

# Project Zero

News and updates from the Project Zero team at Google

Tuesday, January 7, 2020

## Policy and Disclosure: 2020 Edition

Posted by Tim Willis, Project Zero

At Project Zero, we spend a lot of time discussing and evaluating vulnerability disclosure policies and their consequences for users, vendors, fellow security researchers, and software security norms of the the larger industry. We're very happy with how well our disclosure policy has worked over the past five years. We've seen some big improvements to how quickly vendors patch serious vulnerabilities, and now 97.7% of our vulnerability reports are fixed within our 90 day disclosure policy.

In saying that, it's a complex and often controversial topic that is frequently discussed both inside and outside of the team. We often receive feedback from vendors that Project Zero works closely with regarding our current policies: sometimes it's things they want us to change, other times it's how our work has positively impacted their work and users. Conversations like these have helped develop our policies over the years. For example, we introduced our [14-day grace-period](#) in 2015 after helpful discussions with various vendors.

We recently reviewed our policies and the goals we hope to accomplish with our disclosure policy. As a result of that review, we have decided to make some changes to our vulnerability disclosure policy in 2020. We will start by describing the changes to the policy, and then discuss the rationale behind these changes.

## Summary of changes for 2020

For vulnerabilities reported starting January 1, 2020, we are changing our Disclosure Policy: **Full 90 days by default, regardless of when the bug is fixed**.

*Fix a bug in 20 days? We will release all details on Day 90.*

*Fix a bug in 90 days? We will release all details on Day 90.*

If there is mutual agreement between the vendor and Project Zero, bug reports can be opened to the public before 90 days elapse. For example, a vendor wants to synchronize the opening of our tracker report with their release notes to minimize user confusion and questions.

We will try this policy for 12 months, and then consider whether to change it long term.

The current list of changes for 2020:

| 2019  | 2020 Trial  |
|---|---|
| 1. 90 days or when the bug is fixed (decided by researcher discretion), whichever is the earliest.  | 1. Full 90 days, regardless of when the bug is fixed. Earlier disclosure with mutual agreement.   |
| 2. Policy goal: <ul style="list-style-type: none"><li>Faster patch development</li></ul>  | 2. Policy goals: <ul style="list-style-type: none"><li>Faster patch development</li><li>Thorough patch development</li><li>Improved patch adoption</li></ul>      |
| 3. Inconsistent handling of incomplete fixes. Such issues are either filed as separate vulnerabilities or added to existing reports at researcher discretion. | 3. Details of incomplete fixes will be reported to the vendor and added to the existing report (which may already be public) and will not receive a new deadline. |
| 4. Bugs fixed in the grace period* would be opened to the public sometime after a patch was released.   | 4. Project Zero tracker reports are immediately opened when patched during the grace period*.   |
| 5. Project Zero tracker reports are opened at researcher discretion after the deadline expires.   | 5. Project Zero tracker reports are opened automatically on Day 90 (or earlier under mutual agreement).   |

\* The grace period is an additional 14 days that a vendor can request if they do not expect that a reported vulnerability will be fixed within 90 days, but do expect it to be fixed within 104 days. If a grace period is requested, and a bug is fixed between 90 and 104 days after it was reported, bug details will be released on the day it is fixed. Grace periods will not be granted for vulnerabilities that are expected to take longer than 104 days to fix. Note that the [seven-day deadline](#) for vulnerabilities that are being actively exploited "in the wild" will remain unchanged.

We're constantly considering whether our policies are in the interest of user security, and we believe this change is a further step in the right direction. We also think it's simple, consistent and fair.

## Rationale on changes for 2020

For the last five years, the team has used its vulnerability disclosure policy to focus on one primary goal: **Faster patch development**.

We want to make attacks using zero-day exploits more costly. We do this through the lens of offensive vulnerability research and evidence of how real attackers behave. This involves discovering and reporting a large number of security vulnerabilities, and through our experience with this work, we realised that faster patch development and patch deployment were very important and areas for industry improvement.

