NO ORDINARY VULNERABILITY —

4-year campaign backdoored iPhones using possibly the most advanced exploit ever

"Triangulation" infected dozens of iPhones belonging to employees of Moscow-based Kaspersky.

DAN GOODIN - 12/27/2023, 2:03 PM

Tero Vesalainen

Enlarge

Researchers on Wednesday presented intriguing new findings surrounding an attack that over four years backdoored dozens if not thousands of iPhones, many of which belonged to employees of Moscow-based security firm Kaspersky. Chief among the discoveries: the unknown attackers were able to achieve an unprecedented level of access by exploiting a vulnerability in an undocumented hardware feature that few if anyone outside of Apple and chip suppliers such as ARM Holdings knew of.

“The exploit's sophistication and the feature's obscurity suggest the attackers had advanced technical capabilities,” Kaspersky researcher Boris Larin wrote in an email. “Our analysis hasn't revealed how they became aware of this feature, but we're exploring all possibilities, including accidental disclosure

in past firmware or source code releases. They may also have stumbled upon it through hardware reverse engineering."

Four zero-days exploited for years

Other questions remain unanswered, wrote Larin, even after about 12 months of intensive investigation. Besides how the attackers learned of the hardware feature, the researchers still don't know what, precisely, its purpose is. Also unknown is if the feature is a native part of the iPhone or enabled by a third-party hardware component such as ARM's CoreSight

FURTHER READING
"Clickless" iOS exploits infect Kaspersky iPhones with never-before-seen malware

The mass backdooring campaign, which according to Russian officials also infected the iPhones of thousands of people working inside diplomatic missions and embassies in Russia, according to Russian government officials, came to light in June. Over a span of at least four years, Kaspersky said, the infections were delivered in iMessage texts that installed malware through a complex exploit chain without requiring the receiver to take any action.

With that, the devices were infected with full-featured spyware that, among other things, transmitted microphone recordings, photos, geolocation, and other sensitive data to attacker-controlled servers. Although infections didn't survive a reboot, the unknown attackers kept their campaign alive simply by sending devices a new malicious iMessage text shortly after devices were restarted.

A fresh infusion of details disclosed Wednesday said that "Triangulation"—the name Kaspersky gave to both the malware and the campaign that installed it—exploited four critical zero-day vulnerabilities, meaning serious programming flaws that were known to the attackers before they were known to Apple. The company has since patched all four of the vulnerabilities, which are tracked as:

- CVE-2023-32434
- CVE-2023-32435
- CVE-2023-38606
- CVE-2023-41990

Besides affecting iPhones, these critical zero-days and the secret hardware function resided in Macs, iPods, iPads, Apple TVs, and Apple Watches. What's more, the exploits Kaspersky recovered were intentionally developed to work on those devices as well. Apple has patched those platforms as well. Apple declined to comment for this article.

Detecting infections is extremely challenging, even for people with advanced forensic expertise. For those who want to try, a list of Internet addresses, files, and other indicators of compromise is here.

Mystery iPhone function proves pivotal to Triangulation's success

The most intriguing new detail is the targeting of the heretofore-unknown hardware feature, which proved to be pivotal to the Operation Triangulation campaign. A zero-day in the feature allowed the attackers to bypass advanced hardware-based memory protections designed to safeguard device system integrity even after an attacker gained the ability to tamper with memory of the underlying

kernel. On most other platforms, once attackers successfully exploit a kernel vulnerability they have full control of the compromised system.

On Apple devices equipped with these protections, such attackers are still unable to perform key post-exploitation techniques such as injecting malicious code into other processes, or modifying kernel code or sensitive kernel data. This powerful protection was bypassed by exploiting a vulnerability in the secret function. The protection, which has rarely been defeated in exploits found to date, is also present in Apple's M1 and M2 CPUs.

Kaspersky researchers learned of the secret hardware function only after months of extensive reverse engineering of devices that had been infected with Triangulation. In the course, the researchers' attention was drawn to what are known as hardware registers, which provide memory addresses for CPUs to interact with peripheral components such as USBs, memory controllers, and GPUs. MMIOs, short for Memory-mapped Input/Outputs, allow the CPU to write to the specific hardware register of a specific peripheral device.

