Vulnerability Disclosure Policy

Version **v1.2** Date **2022-07-01**

Disclosure Policy

We believe that vulnerability disclosure is a two-way street. Vendors, as well as researchers, must act responsibly. This is why modzero adheres to a **90-day** disclosure deadline. We notify vendors of vulnerabilities immediately, with details shared in public with the defensive community after 90 days, or sooner if the vendor releases a fix. That deadline can vary in the following ways:

1) If a deadline is due to expire on a weekend, Swiss or German public holiday, the deadline will be moved to the next normal work day.
2) If the vendor does not respond to multiple contact attempts during the first 14 days after initial notification, we will disclose information about the vulnerabilities immediately.
3) Before the 90-day deadline has expired, if a vendor lets us know that a patch is scheduled for release on a specific day that will fall within 14 days following the deadline, we will delay the public disclosure until the availability of the patch.
   When we observe a previously unknown and unpatched vulnerability in software under active exploitation (a "0-day"/Zero-day), we believe that more urgent action—within 7 days—is appropriate. The reason for this special designation is that each day an actively exploited vulnerability remains undisclosed to the public and unpatched, more devices or accounts will be compromised. Seven days is an aggressive timeline and may be too short for some vendors to update their products, but it should be enough time to publish advice about possible mitigations, such as temporarily disabling a service, restricting access, or contacting the vendor for more information. As a result, after 7 days have elapsed without a patch or advisory, we will support researchers making details available so that users can take steps to protect themselves

As always, we reserve the right to bring deadlines forwards or backwards based on extreme circumstances. We remain committed to treating all vendors strictly equally. modzero expects to be held to the same standard.

This policy is strongly in line with our desire to improve industry response times to security bugs, but also results in softer landings for bugs marginally over deadline. Creating pressure towards more reasonably-timed fixes will result in smaller windows of opportunity for “blackhats” to abuse vulnerabilities. In our opinion, vulnerability disclosure policies such as ours result in greater overall safety for users of the Internet.

FAQ

This FAQ is based on Google's Project Zero Vulnerability Disclosure FAQ[1]. modzero essentially shares the same views and experiences while dealing with vendors. Therefore, most of the FAQ is taken verbatim from Project Zero.

What is modzero's 90-day disclosure deadline policy?

When modzero finds a new vulnerability, we send a detailed technical description of the issue to the relevant vendor or open-source project. This initial vulnerability report includes the following statement:

> *"This bug is subject to a 90-day disclosure deadline. After 90 days elapse or a patch has been made broadly available (whichever is earlier), the bug report will become visible to the public."*

Our expectation is that the developer will fix the security vulnerability within 90 days. modzero won't publicly discuss details about the vulnerability until the issue has been fixed, or until 90 days pass without a patch being made available to users, whichever is earlier.

What happens if a patch isn't broadly available after 90 days?

If the patch is expected to arrive within 14 days of the deadline expiring, then modzero can offer an extension. We implemented a 14-day grace extension after receiving some good feedback from other vendors. If the timing is awkward, for example, where a vendor's scheduled monthly patch release is due two days after the deadline expiry date, we may agree that a clearly defined grace extension is a reasonable compromise for this situation.

If we don't think a fix will be ready within 14 days, then we use the original 90-day deadline as the time of disclosure. That means we grant a 14-day grace extension when there's a commitment by the developer to ship a fix within the 14-day grace period.

[1] https://googleprojectzero.blogspot.com/p/vulnerability-disclosure-faq.html

When does modzero disclose a vulnerability after 14 days of initial notification?

In case the vendor is non-responsive to multiple attempts of contact, the vulnerability and its details will be made public after this initial 14-day deadline. We do this to urge the vendor to acknowledge the vulnerability and initiate a discussion.

How does modzero publicly disclose a vulnerability?

Initially, all of our bug reports are restricted so that only modzero employees can see the technical content. When it's time to disclose, the technical description of the vulnerability will become publicly accessible, for example, by publishing it on a webserver. If the disclosure happens because of a missed deadline, the "Deadline-Exceeded" label is used. If the 14-day grace extension was applied, the bug will have the "Deadline-Grace" label.