If patches take a long time to develop and deploy, then we quickly fall behind the curve: more bugs are introduced than vendors can fix and a herculean effort is required to get things back on track.

We also regularly uncover cases of "bug collisions" with our research. This is where the vulnerability we discovered was previously found and exploited by a real attacker. Knowing that the vulnerabilities that we find are often already being secretly exploited to harm users creates a sense of urgency, and so we ask vendors to fix issues as quickly as possible.

After five years of applying a 90-day disclosure deadline, we're proud of the results we've seen: vulnerabilities are being fixed faster than ever. For example, around the time Project Zero started in 2014, some issues were taking upwards of six months to fix. Fast forward to 2019, and [97.7% of our issues are fixed under deadline](#). That said, we know there is still room for improvement, both in industry-wide patch development speed and over the entire vulnerability management lifecycle.

## Revisiting our underlying policy principles and goals

We recently spent some time articulating the underlying principles of our policies:

- Simple.** Simplicity is important because we want to be easily understood. We also want to operate at scale, otherwise it's even harder to aggressively push all of industry to do better.
- Consistent.** We want to be reliable and predictable. We want to apply our deadlines in a deterministic manner without fear or favour, demonstrating that we mean what we say. The bar for any exception/inconsistency needs to remain extremely high (note: we've only had [two exceptions](#) in the five year history of the team).
- Fair.** We want to be equitable, balanced and impartial. We don't want to be in a position where different vendors (including Google!) get a form of preferential treatment. The same rules should apply to everyone.

We also realised as part of our discussions that there are two other policy goals that we wanted to include. Based on those discussions, here are our policy goals for 2020:

- Faster patch development (existing):** We want vendors to develop patches quickly and have processes in place to get them into the hands of end users. We will continue to pursue this with urgency.
- Thorough patch development (new):** Too many times, we've seen vendors patch reported vulnerabilities by "papering over the cracks" and not considering variants or addressing the root cause of a vulnerability. One concern here is that our policy goal of "faster patch development" may exacerbate this problem, making it far too easy for attackers to revive their exploits and carry on attacking users with little fuss.
- Improved patch adoption (new):** End user security doesn't improve when a bug is found, and it doesn't improve when a bug is fixed. It improves once the end user is aware of the bug and typically patches their device. To this end, improving timely patch adoption is important to ensure that users are actually acquiring the benefit from the bug being fixed.

This new policy direction for 2020 gives clear incentives for vendors, especially those that have raised the following issues with us in the past.

- Since we founded Project Zero, some vendors hold the view that our disclosures prior to significant patch adoption are harmful. Though we disagree (since this information is already public and being used by attackers per [our FAQ here](#)), under this new policy, we expect that vendors with this view will be **incentivised to patch faster**, as faster patches will allow them "additional time" for patch adoption.
- The full 90 day window is available to perform root cause and variant analysis. We expect to see iterative and **more thorough patching** from vendors, removing opportunities that attackers currently have to make minor changes to their exploits and revive their zero-day exploits.
- We're also being explicit on **improving patch adoption**, since we're incentivising that vendors should be able to offer updates and encourage installation to a large population within 90 days.

We also really like that the new policy will improve the consistency of our disclosure process, while also remaining simple and fair. For example, some vendors considered our determination of when a vulnerability was fixed as unpredictable, especially when working with more than one researcher on the team at a given time. They saw it as a barrier to working with us on larger problems, so we're going to remove the barrier and see if things improve. We hope this experiment will encourage vendors to be transparent with us, to share more data, build trust and improve collaboration.

Disclosure policy is a complex topic with many trade-offs to be made. We don't expect this policy to please everyone, but we're optimistic that it will improve on our current policy, encompasses a good balance of incentives and will be a positive step for user security. We plan to re-evaluate whether it is accomplishing our policy goals in late 2020.

Posted by Tim at 9:11 AM



No comments:

## Post a Comment

Enter your comment...

