Legacy Macs, Modern Solutions: A Hacker's Approach to Mac Sustainability.

BSides Calgary 2023

Mykola Grymalyuk

$ 'whoami'

> "Mykola Grymalyuk"

- Project lead of OpenCore Legacy Patcher, messing with (U)EFI firmware, XNU (macOS kernel) and macOS userspace.
- Research and document macOS internals:
  - Rapid Security Response Implementation.
  - Internals of Virtualization Stack on Apple Silicon.
- Software penetration testing on enterprise macOS applications.
  - Ex: CVE-2023-44077, CVE-2023-46368
- Work part time as a security and development technician, building out Mobile Device Management Infrastructure for Apple devices.
- Information Systems Security Student here in Calgary.

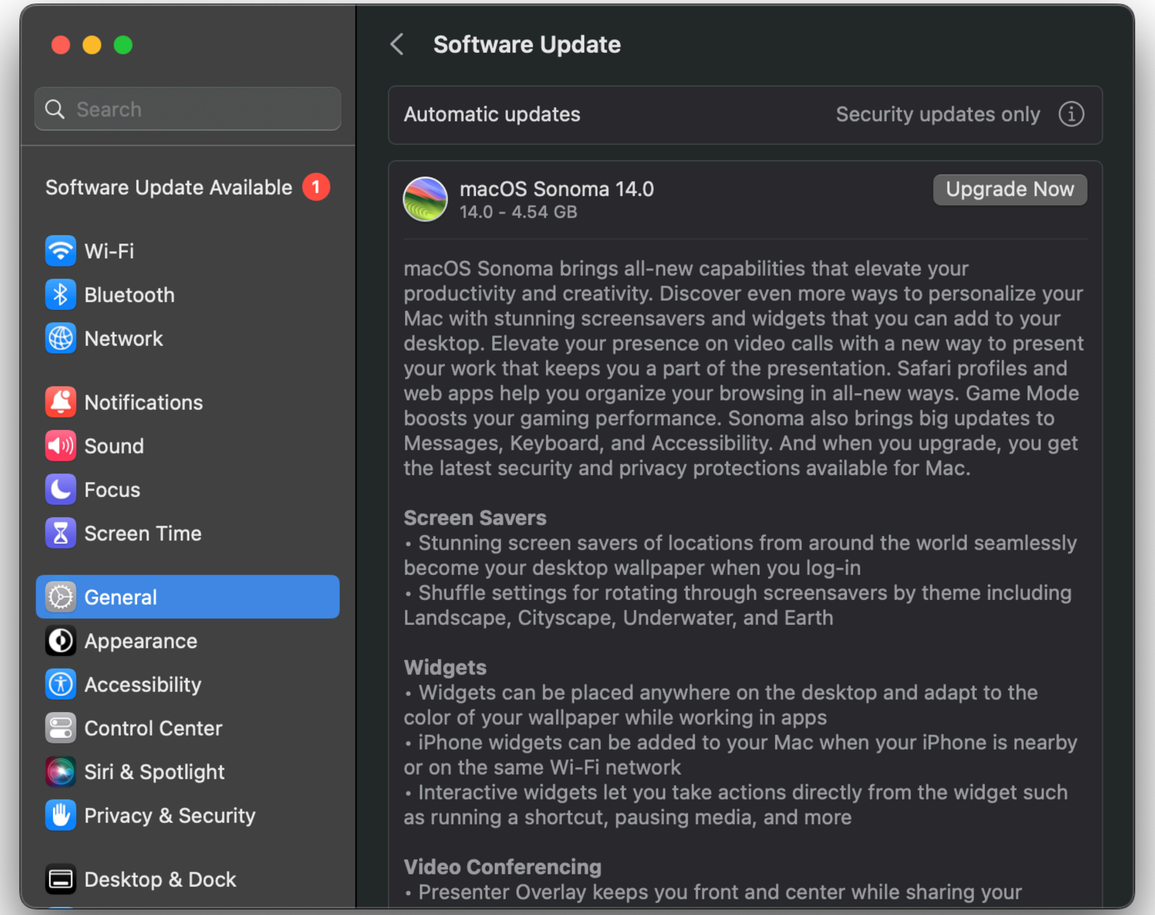
Agenda

- Understand Apple's Software Update Life Cycle.
- Look into the average lifetime of a Mac.
- Learn about the secret “tier 2” citizenship in macOS.
- See how projects like OpenCore Legacy Patcher bridge the gap for unsupported Macs.

Software Updates

What counts as one?

- We care about OS updates, namely major and minor versions, as well as security updates.
- Every year in June, a new major OS version is announced and released officially in the fall.
- Major OS updates don’t just include new fun features, can radically change the security architecture of the platform.



OS Life Cycle

What we *should* expect

- Year 1 - June: OS announced.
- Year 1 - Fall: OS released.
- Year 1 - Fall - Summer: OS gets minor and security updates.
- Year 2 - Summer: OS enters maintenance mode.
  - (Cycle repeats every year, so new OS just announced)
- Year 2 - 3: OS gets occasional security updates.
- Year 4: OS is no longer updated.

Issues with n-1 and n-2

- Idea of supporting 3 versions of macOS: n, n-1 and n-2
  - n: macOS 14, Sonoma.
  - n-1: macOS 13, Ventura.
  - n-2: macOS 12, Monterey.
- Apple always prioritizes n (latest) while n-1 and n-2 may get delayed updates.
- Apple has been known to omit fixes and never disclose if older OSes were affected (n-1+)
  - OBTS v4.0: "n-1 and n-2: Should we really trust in you?" - Josh Long
- Apple's guidance is to always be on the latest OS, cannot promise support for older OSes.

n: Immediately patched

n-1: Almost immediately patched

n-2: Usually patched 🤞

n-3: No love 💔

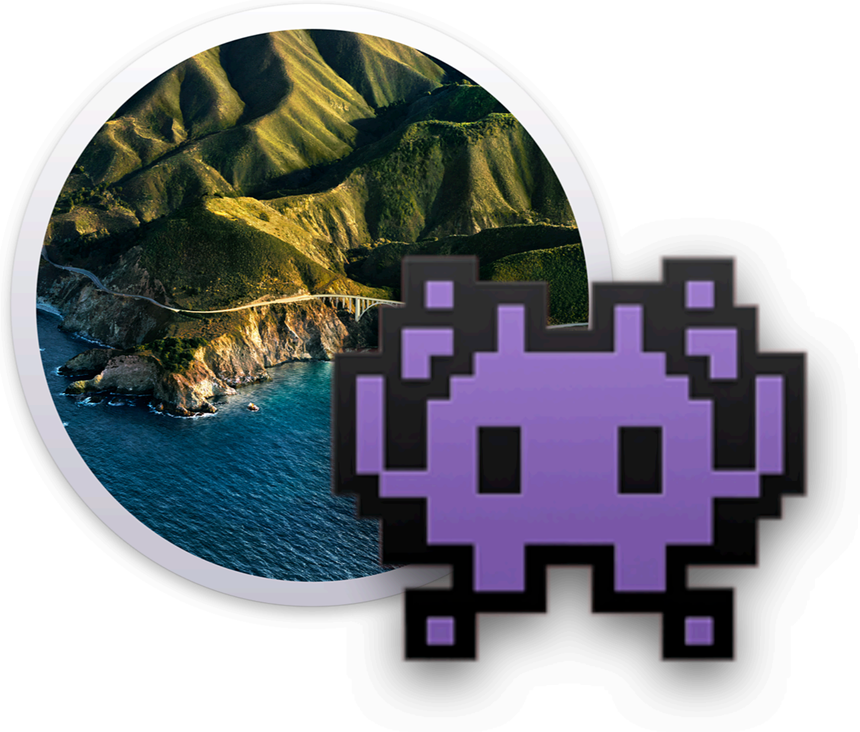
macOS Big Sur

First Apple Silicon OS to be dropped

- June 22nd, 2020: OS Unveiled at WWDC.
- November 12th, 2020: OS releases to the public.
- July 15th, 2021: Last proper OS update (11.5).
- September 11th, 2023: Last security update (11.7.10).

Unpatched exploits in Big Sur?
Fixes only in 12.7.1, 13.6.1 and 14.1

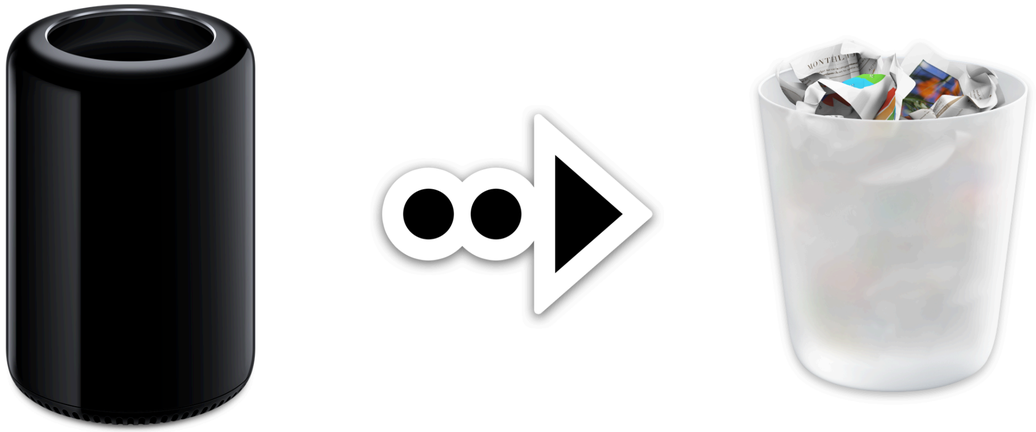
- CVE’s in all 3 macOS updates:
  - CVE-2023-40446: Arbitrary Code Execution.
  - CVE-2023-42856: Arbitrary Code Execution.
  - CVE-2023-42849: Kernel Memory Mitigation Bypass.
  - CVE-2023-40423: Execute arbitrary code with kernel privileges.
  - Etc...
- 8 different Mac Models left on Big Sur, cannot run anything newer.
  - ie. Always vulnerable to these CVEs.