Why are disclosure deadlines necessary?

We were concerned that patches were taking a long time to be developed and released to users, and we felt that disclosure deadlines set up the right balance of incentives.

Software vendors have responded to disclosure deadlines in a way that other options were historically unable to accomplish. Prior to modzero our researchers had tried a number of different disclosure policies, such as coordinated vulnerability disclosure. Coordinated vulnerability disclosure is premised on the idea that any public disclosure prior to a fix being released unnecessarily exposes users to malicious attacks, and so the vendor should always set the time frame for disclosure.

We used this model of disclosure for over a decade, and the results weren't particularly compelling. Many fixes took over six months to be released, while some of our vulnerability reports went unfixed entirely. We were optimistic that vendors could do better, but we weren't seeing the improvements to internal triage, patch development, testing, and release processes that we knew would provide the most benefit to users.

But why do slow patch timelines matter? If you assume that only the vendor and the reporter have knowledge of the vulnerability, then the issue can be fixed without urgency. However, we increasingly have evidence that offensive attackers are finding (or acquiring) many of the same vulnerabilities that defensive security researchers are reporting.

We can't know for sure when a security bug we have reported has previously been found by an attacker (recent attempts to quantify the rate of bug collision can be found for

example at RAND[2] and at the Belfer Center[3]), but we know that it happens regularly enough to factor into our disclosure policy. We think that our policy introduces an appropriate level of urgency into the vulnerability remediation process.

Essentially, disclosure deadlines are a way for security researchers to set expectations and provide a clear incentive for vendors and open-source projects to improve their vulnerability remediation efforts. We tried to calibrate our disclosure timeframes to be ambitious, fair, and realistically achievable.

While every vulnerability disclosure policy has certain pros and cons, modzero has concluded that a 90-day disclosure deadline policy is currently the best option available for user security. Based on our experiences with using this policy, we can say that we're very satisfied with the results. No one at modzero is happy when a deadline is missed, but a consistent and fair approach to enforcing disclosure deadlines goes a long way.

Doesn't disclosing a vulnerability when there's no fix endanger users?

The answer is counterintuitive at first: disclosing a small number of unfixed vulnerabilities doesn't meaningfully increase or decrease attacker capability. Our "deadline-based" disclosures have a neutral short-term effect on attacker capability.

We certainly know that there are groups and individuals that are waiting to use public attacks to harm users (like exploit kit authors), but we also know that the cost of turning a typical vulnerability report into a practical real-world attack is non-trivial.

Since modzero typically discloses only one part of an exploit chain, attackers need to perform substantial additional research and development to complete the exploit and make it reliable. Any attacker with the resources and technical skills to turn a bug report into a reliable exploit chain would usually be able to build a similar exploit chain even if we had never disclosed the bug. They would either have the ability to find and exploit their own 0day vulnerabilities, or have access to a range of other interchangeable bugs (e. g. other fixed/disclosed bugs from the past weeks/months).

Also, the window of exposure between disclosure and a fix being released is very small, i.e., a patch usually arrives shortly after a deadline is missed, and the attacker's risk of detection increases rapidly from the point of disclosure.

[2] https://www.rand.org/pubs/research_reports/RR1751.html
[3] https://www.belfercenter.org/publication/taking-stock-estimating-vulnerability-rediscovery

For any attackers that are willing to exploit publicly disclosed bugs (despite the increased risk of failure or detection), there currently seems to be two alternative options that are preferred for their cost-effectiveness:

1) Waiting for disclosed bugs that require only a small amount of additional research and development (design flaws and logic bugs, or other easily exploitable conditions); or
2) Waiting for a fully developed and reliable exploit to be leaked (typically when a targeted exploit attempt using 0day is detected).

