnaissance payload uses a profiling framework drawing canvas to identify the target's exact iPhone model, a technique used by many other actors. The iPhone model is sent back to the C2 along

with screen size, whether or not a touch screen is present, and a unique identifier per initial GET request (e.g., 1lwuzddaxoom5ylli37v90kj).

The server replies with either an AES encrypted next stage or 0, indicating that no payload is available for this device. The payload makes another request to the exploit server with gcr=1 as a parameter to get the AES decryption key from the C2.

Exploit

The exploit from this watering hole used the exact same trigger as the exploit used by Intellexa as seen in the screenshot below, strongly suggesting the authors and/or providers are the same. We do not know how attackers in the recent watering hole campaigns acquired this exploit.



The exploits used in the November 2023 watering hole attack (left image) and by Intellexa in September 2023 (right image) share the same trigger code.

The underlying bug is an optimization problem occurring during FTL JIT compilation. Both exploits also share the same exploitation framework, which provide attackers with a set of utilities to execute arbitrary code (e.g. custom MachO loader and parser, PAC and JIT cage bypasses).

There are several minimal differences between the two exploits, which include:

- **Failure mode.** If something goes wrong during exploitation, the exploit from the watering hole will send back the information to the C2 and try to crash the browser with an out-of-memory error. If the

Intellexa exploit fails, it does not send information back and will just redirect the user to a legitimate website.

- **Additional data collection from target device.** The exploit from the watering hole has an additional function named dacsiloscope using the read/write primitives to collect even more information about the targeted device. This information is later used to decide whether or not the cookie stealer payload should be executed. For example, if the device doesn't have PAC — which might be the case for an iPhone 8 running iOS 16.X — the cookie stealer payload will simply not execute.

Cookie stealer

The iOS exploit loaded the same cookie stealer framework that TAG observed in March 2021 when a Russian government-backed attacker exploited CVE-2021-1879 to acquire authentication cookies from prominent websites such as LinkedIn, Gmail and Facebook. In that campaign, attackers used LinkedIn Messaging to target government officials from western European countries by sending them malicious links.

In the watering hole campaigns, the flow on iOS versions older than 16.6 is the same as described in the Root Cause Analysis for CVE-2021-1879. For each targeted website:

- Create a websocket w connected to an attacker-controlled IP address.
- Set m_universalAccess to 1 inside the SecurityOrigin class by traversing a set of pointers.
- Create a new URL object u pointing to the targeted domain.
- Overwrite all Document URLS of the websocket w with the ones from the u URL.
- Overwrite m_url field of the websocket w with the u URL.
- Trigger a send on the websocket.
- At the end of the websocket, the attacker receives requests as they would be delivered to the targeted websites u including the authentication cookies for the targeted websites.
- Restore m_universalAccess back to its original state.

The cookie stealer module is targeting the following hard-coded set of websites:

["webmail.mfa.gov.mn/owa/auth", "accounts.google.com", "login.microsoftonline.com", "mail.google.com/mail/mu/0", "www.linkedin.com", "linkedin.com", "www.office.com", "login.live.com", "outlook.live.com", "login.yahoo.com", "mail.yahoo.com", "facebook.com", "github.com", "icloud.com"]