The researchers found that several of MMIO addresses the attackers used to bypass the memory protections weren't identified in any so-called device tree, a machine-readable description of a particular set of hardware that can be helpful to reverse engineers. Even after the researchers further scoured source codes, kernel images, and firmware, they were still unable to find any mention of the MMIO addresses.

"This is no ordinary vulnerability," Larin said in a press release that coincided with a presentation he made at the 37th Chaos Communication Congress in Hamburg, Germany. "Due to the closed nature of the iOS ecosystem, the discovery process was both challenging and time-consuming, requiring a comprehensive understanding of both hardware and software architectures. What this discovery teaches us once again is that even advanced hardware-based protections can be rendered ineffective in the face of a sophisticated attacker, particularly when there are hardware features allowing to bypass these protections."

In a research paper also published Wednesday, Larin added:

> If we try to describe this feature and how attackers use it, it all comes down to this: attackers are able to write the desired data to the desired physical address with [the] bypass of [a] hardware-based memory protection by writing the data, destination address and hash of data to unknown, not used by the firmware, hardware registers of the chip.
>
> Our guess is that this unknown hardware feature was most likely intended to be used for debugging or testing purposes by Apple engineers or the factory, or was included by mistake. Since this feature is not used by the firmware, we have no idea how attackers would know how to use it

On the same day last June that Kaspersky first disclosed Operation Triangulation had infected the iPhones of its employees, officials with the Russian National Coordination Center for Computer Incidents said the attacks were part of a broader campaign by the US National Security Agency that infected several thousand iPhones belonging to people inside diplomatic missions and embassies in Russia, specifically from those representing NATO countries, post-Soviet nations, Israel, and China. A

separate alert from the FSB, Russia's Federal Security Service, alleged Apple cooperated with the NSA in the campaign. An Apple representative has denied the claim. Kaspersky researchers, meanwhile, have said they have no evidence corroborating the claim of involvement by either the NSA or Apple.

"Currently, we cannot conclusively attribute this cyberattack to any known threat actor," Larin wrote in the email. "The unique characteristics observed in Operation Triangulation don't align with

Possibly the most sophisticated exploit ever

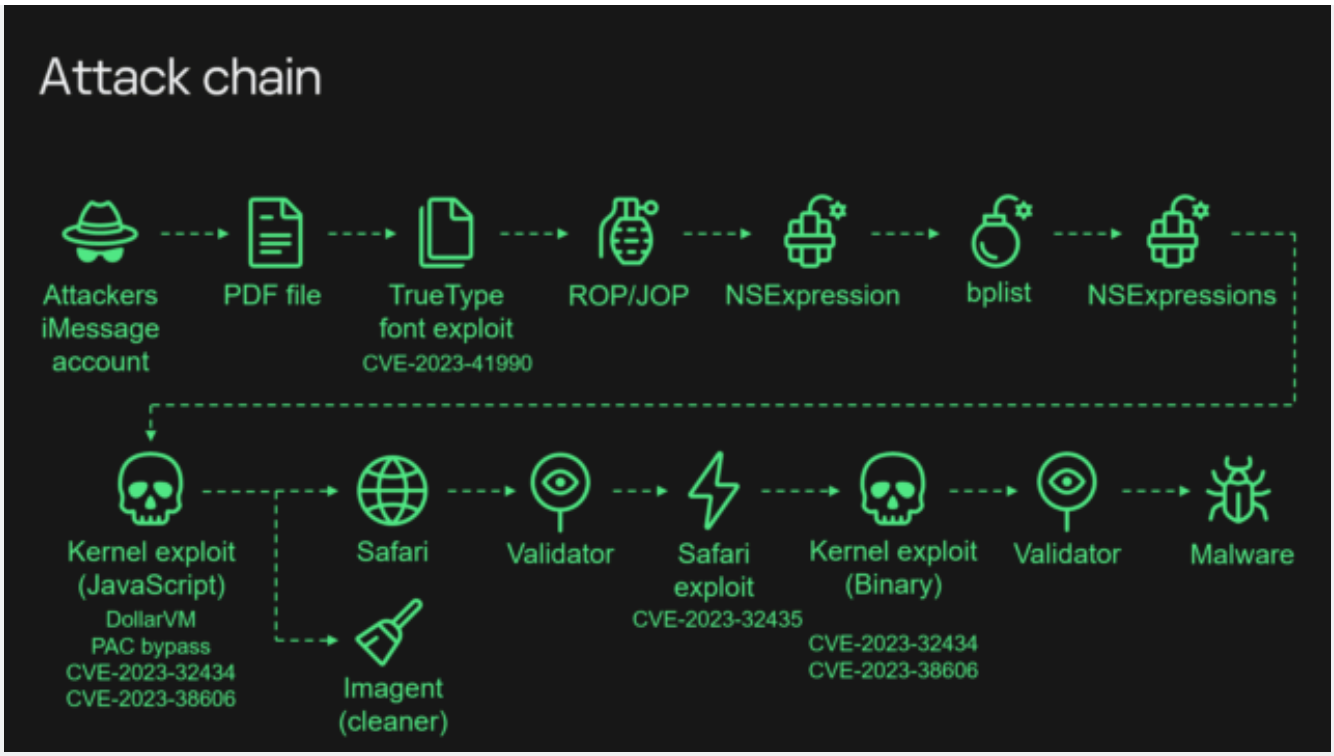
The findings presented Wednesday also detail the intricacies of the exploit chain that underpinned the Triangulation infections. As noted earlier, the chain exploited four zero-days to ensure that the Triangulation malware ran with root privileges and gained complete control over the device and user data stored on it.