Comment as: Google Account

Publish

Preview

Newer Post Older Post

Subscribe to: [Post Comments \(Atom\)](#)

Search This Blog

Search

Pages

- About Project Zero
- Working at Project Zero
- 0day "In the Wild"
- 0day Exploit Root Cause Analyses
- Vulnerability Disclosure FAQ

Archives

- 2021
- In-the-Wild Series: October 2020 0-day discovery (Mar)
  - Déjà vu-lerability (Feb)
  - A Look at iMessage in iOS 14 (Jan)
  - Windows Exploitation Tricks: Trapping Virtual Memory... (Jan)
  - The State of State Machines (Jan)
  - Hunting for Bugs in Windows Mini-Filter Drivers (Jan)
  - In-the-Wild Series: Android Post-Exploitation (Jan)
  - In-the-Wild Series: Windows Exploits (Jan)
  - In-the-Wild Series: Android Exploits (Jan)
  - In-the-Wild Series: Chrome Infinity Bug (Jan)
  - In-the-Wild Series: Chrome Exploits (Jan)
  - Introducing the In-the-Wild Series (Jan)

2020

- An iOS hacker tries Android (Dec)
- An iOS zero-click radio proximity exploit odyssey (Dec)
- Oops, I missed it again! (Nov)
- Enter the Vault: Authentication Issues in HashiCor... (Oct)
- Announcing the Fuzzilli Research Grant Program (Oct)
- Attacking the Qualcomm Adreno GPU (Sep)
- JITSploitation I: A JIT Bug (Sep)
- JITSploitation II: Getting Read/Write (Sep)
- JITSploitation III: Subverting Control Flow (Sep)
- MMS Exploit Part 5: Defeating Android ASLR, Gettin... (Aug)
- Exploiting Android Messengers with WebRTC: Part 3 (Aug)
- Exploiting Android Messengers with WebRTC: Part 2 (Aug)
- MMS Exploit Part 4: MMS Primer, Completing the ASLR... (Aug)
- Exploiting Android Messengers with WebRTC: Part 1 (Aug)
- The core of Apple is PPL: Breaking the XNU kernel&#39;... (Jul)
- One Byte to rule them all (Jul)
- Root Cause Analyses for 0-day In-the-Wild Exploits (Jul)
- Detection Deficit: A Year in Review of 0-days-Used... (Jul)
- MMS Exploit Part 3: Constructing the Memory Corrup... (Jul)
- MMS Exploit Part 1: Introduction to the Samsung S7m... (Jul)
- How to uncover a 0-day in 4 hours or less (Jul)
- FF Sandbox Escape (CVE-2020-12388) (Jun)
- A survey of recent iOS kernel exploits (Jun)
- Fuzzing ImageIO (Apr)
- You Won&#39;t Believe what this One Line Change Did to... (Apr)
- TFW you get really excited you patch-diffed a 0-day... (Apr)
- Escaping the Chrome Sandbox with RIDL (Feb)
- Mitigations are attack surface, too (Feb)
- A day/W/W Several months in the life of Project Ze... (Feb)
- Part II: Returning to Adobe Reader symbols on macOS (Jan)
- Remote iPhone Exploitation Part 3: From Memory Cor... (Jan)
- Remote iPhone Exploitation Part 2: Bringing Light... (Jan)
- Remote iPhone Exploitation Part 1: Poking Memory v... (Jan)
- Policy and Disclosure: 2020 Edition (Jan)