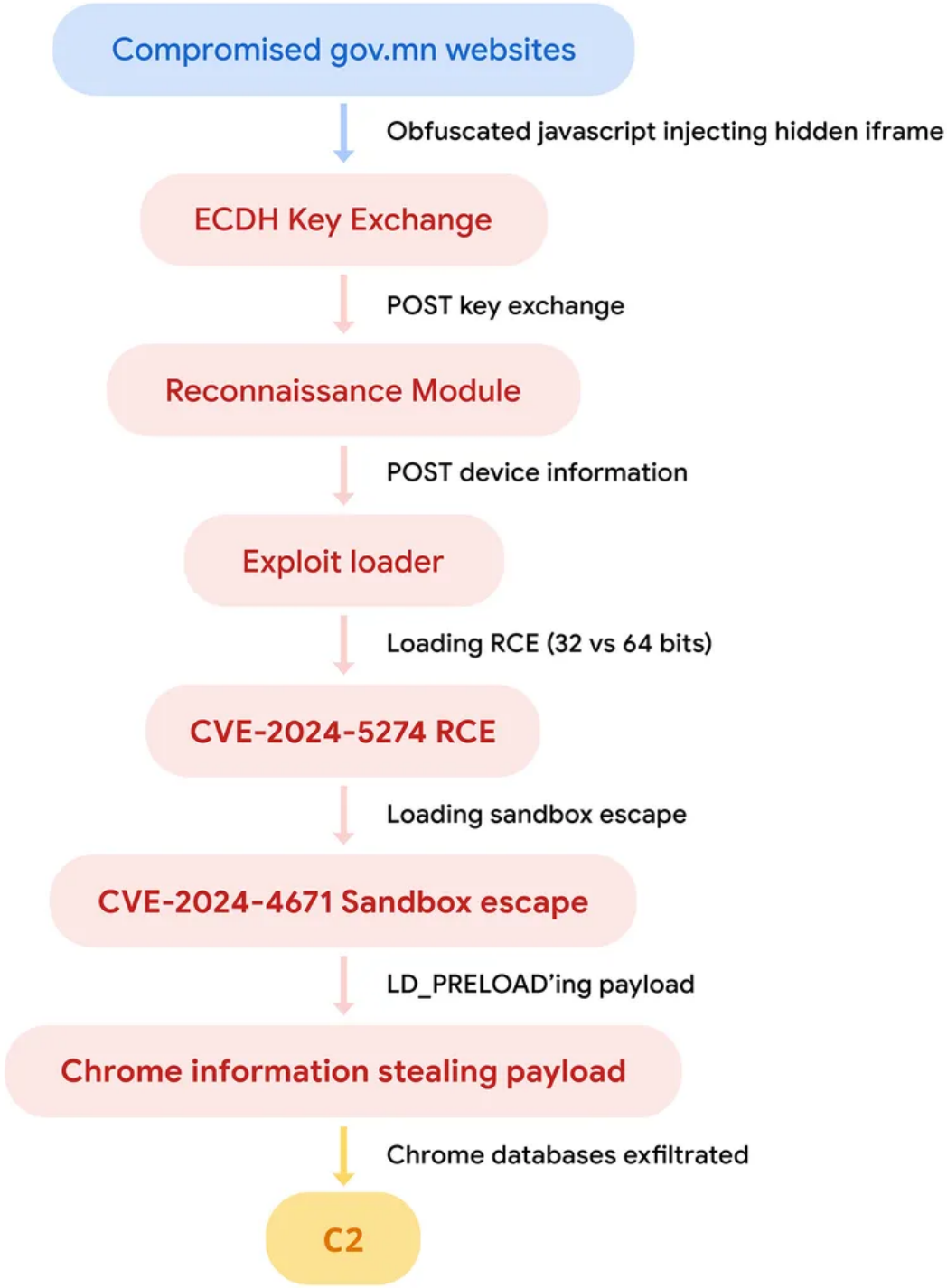
On more recent versions of iOS, the payload is calling WebCore::NetworkStorageSession::getAllCookies() to collect all cookies before exfiltrating them back to the C2.

Google Chrome campaign

At the end of July 2024, a new watering hole appeared on the mfa.gov[.]mn website where track-adv[.]com was re-used to deliver a Google Chrome exploit chain to Android users. From a high-level overview, the attack and end goal are essentially the same as the iOS one — using n-day vulnerabilities

in order to steal credential cookies — with some differences on the technical side. In this case, the attack required an additional sandbox escape vulnerability to break out of Chrome site isolation.

- Instead of a simple iframe directly added into the HTML, the attackers are now using a piece of obfuscated javascript to inject the malicious iframe pointing to https://track-adv[.]com/analytics.php?personalization_id=<random number>.
- Before sending any stages, crypto keys are generated and exchanged using proper ECDH key exchange. Previous campaigns received a static decryption key from the C2.
- In both campaigns the attack uses indexedDB to store status information on the client side. In the iOS exploit the database was named minus and in the Chrome exploit the database was named tracker.
- A unique identifier using the same format (e.g., 2msa5mmjhqxpdsyb5vlcnd2t) was generated and passed as tt= parameter during all stages.



Attack chain used during the July 2024 campaign targeting Google Chrome.