It began by exploiting CVE-2023-41990, a vulnerability in Apple's implementation of the TrueType font. This initial chain link, which used techniques including return oriented programming and jump oriented programming to bypass modern exploit defenses, allowed the attackers to remotely execute code, albeit with minimum system privileges.

With that initial hurdle cleared, the next link in the exploit chain targeted the iOS kernel, the core of the OS reserved for the most sensitive device functions and data. This kernel manipulation came courtesy of the CVE-2023-32434 and CVE-2023-38606. CVE-2023-32434 is a memory-corruption vulnerability in XNU, a mechanism designed to withstand attempts to corrupt the memory inside the iOS kernel. This link went on to exploit CVE-2023-38606, the vulnerability residing in the secret MMIO registers. It allowed the bypassing of the Page Protection Layer, the defense discussed earlier that prevents malicious code injection and kernel modification even after a kernel has been compromised.

The chain then exploited a Safari vulnerability tracked as CVE-2023-32435 to execute shellcode. The resulting shellcode, in turn, went on to once again exploit CVE-2023-32434 and CVE-2023-38606 to finally achieve the root access required to install the last spyware payload.

Kaspersky provided the following diagram of the exploit chain.



Enlarge / **Diagram of the Triangulation exploit chain.**

Kaspersky's summary of the exploit chain is:

> - Attackers send a malicious iMessage attachment, which is processed by the application without showing any signs to the user
> - This attachment exploits vulnerability CVE-2023-41990 in the undocumented, Apple-only TrueType font instruction ADJUST for a remote code execution. This instruction existed since the early 90's and the patch removed it.
> - It uses return/jump oriented programming, multiple stages written in NSExpression/NSPredicate query language, patching JavaScriptCore library environment to execute a privilege escalation exploit written in JavaScript.
> - This JavaScript exploit is obfuscated to make it completely unreadable and to minimize its size. Still it has around 11000 lines of code which are mainly dedicated to JavaScriptCore and kernel memory parsing and manipulation.
> - It's exploited JavaScriptCore's debugging feature DollarVM ($vm) to get the ability to manipulate JavaScriptCore's memory from the script and execute native API functions.
> - It was designed to support old and new iPhones and included a Pointer Authentication Code (PAC) bypass for exploitation of newer models.
> - It used an integer overflow vulnerability CVE-2023-32434 in the XNU's memory mapping syscalls (mach_make_memory_entry and vm_map) to get read/write access to [the] whole physical memory of the device from the user level.
> - It uses hardware memory-mapped I/O (MMIO) registers to bypass Page Protection Layer (PPL). This was mitigated as CVE-2023-38606.
> - After exploiting all the vulnerabilities, the JavaScript exploit can do whatever it wants to the device and run spyware, but attackers chose to: a) launch the imagent process and

inject a payload that cleans the exploitation artifacts from the device; b) run the Safari process in invisible mode and forward it to the web page with the next stage.