2019

- Calling Local Windows RPC Servers from .NET (Dec)
- SocketPuppet: A Walkthrough of a Kernel Exploit for... (Dec)
- Bad Binder: Android In-The-Wild Exploit (Nov)
- KTRW: The journey to build a debuggable iPhone (Oct)
- The story of Adobe Reader symbols (Oct)
- Windows Exploitation Tricks: Spoofing Name... (Sep)
- A very deep dive into iOS Exploit chains found in ... (Aug)
- In-the-wild iOS Exploit Chain 1 (Aug)
- In-the-wild iOS Exploit Chain 2 (Aug)
- In-the-wild iOS Exploit Chain 3 (Aug)
- In-the-wild iOS Exploit Chain 4 (Aug)
- In-the-wild iOS Exploit Chain 5 (Aug)
- Implant Teardown (Aug)
- JSC Exploits (Aug)
- The Many Possibilities of CVE-2019-8646 (Aug)
- Down the Rabbit-Hole... (Aug)
- The Fully Remote Attack Surface of the iPhone (Aug)
- Trashing the Flow of Data (May)
- Windows Exploitation Tricks: Abusing the User-Mode... (Apr)
- Virtually Unlimited Memory: Escaping the Chrome Sa... (Apr)
- Splitting atoms in XNU (Apr)
- Windows Kernel Logic Bug Class: Access Mode Mismat... (Mar)
- Android Messaging: A Few Bugs Short of a Chain (Mar)
- The Curious Case of Convexity Confusion (Feb)
- Examining Pointer Authentication on the iPhone XS (Feb)
- voucher\_swap: Exploiting MIG reference counting in... (Jan)
- Taking a page from the kernel&#39;s book: A TLB issue... (Jan)

2018

- On VBScript (Dec)
- Searching statically-linked vulnerable library fun... (Dec)
- Adventures in Video Conferencing Part 5: Where Do... (Dec)
- Adventures in Video Conferencing Part 4: What Didn... (Dec)
- Adventures in Video Conferencing Part 3: The Even... (Dec)
- Adventures in Video Conferencing Part 2: Fun with... (Dec)
- Adventures in Video Conferencing Part 1: The Wild... (Dec)
- Injecting Code into Windows Protected Processes us... (Nov)
- Heap Feng Shader: Exploiting SwiftShader in Chrome (Oct)
- Déjà-XNU (Oct)
- Injecting Code into Windows Protected Processes us... (Oct)
- 365 Days Later: Finding and Exploiting Safari Bugs... (Oct)
- A cache invalidation bug in Linux memory management (Sep)
- QATmeal on the Universal Cereal Bus: Exploiting An... (Sep)
- The Problems and Promise of WebAssembly (Aug)
- Windows Exploitation Tricks: Exploiting Arbitrary ... (Aug)
- Adventures in vulnerability reporting (Aug)
- Drawing Outside the Box: Precision Issues in Graph... (Jul)
- Detecting Kernel Memory Disclosure - Whitepaper (Jun)
- Bypassing Mitigations by Attacking JIT Server in M... (May)
- Windows Exploitation Tricks: Exploiting Arbitrary... (Apr)
- Reading privileged memory with a side-channel (Jan)

2017

- aPAColypse now: Exploiting Windows 10 in a Local N... (Dec)
- Over The Air - Vol. 2, Pt. 3: Exploiting The Wi-Fi... (Oct)
- Using Binary Diffing to Discover Windows Kernel Me... (Oct)
- Over The Air - Vol. 2, Pt. 2: Exploiting The Wi-Fi... (Oct)
- Over The Air - Vol. 2, Pt. 1: Exploiting The Wi-Fi... (Sep)
- The Great DOM Fuzz-off of 2017 (Sep)
- Bypassing VirtualBox Process Hardening on Windows (Aug)
- Windows Exploitation Tricks: Arbitrary Directory C... (Aug)
- Trust Issues: Exploiting TrustZone TEEs (Jul)
- Exploiting the Linux kernel via packet sockets (May)
- Exploiting .NET Managed DCOM (Apr)
- Exception-oriented exploitation on iOS (Apr)
- Over The Air: Exploiting Broadcom's Wi-Fi Stack (P... (Apr)
- Notes on Windows Uniscribe Fuzzing (Apr)
- Pandavirtualization: Exploiting the Xen hypervisor (Apr)
- Over The Air: Exploiting Broadcom's Wi-Fi Stack (P... (Apr)
- Project Zero Prize Conclusion (Mar)
- Attacking the Windows NVIDIA Driver (Feb)
- Lifting the (Hyper) Visor: Bypassing Samsung's Rea... (Feb)