Lifetime of a Mac

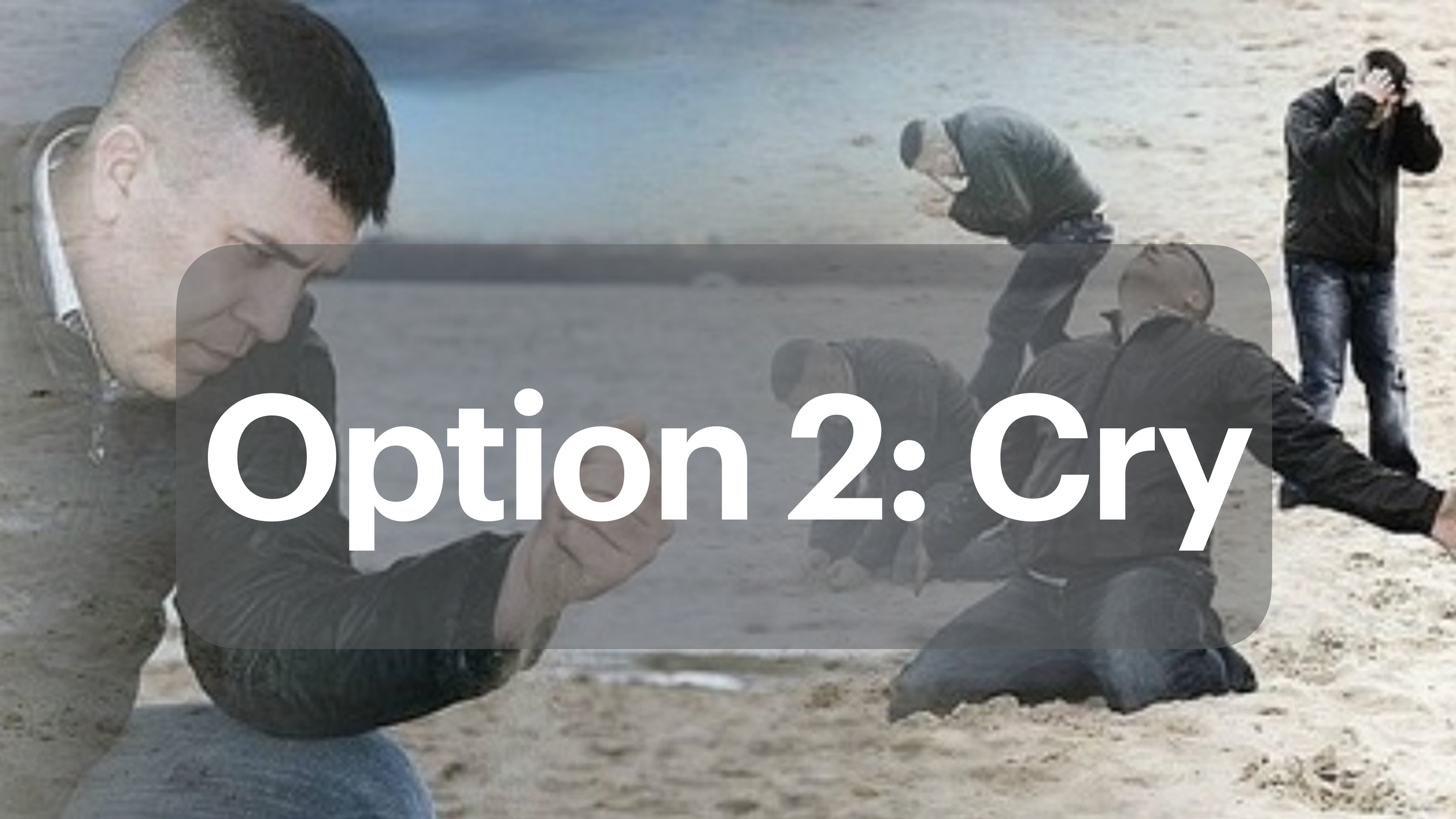
From Trash Can to Trash Can

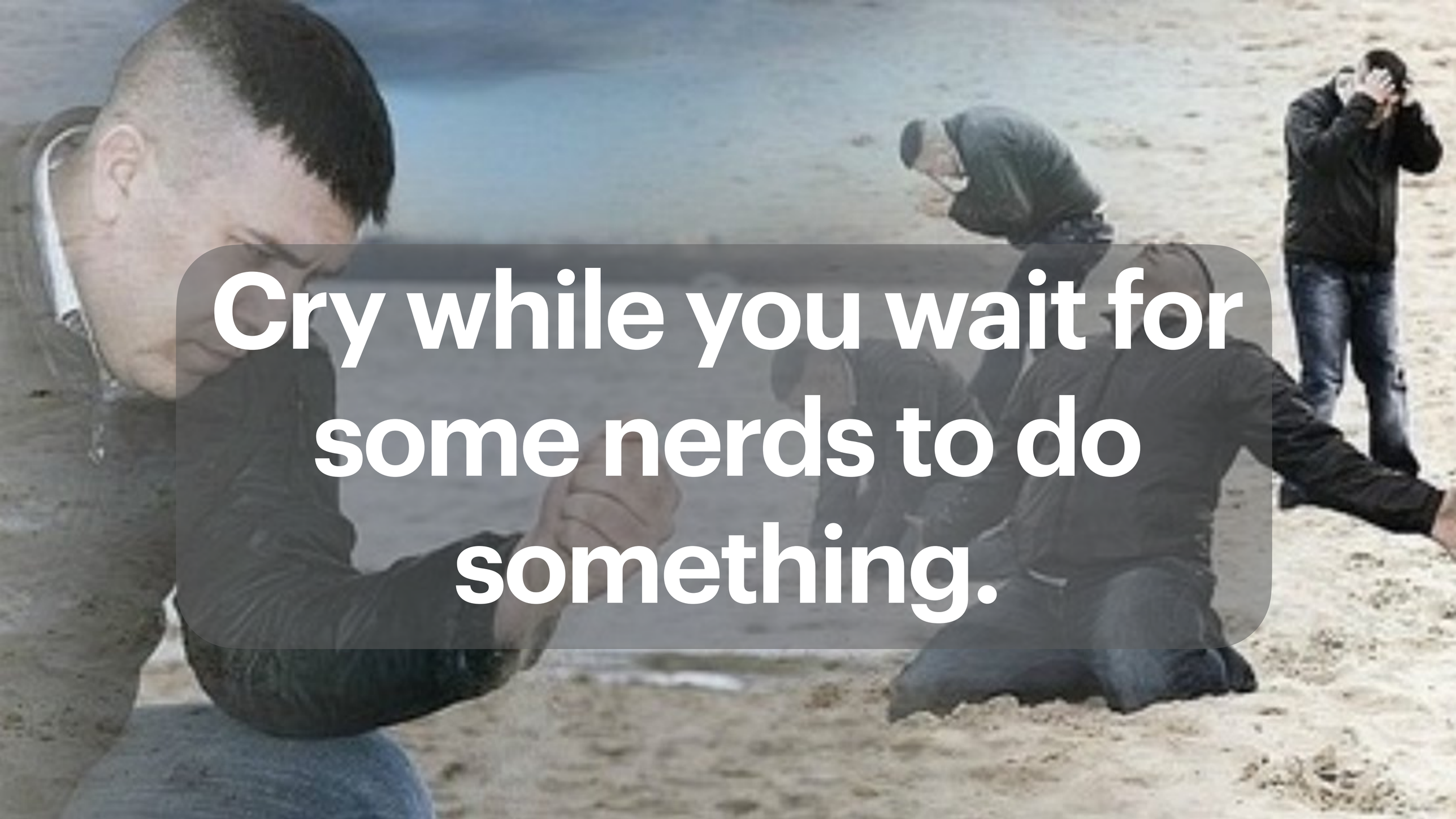
- Slowly going down since Apple Silicon released in 2020.
- Mac from 2012:
  - Lion (2012) through Catalina (2019).
  - 7 years of major OS updates.
- Mac from 2017:
  - Sierra (2017) through Ventura (2022).
  - 5 years of major OS updates.
- Each OS receives 2 additional years of security updates.



So what can you do once your Mac no longer get OS updates?

Option 1: Give up and install Windows/Linux


Option 2: Cry


Cry while you wait for some nerds to do something.

What limits machines to old OSes?

Can’t brute force it?

- Boot loader restrictions through boot.efi’s Board ID checks.
- Installer restrictions through hardware and firmware checks.
- Server restrictions through Software Update (Pallas) only returning older OSes.

How do Virtual Machines work?