- Web page has the script that verifies the victim and, if the checks pass, it receives the next stage—the Safari exploit.
- Safari exploit uses vulnerability CVE-2023-32435 to execute a shellcode.
- Shellcode executes another kernel exploit in the form of mach object file. It uses the same vulnerabilities CVE-2023-32434 and CVE-2023-38606, it's also massive in size and functionality, but it is completely different from the kernel exploit written in JavaScript. Only some parts related to exploitation of the above-mentioned vulnerabilities are the same. Still most of its code is also dedicated to the parsing and manipulation of the kernel memory. It has various post-exploitation utilities, which are mostly unused.
- Exploit gets root privileges and proceeds to execute other stages responsible for loading of spyware. We already covered these stages in our previous posts.

Wednesday's presentation, titled What You Get When You Attack iPhones of Researchers, is a further reminder that even in the face of innovative defenses like the one protecting the iPhone kernel, ever more sophisticated attacks continue to find ways to defeat them.

"We discover and analyze new exploits and attacks using them on a daily basis," Larin wrote. "We've discovered and reported more than thirty in the wild zero-days in Adobe/Apple/Google/Microsoft products, but this is definitely the most sophisticated attack chain we've ever seen."

READER COMMENTS 157

DAN GOODIN

Dan Goodin is Senior Security Editor at Ars Technica, where he oversees coverage of malware, computer espionage, botnets, hardware hacking, encryption, and passwords. In his spare time, he enjoys gardening, cooking, and following the independent music scene.

Promoted Comments

Jt21

> Both options are likely and equally valid. Seeing how it was predominantly aimed at Russian based targets, and the vulnerabilities/features would have been at least known to Apple/ARM it's not unreasonable to assume that the NSA was behind this either at the root introducing it, or later on just exploiting it.
>
> It's not even unexpected given that this is how the spying game works

> and the NSA was already known to impose such vulnerabilities on everyone else in the world for decades.

It didn't target Russians. It targeted foreign embassies in Russia. My bet would be FSB or Chinese intelligence. Who's more likely to want to spy on NATO embassy personnel, the NSA or FSB?
December 27, 2023 at 6:34 pm

SeanJW

> What I don't understand, and maybe someone can explain, is why iMessages or SMS messages are allowed to do anything without user interaction. It has the aroma of Outlook opening email attachments indiscriminately or Microsoft Office automatically running macros and VBScripts embedded in documents. This stuff should have been *ganz streng verboten* more than two decades ago. Why does it still exist?

Every program does things without interaction required every day. You opened a program? Boom, it's doing what's necessary to show you the opening screen. In this case, that's what iMessage is doing - it's rendering the messages, which includes the fonts, which triggers a CVE that's now fixed.

Edit: that can include previewing them in a notification, or otherwise just downloading them and storing them. All things you expect to happen.
December 27, 2023 at 10:45 pm

Unsolved Mysteries Of Quantum Leap With Donald P. Bellisario

Today "Quantum Leap" series creator Donald P. Bellisario joins Ars Technica to answer once and for all the lingering questions we have about his enduringly popular show. Was Dr. Sam Beckett really leaping between all those time periods and people or did he simply imagine it all? What do people in the waiting room do while Sam is in their bodies? What happens to Sam's loyal ally Al? 30 years following the series finale, answers to these mysteries and more await.


Unsolved Mysteries Of Quantum Leap With Donald P. Bellisario


Unsolved Mysteries Of Warhammer 40K With Author Dan Abnett


SITREP: F-16 replacement search a signal of F-35 fail?


Sitrep: Boeing 707

More videos

← PREVIOUS STORY

NEXT STORY →

Related Stories

Today on Ars