SentinelLABS

SECURITY RESEARCH

Privilege Escalation | macOS Malware & The Path to Root Part 1

PHIL STOKES | NOVEMBER 4, 2019

In this two-part series, we take a look at privilege escalation on macOS. In Part 1, we look at some of the vulnerabilities that have been discovered by security researchers in recent versions of Apple's Desktop OS, focusing on those that can be turned into reliable exploits. We draw conclusions for enterprise and end users alike based on this review. In Part 2, we switch from researcher to threat actor perspective to see how and why the methodology of macOS threat actors takes quite a different path from that of the research community.

What is Privilege Escalation?

Let's start by defining our terms. Whenever code executes, it does so within the context of a user who invokes it. Technically, users need not always actually be people, but for our purposes here we'll stick to the simple case of a user deciding to launch some application or script. When that happens, the code runs access to the same resources that the user has. That is, access to system files, other users' files and any other protected resources is usually out of scope.

Unless, of course, that user happens to be the root user, or has launched the process by first requesting to be run as root. Users familiar with the command line will recognize as an example of the second scenario those times when they execute a command prepending with `sudo`. In the GUI-based user interface, the same thing occurs when an installer or other tool prompts for permissions to make some changes that the current user does not have the authority to make, at least not without authentication (i.e., providing the required credentials). This can be anything from installing a new helper tool, moving or deleting a folder outside of the user's home folder or executing a script that requires elevated privileges.

This kind of privilege elevation is all well and good, but privilege escalation occurs when a user or process acquires these same elevated privileges when they are not supposed to. Privilege escalation can occur through software or OS vulnerabilities – largely the subject of this post – but also through social engineering, which we'll mention more about in Part 2.

How Common is Privilege Escalation on macOS?

If you pay attention to Apple's product security announcements (you can sign up here for email alerts), you may be surprised at just how many code execution bugs are squashed on each update. That, of course, is great! Apple and a small body of dedicated researchers around the world are constantly exposing and fixing serious security bugs.


Image apple security updates

That said, not all of the bugs that Apple fix are privilege escalation bugs. Many are a more general class of bugs known as 'arbitrary code execution', meaning only that the flaw could allow an attacker to execute any code they choose in a certain context. Arbitrary code execution can be a precursor of privilege escalation, however. An exploit chain often begins with the ability to execute arbitrary code, but whether the attacker can use that to escalate code that will cross privileges is a separate matter.

Even when a vulnerability does allow privilege escalation, not all such vulnerabilities turn into working exploits. The flaw may be mitigated by other circumstances that are so rare as to make the bug merely of technical interest. Take, for example, this macOS 10.7.0 Kernel – Local Privilege Escalation, which the author describes as "pretty much unusable for in the wild exploitation" but nevertheless still of interest to security researchers:


Image of exploit db 1


Image of exploit db 2

Despite this, there's still been quite a few serious privilege escalation bugs in macOS for which there are working exploits. Let's take a look at some of them.

Privilege Escalation on macOS El Capitan 10.11, macOS Sierra 10.12

To keep it relevant, we'll start with exploits that still affect at least El Capitan, the version where Apple first introduced System Integrity Protection and, arguably, began to take security seriously. As of Oct 2019, El Capitan is estimated to hold around 7% of macOS market share, with Sierra a little above that at 8.99%.


Image of macos stats

Psystem is an exploit that can be used on older OSs, of course. I've personally tested it on systems as old as OSX 10.9 Mavericks, and the author of the exploit believes the code probably dates back to OSX 10.5 Leopard. And because all versions of OSX El Capitan, Sierra, and El Capitan versions 10.11.5 and above as well as Sierra 10.12.3 and 10.12.5. Psystem exploits either CVE-2016-1825 or CVE-2016-7617 depending on the target system.

The source code for psystem is publicly available, and thus accessible to attackers. As a local privilege escalation, it requires a user to download and run some 3rd party software, a fairly regular occurrence, which unknown to the victim contains the malicious code. A seemingly benign piece of software (perhaps a fake Adobe Flash installer, Media downloader or similar) could contain the psystem binary and allow that process to elevate to root without the user needing to type in an admin or any other kind of password. Of course, once root is achieved, the software can now do as it wishes without the user's interaction. The whole process would be invisible to the user from the point that they launch the supposedly benign software, which might – to avoid raising suspicion – perform otherwise exactly as expected.

macOS 10.13 High Sierra Privilege Escalations

The vulnerability that makes psystem possible was patched in 10.12.2 onwards, but there's a whole class of other vulnerabilities that affect unpatched versions of El Capitan, Sierra and High Sierra and which was detailed in this blog post.

Although far from simple to exploit, that didn't stop the determined researchers from crafting a working exploit from a flaw in macOS's Windows Server and achieving arbitrary code execution with system privileges.

Chained with other exploits such as a sandbox escape, CVE-2018-4193 could be used to achieve remote code execution, requiring only that a user click a malicious link. It is estimated that around 18% of macOS installs are still running some version of High Sierra 10.13.

macOS 10.14 Privilege Escalations

Mojave, which accounts for around 42% of macOS installs, is not immune from privilege escalation, either. Product security and vulnerability researcher @CodeColorist has discovered two vulnerabilities, CVE-2019-8565 and CVE-2019-8513 that lead to privilege escalation on macOS Mojave 10.14.3 and earlier. We have already been incorporated into Metasploit and are available to red teamers.

The first exploits a race condition in a little known but native macOS application called Feedback Assistant. This app resides in an obscure "Applications" folder within the `System/Library/CoreServices` path.