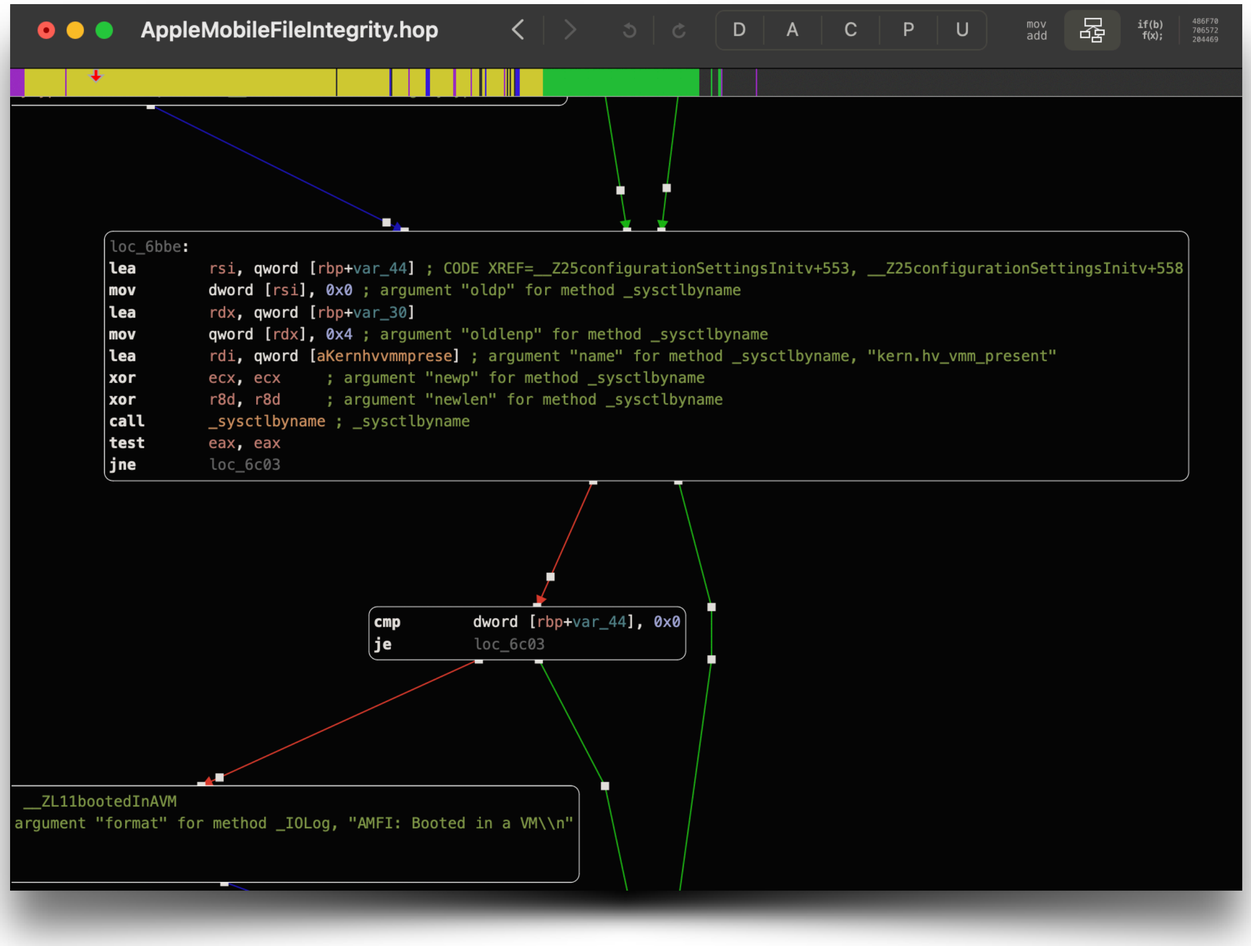
Tier 2 Citizenship

The magic that makes VMs work

- Provides exemptions for many models checks in macOS.
- Still requires some work on the Virtual Machine Manager’s end.
  - Firmware support for boot.efi.
  - DSMOS.kext (Don’t Steal Mac OS) - SMC check.
- macOS uses the VMM CPUID bit (`0x80000000`) to determine if host is a VM on x86_64 (Intel).
  - Apple Silicon is boring, just a device tree property.
- Tier 2 means little to no official support from Apple, you can install but the rest is up to you to get working.

How does macOS check VMM?

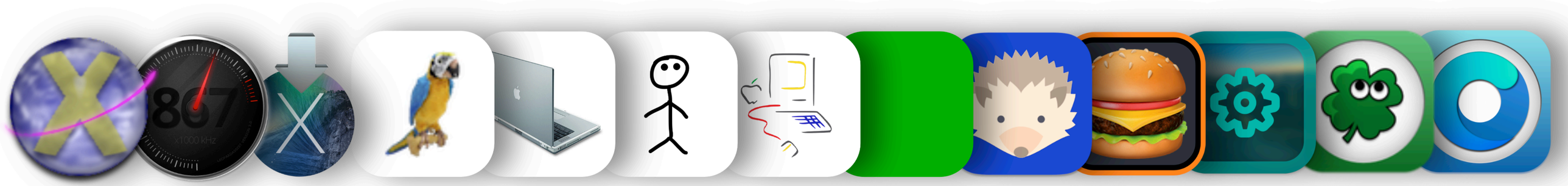
- In UEFI and kernel, checking against `'cpuid'` instruction directly.
- In kernel extensions and userspace, using the `'kern.hv_vmm_present'` value from kernel's `'sysctl'`.
  - Introduced with macOS 11.3.
  - Set by the kernel very early on.
  - Previous versions perform string comparison on CPU feature array.
    - Incompatible with Apple Silicon CPUs due to not reporting CPU features the same way, thus a new "streamlined" property was needed.



AMFI Kernel Extension performing a VM check with `'sysctlbyname'`

So how do we make our old Mac a tier 2 citizen?

Enter the world of macOS patchers



XPostFacto4, LeopardAssist, MacPostFactor, ParrotGeek1's NexPostFacto, dosdude1's macOS Patchers (Sierra - Catalina), BarryKN's micropatcher, Jacklukem's BigSurFixes and MontereyFixes, ASentientPatcher, Automator for micropatcher, Big Mac Patcher, Patched Sur, Clover Modern Patcher (DortaniaInternal version of OpenCore Legacy Patcher), OpenCore Legacy Patcher

What is a macOS patcher?

- Basic idea is to get macOS running on unsupported Mac hardware.
- Internal approaches vary, however common goal is to bypass Apple's model checks and restore hardware support.
- Existed for as long as Apple has dropped hardware support.
  - XPostFacto4 supported MacOS OS X 10.4, released in 2005 for PowerPC Macs!