Reconnaissance payload

The reconnaissance payload is relatively simple and sends back browser information that is accessible from javascript to make sure the target is a real Chrome browser behind a real device. For example, this includes:

- Screen sizes
- Number of CPUs
- Brands property from client hints
- GPU information

- Information from window.navigator

Exploit

The exploit chained 2 vulnerabilities:

- CVE-2024-5274 to compromise the renderer. TAG and Chrome Security discovered and reported CVE-2024-5274 as an in-the-wild 0-day in May 2024 used by NSO Group, another CSV. Unlike the CVE-2023-41993 case above, here the attacker adapted NSO Group's exploit. Even though they share a very similar trigger, as seen in the screenshot below, the two exploits are conceptually different and the similarities are less obvious than the iOS exploit. For example, the NSO exploit was supporting Chrome versions ranging from 107 to 124, and the exploit from the watering hole was only targeting versions 121, 122 and 123 specifically.

```
o();
((n = class {
    static {
        this
    }
    constructor() {}
}) => {})()
};
r();
if (t == 11) {
    he()
}
```

```
Cc();
((XXX = class {
    static {
        this;
    }
    constructor() {}
}) => {})();;
};
Dc();
if (bc == wc - 2) {
    mc();
}
```

The triggers for CVE-2024-5274 used in the July 2024 watering hole attack (left image) and by NSO in May 2024 (right image).

- CVE-2024-4671 to escape the Chrome sandbox: The vulnerability was reported anonymously as an in-the-wild 0-day to Chrome in May 2024. The exploit sample from the watering hole, code named chopin, is similar to a previous Chrome sandbox escape we discovered Intellexa using in-the-wild exploiting CVE-2021-37973.

The attackers type confused Blink objects to escape the V8 heap sandbox, which was a known technique now fixed in Chrome m127.