2016

- Chrome OS exploit: one byte overflow and symlinks (Dec)
- BitJmmap: Attacking Android Ashmem (Dec)
- Breaking the Chain (Nov)
- task\_t considered harmful (Oct)
- Announcing the Project Zero Prize (Sep)
- Return to libstagefright: exploiting libutls on A... (Sep)
- A Shadow of our Former Self (Aug)
- A year of Windows kernel font fuzzing #2: the tech... (Jul)
- How to Compromise the Enterprise Endpoint (Jun)
- A year of Windows kernel font fuzzing #1: the results (Jun)
- Exploiting Recursion in the Linux Kernel (Jun)
- Life After the Isolated Heap (Mar)
- Race you to the kernel! (Mar)
- Exploiting a Leaked Thread Handle (Mar)
- The Definitive Guide on Win32 to NT Path Conversion (Feb)
- Racing MIDI messages in Chrome (Feb)
- Raising the Dead (Jan)

2015

- FireEye Exploitation: Project Zero's Vulnerability... (Dec)
- Between a Rock and a Hard Link (Dec)
- Windows Sandbox Attack Surface Analysis (Nov)
- Hack The Galaxy: Hunting Bugs in the Samsung Galax... (Nov)
- Windows Drivers are True'y Tricky (Oct)
- Revisiting Apple IPC: (1) Distributed Objects (Sep)
- Kaspersky: Mo Unpackers, Mo Problems. (Sep)
- Stagefrightened? (Sep)
- Enabling QR codes in Internet Explorer, or a story... (Sep)
- Windows 10/H4H Symbolic Link Mitigations (Aug)
- One font vulnerability to rule them all #4: Window... (Aug)
- Three bypasses and a fix for one of Flash&#39;s Vector... (Aug)
- Attacking ECMAScript Engines with Redefinition (Aug)
- One font vulnerability to rule them all #3: Window... (Aug)
- One font vulnerability to rule them all #2: Adobe... (Aug)
- One font vulnerability to rule them all #1: Introd... (Jul)
- One Perfect Bug: Exploiting Type Confusion in Flash (Jul)
- Significant Flash exploit mitigations are live in ... (Jul)
- From inter to intra: gaining reliability (Jul)
- When 'int' is the new 'short' (Jul)
- What is a <quot;good"quot; memory corruption vulnerability? (Jun)
- Analysis and Exploitation of an ESET Vulnerability (Jun)
- Owning Internet Printing - A Case Study in Modern ... (Jun)
- Dude, where's my heap? (Jun)
- In-Console-Able (May)
- A Tale of Two Exploits (Apr)
- Taming the wild copy: Parallel Thread Corruption (Mar)
- Exploiting the DRAM rowhammer bug to gain kernel p... (Mar)
- Feedback and data-driven updates to Google's discl... (Feb)
- (\"Exploiting\") CVE-2015-0318/s/\* (info)\* (Flash) (Feb)
- A Token's Tale (Feb)
- Exploiting NuMAP to escape the Chrome sandbox - CV... (Jan)
- Finding and exploiting ntpd vulnerabilities (Jan)

2014

- Internet Explorer EPM Sandbox Escape CVE-2014-6336 (Dec)
- pwn4fun Spring 2014 - Safari - Part II (Nov)
- Project Zero Patch Tuesday roundup, November 2014 (Nov)
- Did the "Man With No Name" Feel Insecure? (Oct)
- More Mac OS X and iPhone sandbox escapes and kerne... (Oct)
- Exploiting CVE-2014-0556 in Flash (Sep)
- The poisoned NUL byte, 2014 edition (Aug)
- What does a pointer look like, anyway? (Aug)
- Mac OS X and iPhone sandbox escapes (Jul)
- pwn4fun Spring 2014 - Safari - Part I (Jul)
- Announcing Project Zero (Jul)