Looking into the techniques of OpenCore Legacy Patcher

OpenCore Legacy Patcher

Experience macOS just like before

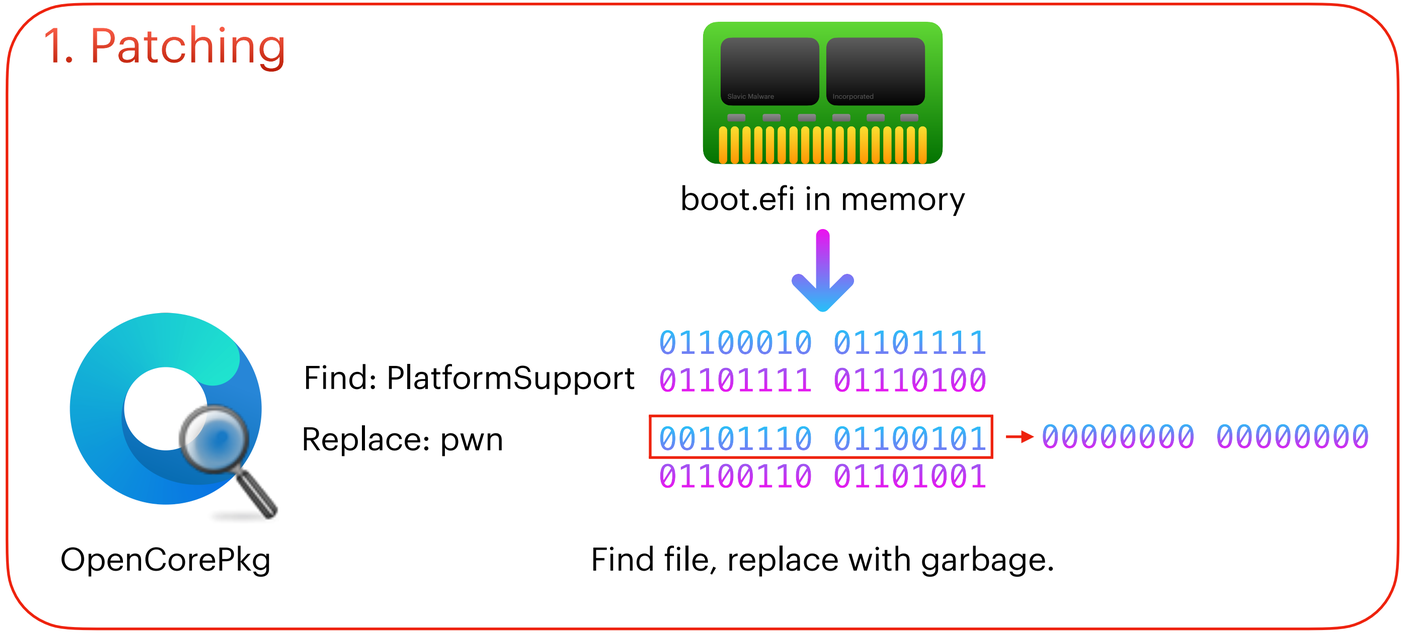
OpenCore Legacy Patcher

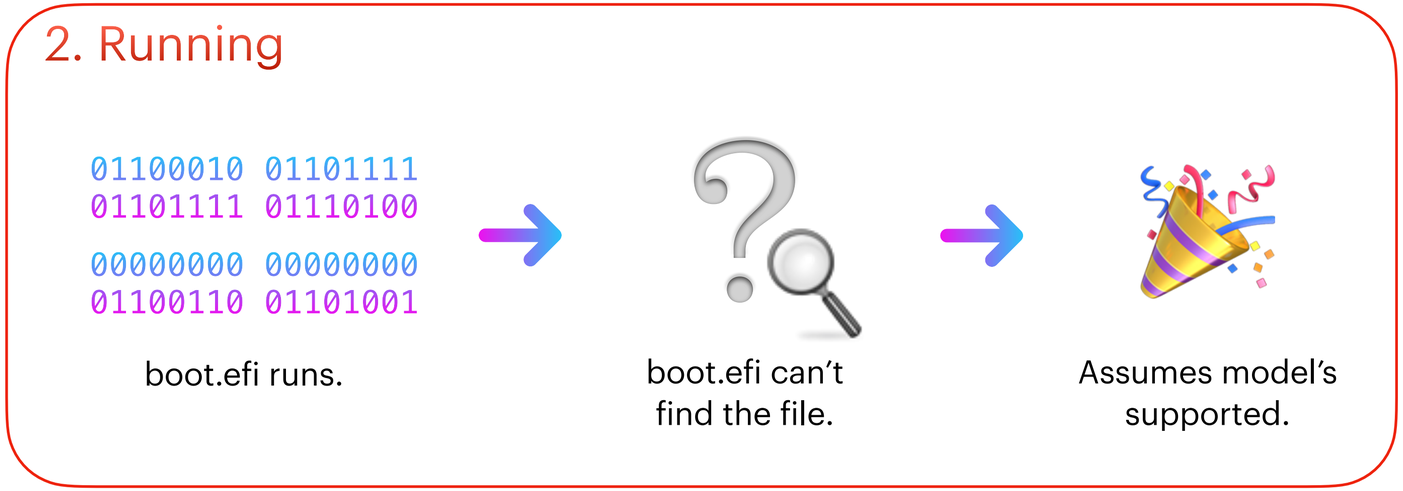
Originally co-authored by DhinakG and Mykola Grymalyuk,
now grown to a small team of passionate developers.

- Introduced with macOS Big Sur support in 2020, and now lived through 4 major OS releases.
- As of macOS Sonoma, supports 83 different Mac models.
- Supports models as old as 2007 to run the latest OS.
- Employs many techniques in the firmware, kernel and user space to restore support.
- Uses Acidanthera's OpenCorePkg backend for firmware and early macOS patching.
  - Where the name originally came from.
- Acidanthera's Lilu Plugin System for additional active kernel patching.
- Encompassed by a Python front end.
  - Over a million lines of Python 🐍
- Free and Open-Source on GitHub!

1. Bypassing firmware checks

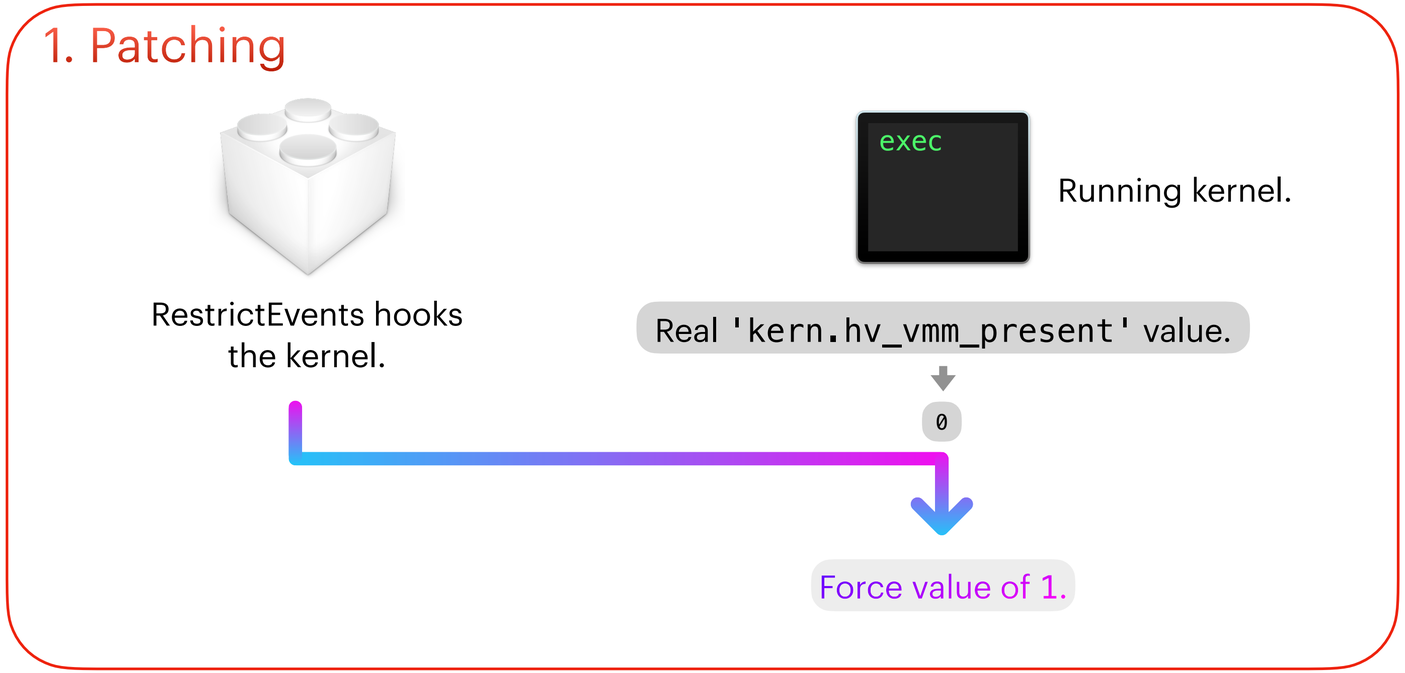
- Uses the OpenCorePkg project for patching.
- Patches the in-memory copy of Apple's boot loader, boot.efi.
- Reroutes the file read for PlatformSupport.plist.
  - PlatformSupport determines what models are supported.
- If the new file is missing, boot.efi assumes that the model is supported and continues.

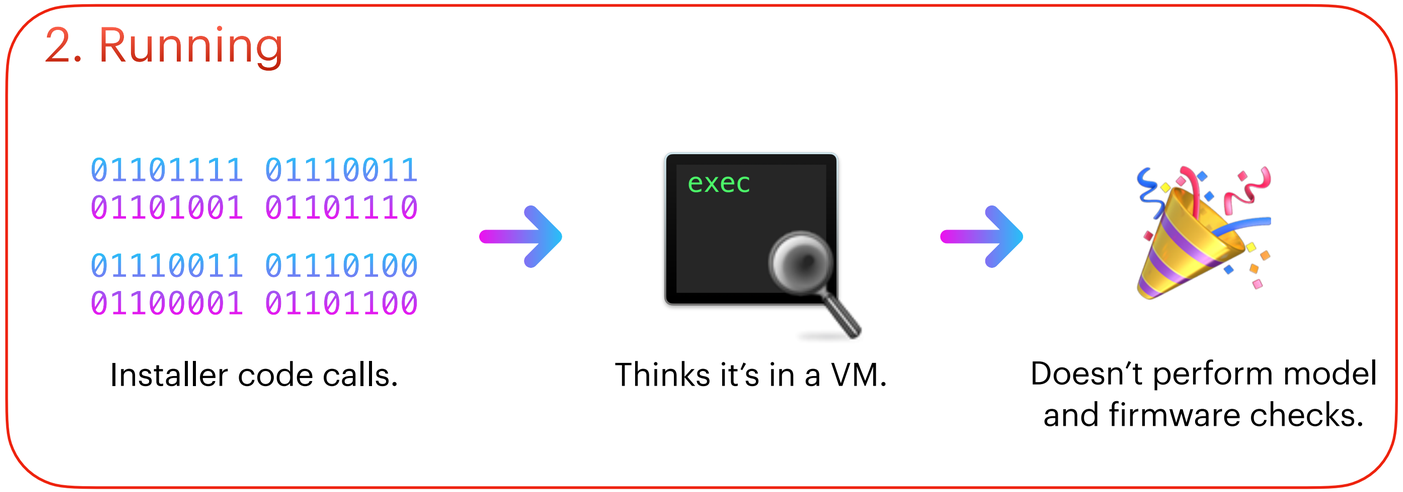




2. Bypassing installer checks

- Using Lilu's plugin, RestrictEvents, in the kernel level (kernel extension).
- Reroutes the value of sysctl's `'kern.hv_vmm_present'` to return 1.
- When this property is set to 1, macOS skips all model and firmware version checks.