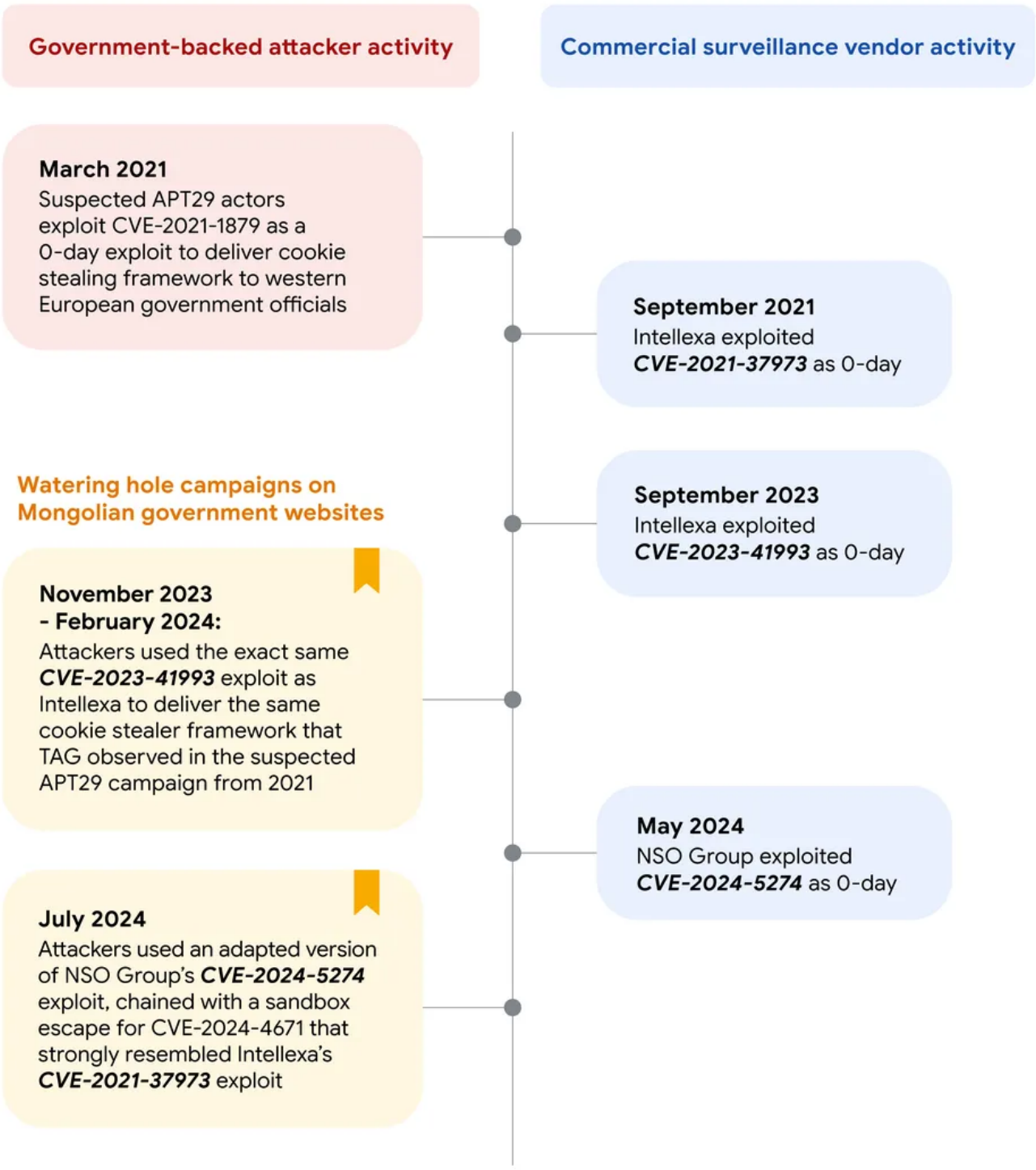
Cookie stealer payload

Once the Chrome sandbox is escaped, a new payload is dropped into /data/data/com.android.chrome/c.so and executed via LD_PRELOAD. Normally, we would expect to see another stage exploiting a vulnerability to elevate privileges and escape from the Chrome user. This campaign delivers a simple binary deleting all Chrome Crash reports and exfiltrating the following Chrome databases back to the track-adv[.]com server — similar to the basic final payload seen in the earlier iOS campaigns.

- Cookies: saved cookies for all websites
- Account Web Data: account related data like saved credit cards
- Login Data: passwords stored in Chrome
- History: user Chrome history
- Trust Tokens: all Trust Tokens

Exploit reuse timeline

As mentioned above, in each iteration of the watering hole campaigns, the attackers used exploits that were identical or strikingly similar to exploits from CSVs, Intellexa and NSO Group. We do not know how the attackers acquired these exploits. What is clear is that APT actors are using n-day exploits that were originally used as 0-days by CSVs. It should be noted that outside of common exploit usage, the recent watering hole campaigns otherwise differed in their approaches to delivery and second-stage objectives.



Conclusion

While we are uncertain how suspected APT29 actors acquired these exploits, our research underscores the extent to which exploits first developed by the commercial surveillance industry are proliferated to dangerous threat actors. Moreover, watering hole attacks remain a threat where sophisticated exploits can be utilized to target those that visit sites regularly, including on mobile devices. Watering holes can still be an effective avenue for n-day exploits by mass targeting a population that might still run unpatched browsers.

Although the trend in the mobile space is towards complex full exploit chains, the iOS campaign is a good reminder of the fact that a single vulnerability can inflict harm and be successful. Google Chrome on Android provides users strong default protections through features like Site Isolation that prevent the ability to steal other website data — including cookies — from a compromised renderer. In order to be successful, the attackers had to chain an additional Chrome sandbox escape to read other website data.

Following our disclosure policy, TAG shares its research to raise awareness and advance security across the ecosystem. We also add all identified websites and domains to Safe Browsing to safeguard users from further exploitation. We urge users and organizations to apply patches quickly and keep software fully up-to-date for their protection. TAG will remain focused on detecting, analyzing, and preventing 0-day exploitation as well as reporting vulnerabilities to vendors immediately upon discovery.

Special thanks to TAG's Josh Atkins and Mandiant's Luke Jenkins for their contributions to this analysis.

Indicators of Compromise (IoCs)

- iOS reconnaissance payload (VALIDVICTOR): 8bd9a73da704b4d7314164bff71ca76c15742dcc343304def49b1e4543478d1a
- iOS cookie stealer module (COOKIESNATCH): d19dcbb7ab91f908d70739968b14b26d7f6301069332609c78aafc0053b6a7e1
- Chrome reconnaissance payload: 21682218bde550b2f06ee2bb4f6a39cff29672ebe27acbb3cee5db79bf6d7297
- Chrome cookie stealer payload (ANDROSNATCH): df21c2615bc66c369690cf35aa5a681aed1692a5255d872427a2970e2894b2e3
- https://ceo-adviser[.]com/fb-connect.php?online=1
- https://track-adv[.]com/market-analytics.php?pc=1
- https://track-adv[.]com/analytics.php?personalization_id=<random number>