Feedback Assistant is an application used primarily by developers and macOS bug testers to submit problem reports directly to Apple. As CodeColorist discovered following work by Project Zero's Ian Beer, the Feedback Assistant leverages a privileged XPC connection with the service name "com.apple.appleseed.fbahelperd". However, since communication with the privileged XPC service is verified only by the caller's process identifier, this makes it subject to a race condition whereby the malicious application first spawns the entitled process and then reuses its PID.


Image of target feedback assistant

In this example, we execute the exploit on a macOS High Sierra instance, but the exploit is quite reliable on macOS versions 10.14.3 and below.

Attempting to use the same exploit on 10.14.4, however, fails after Apple patched the bug.

CVE-2019-8513, which also works up to and including Mojave 10.14.3, shows just how sketchy some of the built-in command line tools can be on macOS. The Time Machine util `tmutilgetsize` is just one example of a whole class of tools that implement XPC logic bugs.

Image of tmutilgetsize

The utility is largely just an Objective-C wrapper around a bunch of other command line utilities. Prior to 10.14.4, one utility that `tmutilgetsize` called to do a privileged task was `du`. Although `du` is a full-blown programming utility in its own right, most people who are familiar with it will know it as a powerful and useful utility for processing text files. The code in `tmutilgetsize` uses `du` to split up some text output from `diskutil list` and then make a system call `diskutil list`.

This is an incredibly unsafe way to execute privileged system processes, the system call can be manipulated by funking with the output of `diskutil list`. Cleverly, CodeColorist's exploit creates a disk image whose name contains the malicious payload. When `tmutilgetsize` is called and runs over the list of disk names, it executes the payload with privileges.

Are There Any Privilege Escalations for macOS Catalina?

And what of post 10.14.4 and 10.15? A quick look at Apple's most recent security update for macOS Catalina suggests we'll be seeing quite a few write-ups in the near future of further privilege escalation vulnerabilities and exploits. The count from the most recent security update for macOS Catalina 10.15.1 is at least 9 bugs that target the 10.15 release or 10.14.6 with impacts that allow arbitrary code execution with system or elevated privileges.

File System Events
Available for: macOS High Sierra 10.13.6, macOS Catalina 10.15
Impact: An application may be able to execute arbitrary code with system privileges
Description: A memory corruption issue was addressed with improved memory handling.
CVE-2019-8788: ABC Research s.r.o. working with Trend Micro's Zero Day Initiative

These bugs affect a wide variety of APIs and services, including the system Kernel Extensions (CVE-2019-8805), PlugInKit (CVE-2019-8715), the Kernel (CVE-2019-8786), Intel Graphics Driver (CVE-2019-8807), GraphicsDriver (CVE-2019-8784), File System Events (CVE-2019-8788), File Quarantine (CVE-2019-8505), AppleGraphicsControl (CVE-2019-8716) and even messages (CVE-2019-8802).

Graphics Driver
Available for: macOS Catalina 10.15
Impact: An application may be able to execute arbitrary code with system privileges
Description: A memory corruption issue was addressed with improved memory handling.

Intel Graphics Driver
Available for: macOS Catalina 10.15
Impact: An application may be able to execute arbitrary code with system privileges
Description: A memory corruption issue was addressed with improved memory handling.
CVE-2019-8807: Ye Wang of Didi Research America

What Can We Learn From macOS Privilege Escalations?

I regularly come across people who are still running older versions of macOS, both privately and at work. The reasons vary from familiarity – "my trusty old 10.xx install" – and mistrust – "What? Another bodged Apple update?" – to hardware or software incompatibility: users with older Macs that won't support an update or who are still relying on old software with dependencies like Rosetta, 32-bit apps, older Java JDK or JREs. The situation will likely escalate when notarization is in full force in 10.15 and later, as widely expected, when the removal of support for software using kernel extensions (kexts) in 10.15 or 10.17 kicks in over the next year or two.

One of the common responses I hear when talking to people running older versions of macOS is that security is not an issue, by which they mean that they are confident that the built-in protections ensure the older system is safe. Gatekeeper is on, System Integrity Protection is on, and the system settings are set to automatically receive background updates to XProtect and MRT.

More circumspect users may have installed some enduser AV suite, and feel that they are more than adequately covered.

Unfailingly, those users express surprise and concern when I point out working exploits like those above are commonly available and in use, whether by either the OS or many 3rd party security solutions, unless they happen to be monitoring execution at a deep level, which most are not.

Conclusion

What all this should teach macOS users is that, like any complex piece of software – and they don't come more complex than an operating system – the OS will always have vulnerabilities that dedicated researchers and enthusiasts will eventually uncover. Relying on built-in security or even 3rd party security solutions that don't have visibility into processes as they execute is no defense. It's why updates are essential, as is a more robust security solution.

In Part Two, we'll shift our focus from researchers to attackers. How does all this translate into what we see in the wild? Are threat actors taking the same path to root and privilege escalation as researchers, or have they found alternative ways to reach the same end?

MALWARE | PRIVILEGE ESCALATION

PHIL STOKES

Phil Stokes is a Threat Researcher at SentinelOne, specializing in macOS threat intelligence, platform vulnerabilities and malware analysis. He began his journey into macOS security as a software developer, creating and later troubleshooting and security tools just at the time when macOS adware and commodity malware first began appearing on the platform. Phil has been closely following the development of macOS threats as well as researching 'flea software and OS vulnerabilities since 2014.