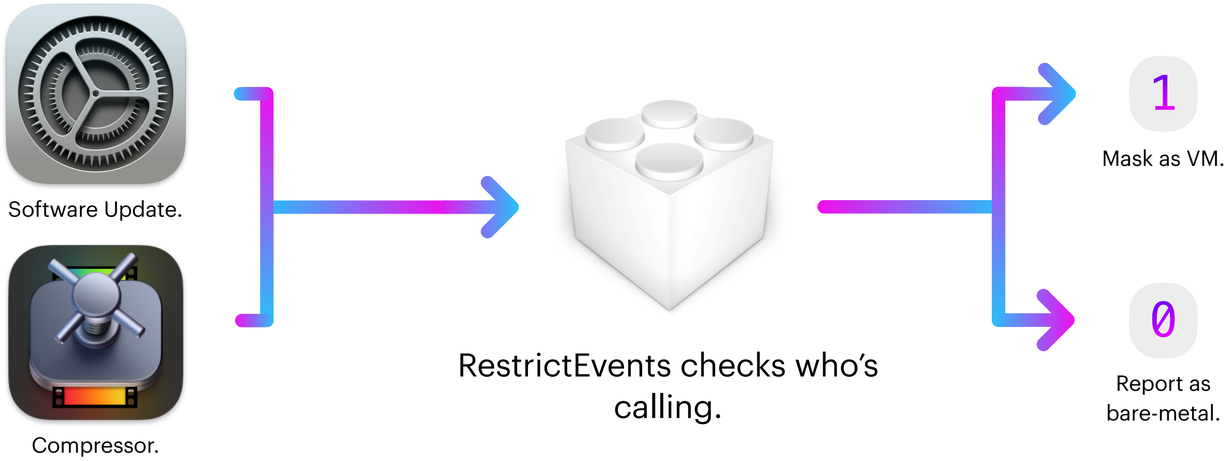




3. Bypassing update limitations

- Similar to bypassing installer checks.
- Instead performed dynamically depending on the calling process.
  - Always set for `'softwareupdated'`, etc.
  - Never set for user-end software with VM checks.
    - Ex. Compressor, Content Caching, etc
- When Software Update detects a VM, it sends `'VMM-x86_64'` instead of the host's model to Apple's servers.
  - Apple then provides the newest OS for Intel Virtual Machines regardless of actual hardware.

1. Patching



2. Running

But are bypasses enough?

Missing hardware drivers

Living in decompiler hell

- Booting an OS is enough of a headache, now make it usable.
- Many core drivers for old hardware are ripped out as soon as a machine is dropped.
- Have to research, reverse and document heavily under documented macOS internals:
  - 3802 LLVM and Metal Graphics.
  - OpenGL Graphics.
  - T1 Security Chip.
  - IO80211 and Broadcom.
- Difficult to inject frameworks dynamically, instead patched into the root volume using the Python front-end.
- Pff, what source code?

Example: T1 Security Chip
Dropped in macOS Sonoma

- Apple's first "Apple Silicon" CPU in a Mac, as a semi co-processor for 2016 and 2017 MacBook Pros.
  - Drove the Camera, Touch Bar, Touch ID, etc.
  - Based on the Apple Watch Series 2 (Watch2,5).
- Kernel side issue:
  - Dropped support for communication with the T1 through KernelRelayHost.kext
  - Requires downgrading following kernel drivers:
    - corecrypto.kext
    - AppleSSE.kext
    - AppleKeyStore.kext
  - Dropped and reinjected by OpenCorePkg.
- Userspace:
  - Dropped communication for EmbeddedOS.
  - Requires downgrading following frameworks and binaries:
    - EmbeddedOSInstall.framework
    - LocalAuthentication.framework
    - biometrickitd
    - Etc...
  - Added via Python front-end, as can't be easily done through a kernel driver.

Problem

No communication between T1 and macOS.

1. Kernel-side

T1 kernel Extensions (kexts).

OpenCorePkg.

Injected into memory (kernel cache) at boot.

2. User-side

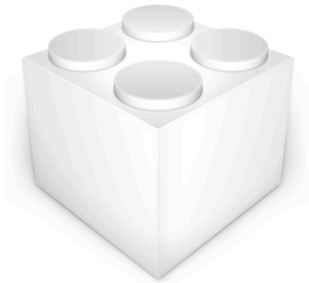
T1 Frameworks and binaries.

OpenCore Legacy Patcher.

Patched onto root volume.

Other neat tricks

- Developing kernel extensions to edit data as it's being paged into memory using the Lilu patching engine.
  - Unlock features limited to newer Macs.
    - AirPlay to Mac, SideCar, Continuity Camera, etc.
    - Performed by FeatureUnlock.kext
  - Allow (very) legacy hardware to be installed through compatibility binaries normally for Apple Silicon (ex. Rosetta 2).
    - Performed by CryptexFixup.kext
- Userspace daemons to help repair macOS after updates.



Why need all these different tools?

- General goal of each part:
  - 1. Firmware and early kernel boot:
    - OpenCorePkg: Firmware compatibility, and injecting kexts into the first stage kernel cache.
  - 2. In-kernel:
    - Lilu: Patching engine while the kernel is running (kernel extension).
    - Used by FeatureUnlock and RestrictEvents
  - 3. In-userspace/on-disk
    - Python front end (OpenCore Legacy Patcher)
    - Add files kexts not compatible in the Boot KC to the Auxiliary KC
    - Add additional frameworks removed from macOS (as this can’t be easily done in a kext)

Putting the pieces together: How it all boots

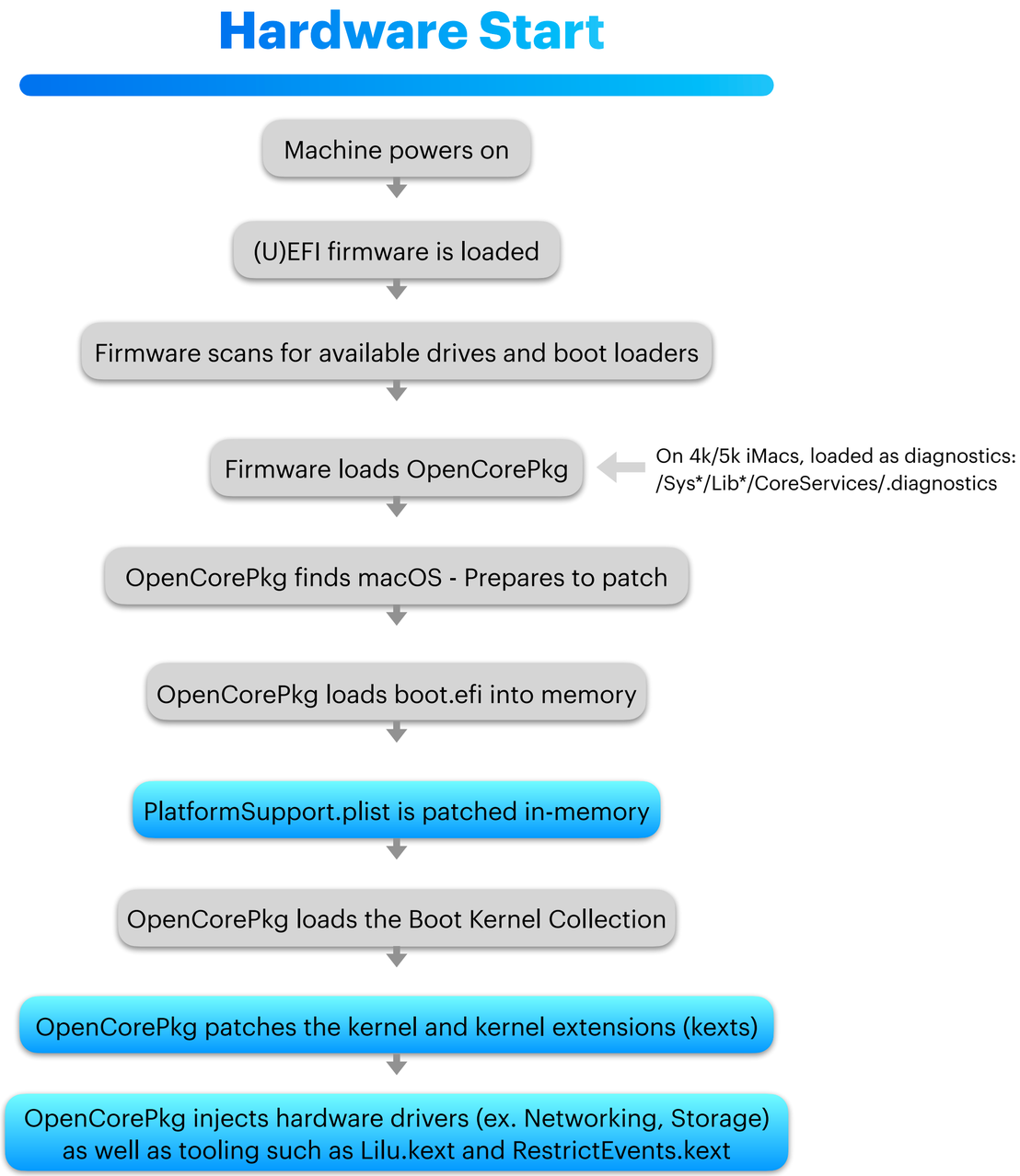
1. Mac Powers On

- 1. Mac searches for FAT32 and other types of volumes to boot.
- 2. Mac matches on OpenCorePkg's boot entry.
  - On 4k/5k iMacs, hidden through diagnostics chain loading to retain Apple's signature with an encrypted file buffer.
    - Allows us to enable Apple-only features, like Dual DisplayPort Timing Controller support for high resolution.
- 3. OpenCore loads boot.efi into memory.
  - Ex. Patches out PlatformSupport.plist
- 4. Kernel Cache is loaded into memory.
  - Cache type: Boot KC, kernel and kexts required to boot.
  - OpenCore patches kernel and kext functions, and injects additional kexts for our old hardware.

OpenCorePkg's Magic

Lilu and Plugins' Magic

OpenCore Legacy Patcher's Magic 🐍



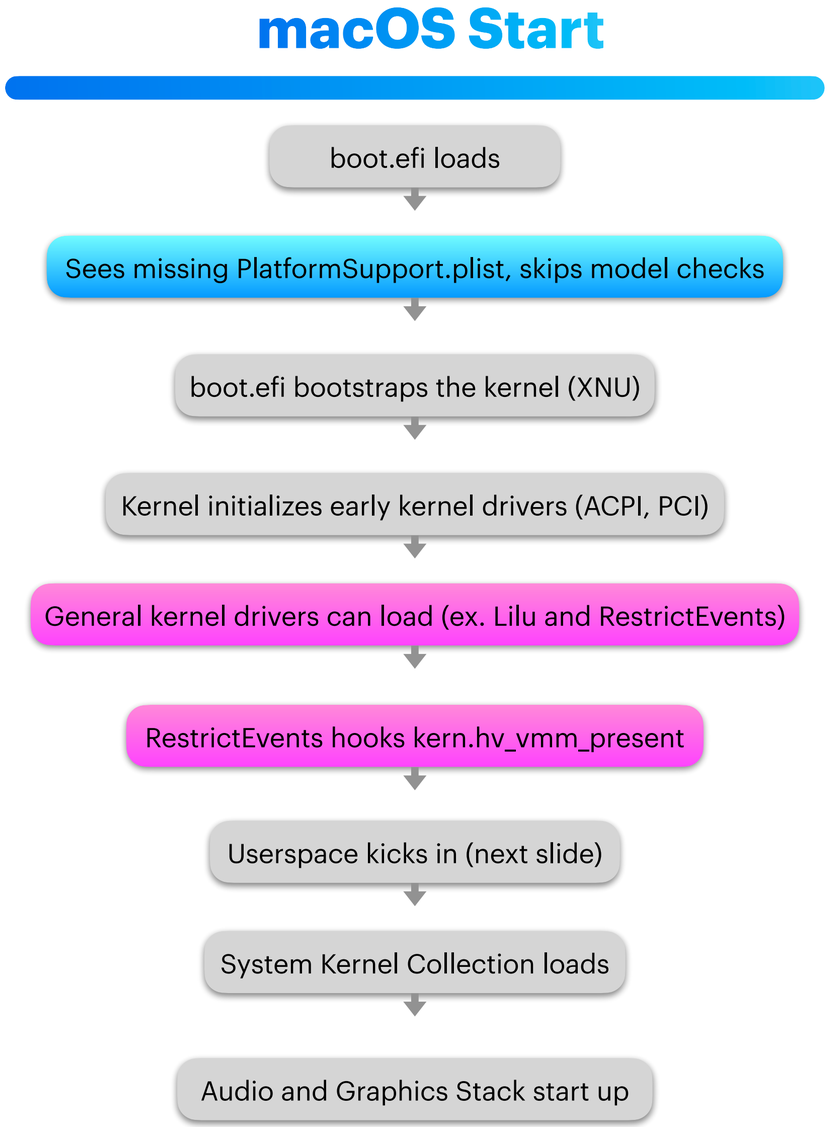
2. boot.efi and XNU

- 1. boot.efi starts and checks for model support.
- 2. Kernel starts up.
- 3. Initializes early kernel extensions.
  - ACPI, PCI, etc.
- 4. Loads general kernel extensions.
  - Includes Lilu, our in-kernel patching engine.
- 5. Lilu and plugins starts and patch as needed.
  - RestrictEvents.kext detects if in an installer or normal OS.
  - If installer, force `'kern.hv_vmm_present'` to return 1 always. Otherwise, check calling process.
- 6. Userspace starts to spin up.
- 7. Additional kernel caches are loaded, and Lilu plugins patch extensions in them as well.
  - Cache type: System KC.
  - Holds additional kexts, ex. audio and graphics stack.
  - Referred to as Pageable KC in XNU.

OpenCorePkg's Magic

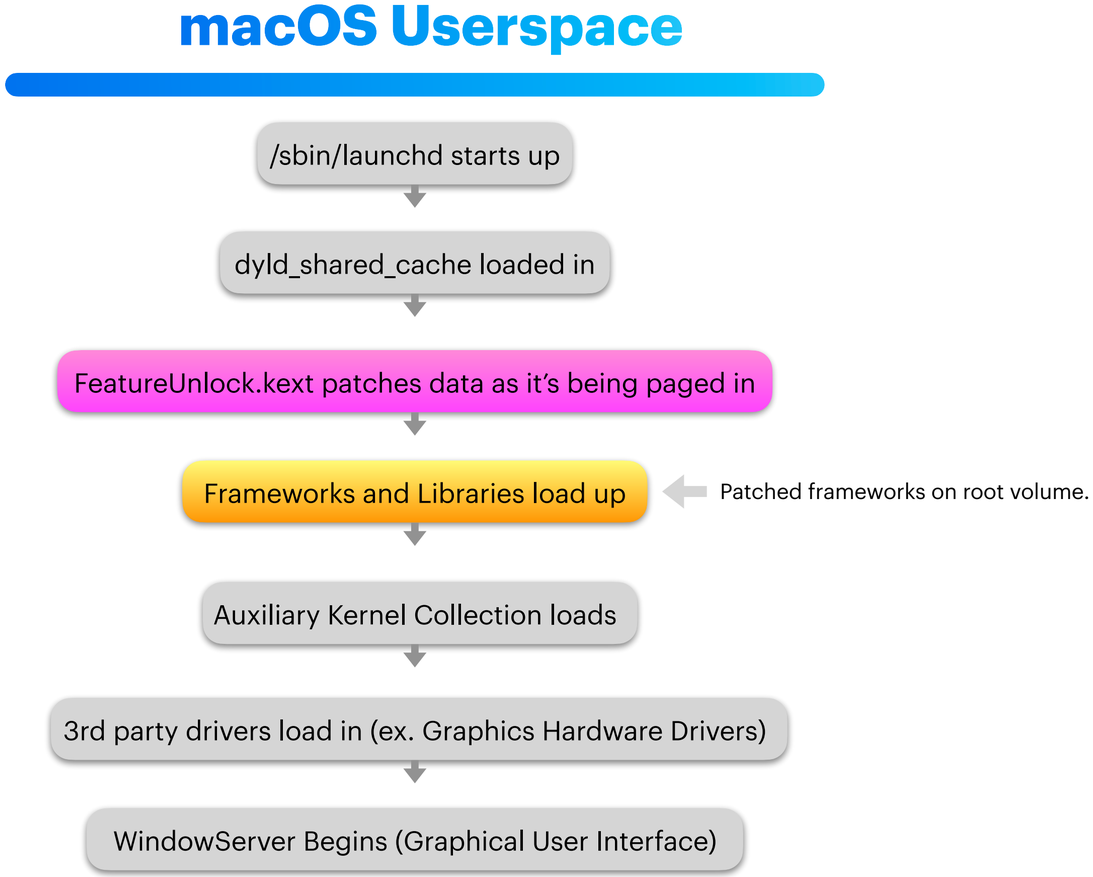
Lilu and Plugins' Magic

OpenCore Legacy Patcher's Magic 🐍



3. Userspace Kicks In

- 1. launchd starts up (first userspace binary).
- 2. dyld_shared_cache loaded.
  - Massive blob with most frameworks fused in.
  - FeatureUnlock.kext patches paged in memory, removing model checks for AirPlay to Mac, Continuity Camera, etc.
- 4. Final kernel cache loaded in:
  - Auxiliary KC, holds 3rd party kexts and some kexts from OpenCore Legacy Patcher.
  - Ex. Kexts that can’t be injected in the BootKC, and can’t easily be modded into SysKC.
- 5. macOS UI kicks in.



OpenCorePkg’s Magic

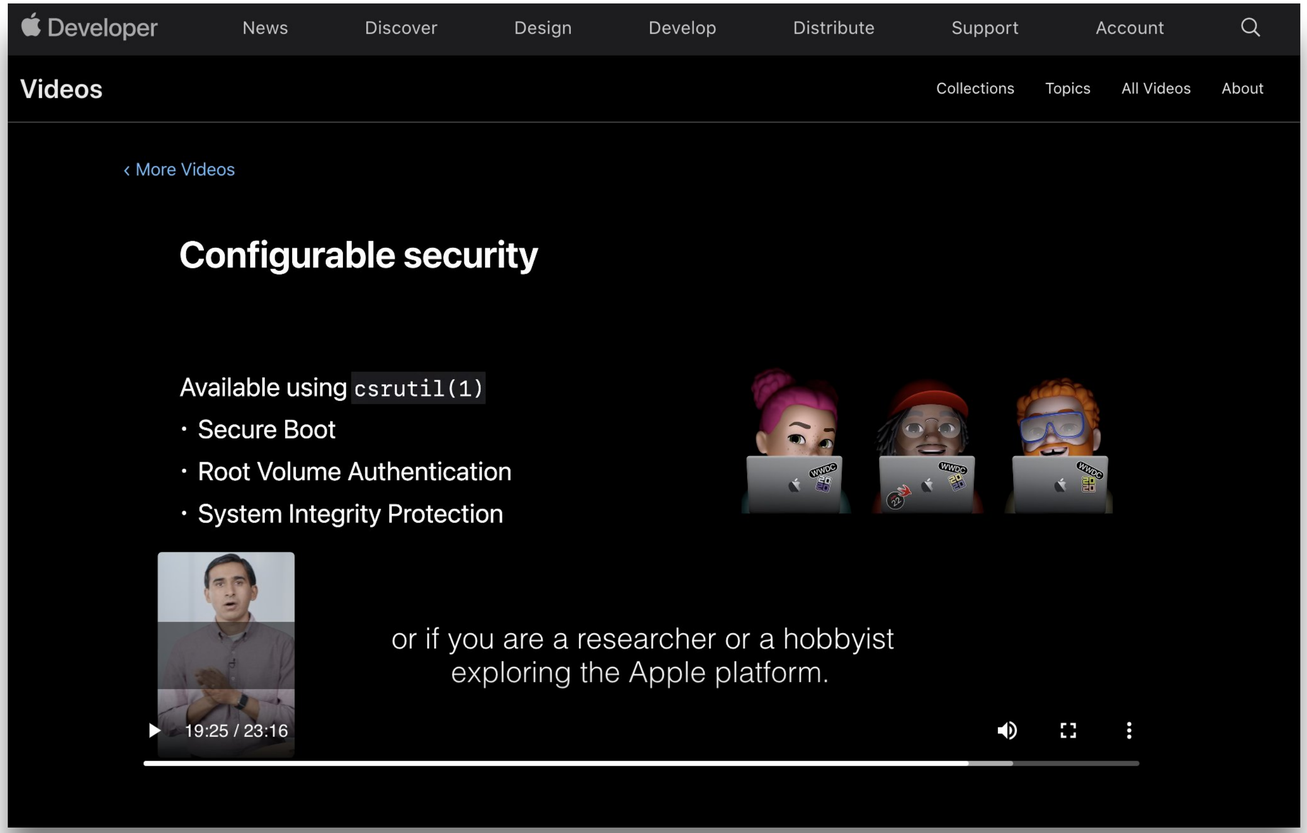
Lilu and Plugins’ Magic

OpenCore Legacy Patcher’s Magic 🐍

Ok, but is Apple letting you do this?

Apple's (somewhat reluctant) support of OS modding

- With every new security improvement in macOS, there's generally some way to disable or at least lower the restriction.
- However as Macs become more iOS-like, there are concerns that this may be locked down in the future.
- Apple Silicon Macs are still thankfully configurable just like with Intel.



WWDC 2020: Explore the new system architecture of Apple silicon Macs

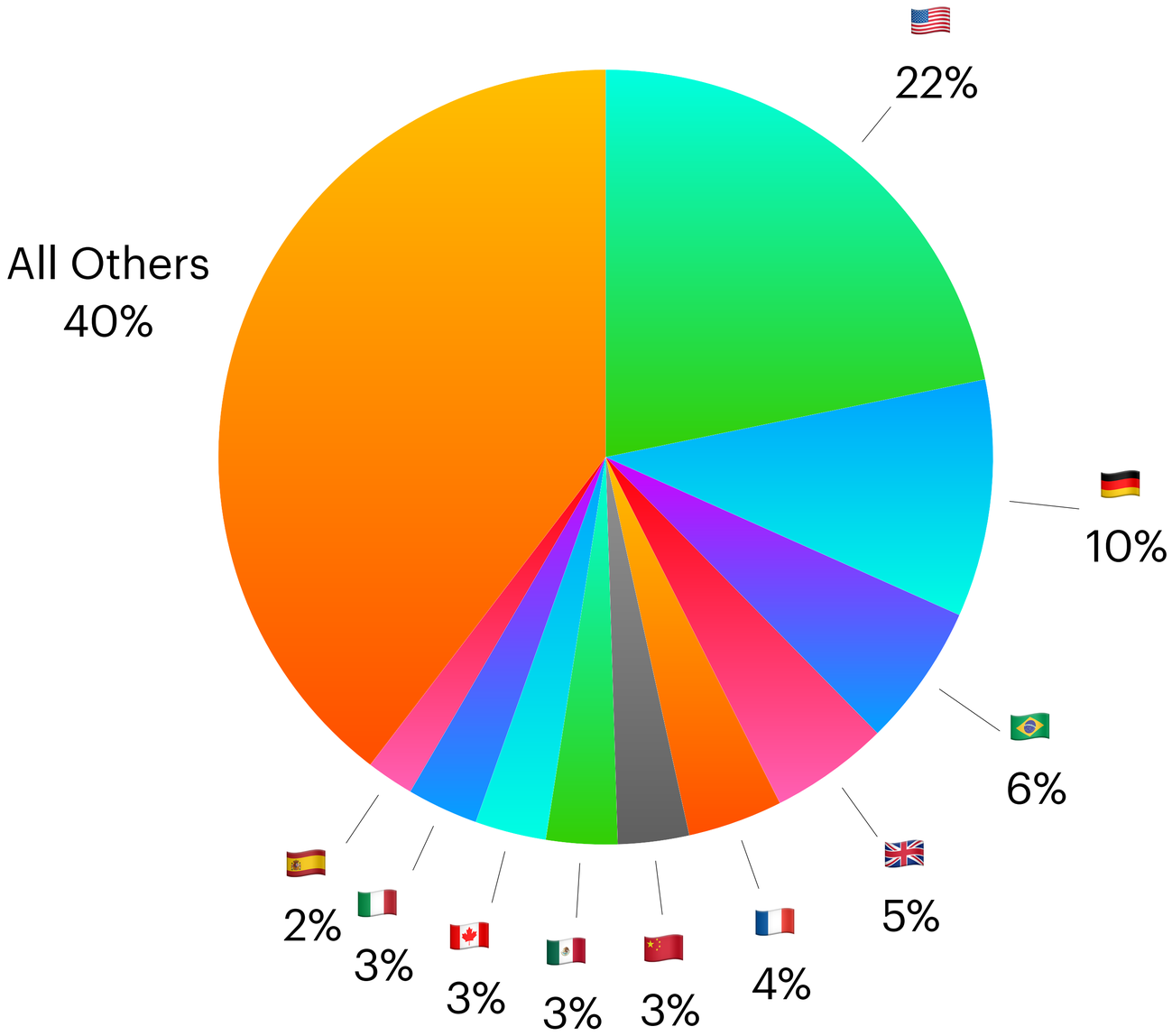
Currently zero threats from Apple 🎉

Shortcomings of macOS Patchers

- Beholden to Apple's mercy.
- Developing against a closed source OS.
  - Open-source parts of the kernel (XNU) are incomplete.
  - 95% of the work is reverse engineering and documenting.
  - Requires skilled developers with deep understanding of macOS internals and background in software disassemblers.
- To restore hardware support, some security features need to be lowered.
- Expecting a small group of hobbyists to maintain the same level of work as a trillion-dollar company.

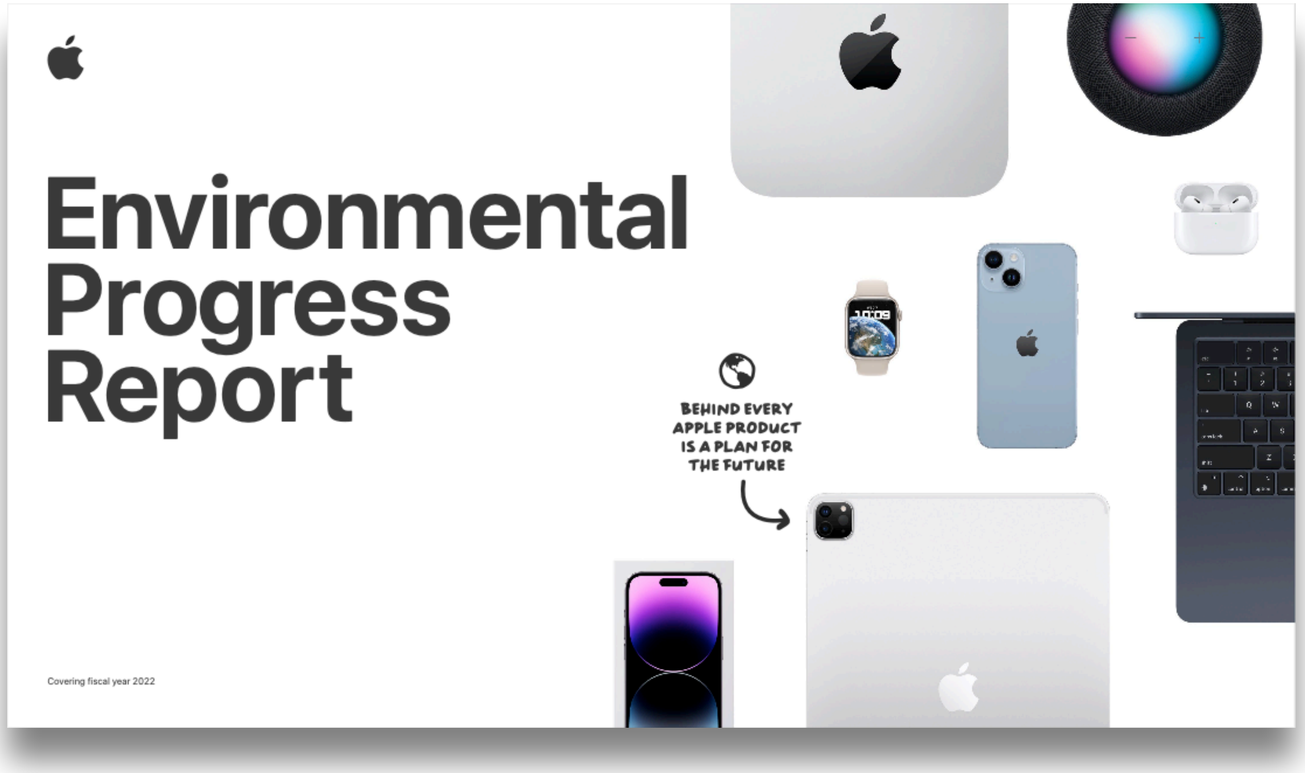
The effect of macOS Patchers
With a global audience

- Allows machines to be used longer, reducing unnecessary E-Waste.
- Lower income families aren’t forced to buy new hardware when they’re not ready.
  - Don’t need a new machine to run software no longer supported on EoL versions of macOS.
- Brazil is the 3rd largest user base of OpenCore Legacy Patchers, due to the difficulties with affording new hardware.
  - iPhone 15 is almost x2 more expensive in Brazil than the United States. 💸
- User base since reporting added in May 2023 is at 650,000 users.
- Total downloads in project’s lifetime, 3 years, is 6.7 million.



Apple and the Environment

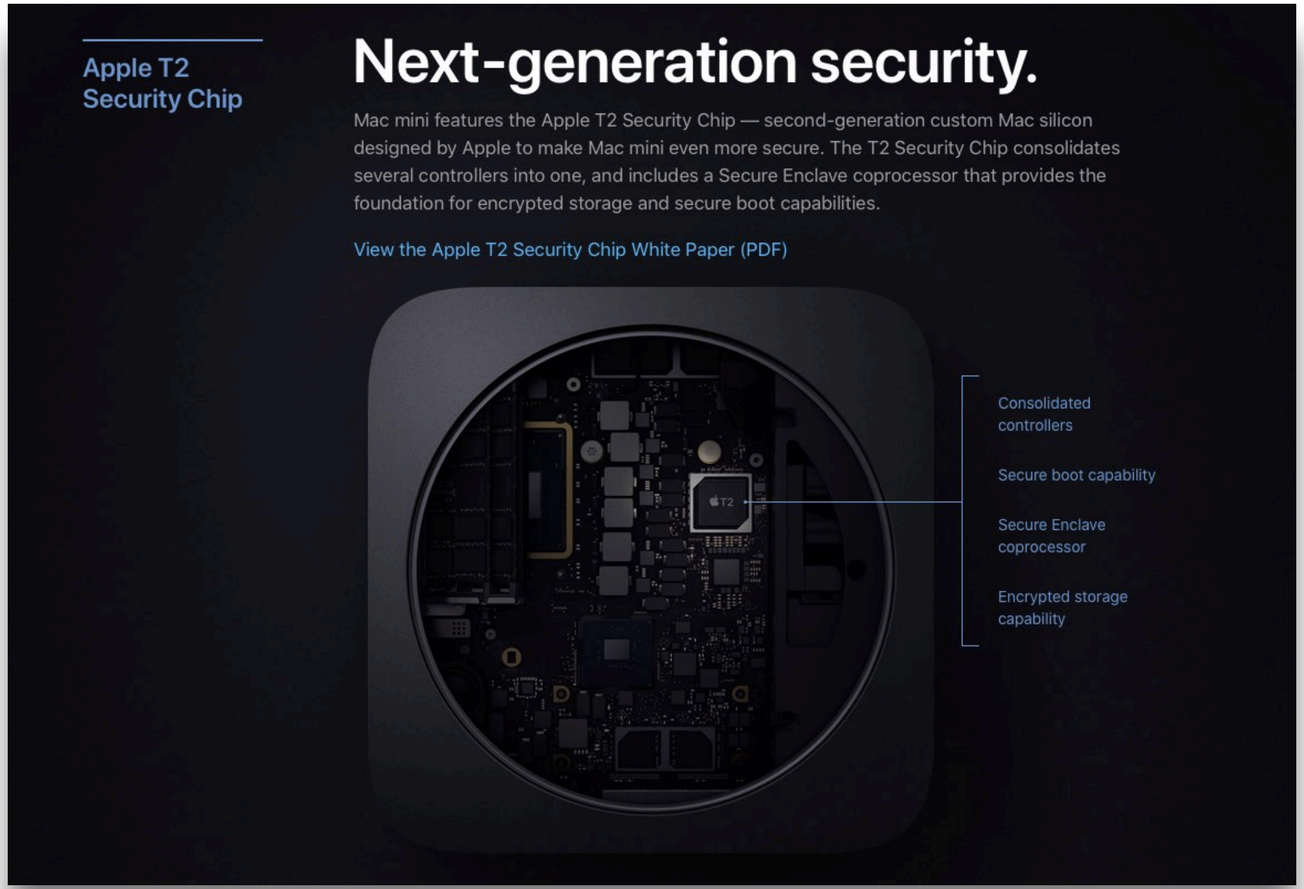
- Apple’s been pushing for reduced carbon foot print.
- By letting these machines become obsolete, they add to the landfill.
- Google will be supporting ChromeOS devices for 10 years of software updates, Macs should get the same (and iPhones too!).
- Services users they’re leaving behind:
  - Can’t use all these fancy iCloud features without newer OSes.
- You’re a trillion dollar company, you have the resources.



Apple’s Environmental Progress Report - 2023

In defence of Apple

- By keeping older hardware supported, inherently allowing vulnerable machines to stay around longer.
- Includes hardware vulnerabilities:
  - Intel: Spectre and Meltdown.
  - T2: Checkma8
- As well as software:
  - Lack of proper Secure Boot support on pre-T2 machines.
    - OBTS v6.0: The Nightmare of Apple's OTA Update - Mickey Jin
  - Having to maintain more drivers, ie. more vectors of attack.
- Can also prevent newer machines from achieving their full potential.
  - Having to keep around fallback code for old machines lacking new security chips (T1, T2 and Apple Silicon)
  - Ex. Lack of Device Attestation on pre-T1 machines and MDM.



Mac mini (2018) - T2 Security Chip

Conclusion

Conclusion

- Old OSes don’t last long, and leave you vulnerable (duh).
- Old Macs are dying at a quicker rate.
  - Unknown if this is due to accelerated death of Intel, or if this will be the new norm.
- To get more life, you have to install Windows/Linux or turn to hobbyist projects. No love from Apple.
- Apple is contributing to their carbon footprint by ignoring their old machines.

And thank you to the many amazing projects, developers and community members keeping our old machines running!

Ausdauersportler, IronApple, UHDBit, thatstella7922, crystall1nedev, Socamx, Paradox94, ASentientHedgehog, Mr.Macintosh, Jazzzny, Flagers, EduCovas, ASentientBot, DhinakG
Joevt, CDF, StarPlayrX, BarryKN, Jackluke, dosdude1, Syncretic, Minh Ton, BenSova, PMHeart, Goldfish64, Mike Beaton, Marvin Häuser, Vitaly Cheptsov

And so many more keeping the community alive!

Q&A

Thank you for listening to me ramble!

- OpenCore Legacy Patcher Info:
  - GitHub Repo: www.github.com/dortania/OpenCore-Legacy-Patcher
- Personal Info:
  - Name: Mykola Grymalyuk
  - Email: khronokernel@icloud.com
  - Twitter: @khronokernel
  - Blog: khronokernel.com
  - Linkedin: www.linkedin.com/in/mykola-grymalyuk

References

Links

- OBTS v4.0: "n-1 and n-2: Should we really trust in you?" - Josh Long
  - https://www.youtube.com/watch?v=o5KUvgXHOFU
- Apple Security Contents:
  - macOS 12.7.1: https://support.apple.com/en-ca/HT213983
  - macOS 13.6.1: https://support.apple.com/en-ca/HT213985
  - macOS 14.1: https://support.apple.com/en-us/HT213984
- OWC: Other World Computing Releases XPostFacto 4.0
  - https://www.owc.com/news/other-world-computing-releases-xpostfacto-4-0
- WWDC 2020: Explore the new system architecture of Apple silicon Macs
  - https://developer.apple.com/videos/play/wwdc2020/10686
- HelloSafe: How Much Does an iPhone 15 Cost Across The World?
  - https://hellosafe.ca/en/telecommunications/cell-phone-plans/iphone-prices-map
- Google: Chromebooks will get 10 years of automatic updates
  - https://blog.google/outreach-initiatives/education/automatic-update-extension-chromebook
- Apple: Environmental Progress Report - 2023
  - https://www.apple.com/environment/pdf/Apple_Environmental_Progress_Report_2023.pdf
- Apple: Mac mini (2018) - T2 Security Chip
  - https://web.archive.org/web/20181206234134/https://www.apple.com/mac-mini

References

Additional Images

- Apple Models, Kexts, executables and other icons:
  - macOS' CoreTypes.bundle
  - macOS installers
  - Compressor
- Virtual Machine icons:
  - VMware Fusion: https://www.vmware.com/products/fusion.html
  - UTM: https://getutm.app
  - Parallels: https://www.parallels.com
- Decompiler icons:
  - Ghidra: https://ghidra-sre.org
  - IDA: https://hex-rays.com/ida-pro
  - Hopper Disassembler: https://www.hopperapp.com
- OpenCorePkg: https://github.com/acidanthera/OpenCorePkg
- OpenCore Legacy Patcher: https://github.com/dortania/OpenCore-Legacy-Patcher
- Additional Patcher icons: See slide 7
- User icons: See slide 44