All of this means that there isn't a substantial difference between deadline enforced disclosures or our normal post-patch disclosure in terms of the observed rates of "opportunistic reuse" by attackers. If most bugs are fixed in a reasonable timeframe (i.e. less than 90 days), then we are only enforcing the deadline on a very small number of unfixed cases. And if disclosing a handful of unfixed vulnerabilities doesn't substantially help attackers in the short-term, but does lead to the demonstrated long term benefits of shortened patch timelines and more frequent patching cycles, then it would follow that a deadline-based disclosure policy is good for user security overall.

Why do you disclose technical details about a bug when it's fixed?

We think there's tremendous long-term benefit in publishing details about research methodologies and results. We use discussions about vulnerabilities and exploits to drive a pipeline of work on structural improvements to software and hardware security: attack surface reduction, exploit mitigations, improved sandboxing, fixing bug classes, and improving the state of public security research.

We're also big believers in the educational benefits of sharing results and insights, and we hope that our blog posts and technical reports can provide a pathway for new researchers to join the security community. Additionally, we want to share our insights and areas of focus with other security experts in order to drive attention towards important attack surfaces, and to encourage more researchers to share their own results.

Information about how a modern exploit works is extremely valuable, and increasingly there are incentives for offensive practitioners to withhold this information from other security researchers, developers, and the public. To counter this shift towards privately held attack research, we think that encouraging high-quality public research on modern attacks is a key part of building a better ecosystem of well-informed defenders.

Why do you release information about the vulnerability so quickly after a fix is released?

It's a tricky balance, but in essence we want to even the playing field.
Attackers have a clear incentive to spend time analyzing security patches in order to learn about vulnerabilities (both through source code review and binary reverse engineering), and they'll quickly establish the full details even if the vendor and researcher attempt to withhold technical data.

Since the utility of information about vulnerabilities is very different for defenders vs attackers, we don't expect that defenders can typically afford to do the same depth of analysis as attackers. The feedback that we get from defenders is that they want more information about the risks that they and their users face.

The information that we release can commonly be used by defenders to immediately improve defenses, testing the accuracy of bug fixes, and can always be used to make informed decisions about patch adoption or short-term mitigations.

Timely information also generates a level of momentum and excitement in the security research community. We aim to harness this to drive follow-up research and to motivate discussions about long-term structural improvements to security, an opportunity that would otherwise be lost. Overall, we think that prompt disclosure of details about fixed bugs favors defenders more than it favors attackers.

How do you decide who to report a vulnerability to?

We think that vulnerability reports should be communicated directly to the vendor or open-source project that is responsible for developing the fix. Generally, we use an official point of contact for security bug reports (e. g., an email address or issue tracker) and we follow each project's documented process for handling security bugs until a bug is fixed or a disclosure deadline has passed.

Sometimes we get asked to share our vulnerability reports with third parties, such as organizations that are affected by the vulnerability. By default, we decline these requests, and we generally ask that vendors refrain from sharing our vulnerability reports with third parties unnecessarily. Others have observed several unintended outcomes from vulnerability sharing under embargo arrangements, such as: increased risk of leaks, slower patch release cycles, and inconsistent criteria for inclusion.

Do you ever help software vendors or open-source projects fix the issues you report?

Absolutely! We want to be involved as much as possible in the patch development process, and encourage vendors to collaborate with our researchers to make sure patches are correct and complete. We often directly suggest a source code patch that will resolve the underlying bug, but for complex cases we will typically work with the software maintainer to develop and verify a correct fix.
modzero employees are always available to provide feedback during the patch development process—an extra pair of eyes on a security patch can make a big difference, so we encourage vendors to reach out to our researchers if they have any questions or ideas that they'd like to discuss further. There have been several occasions where the initial patch was incomplete or inadvertently introduced another vulnerability, and we've happily worked with the maintainer/vendor to come up with a correct fix.

We often include additional guidance about opportunities for code hardening, attack surface reduction, design improvements, testing and so on. This often results in structural improvements above-and-beyond an individual bug fix. Collaborating on these structural improvements is a specific goal for modzero, and is seen as an important long-term component of our work.

Would you recommend other security researchers use a disclosure deadline policy?

Yes, we'd encourage other security researchers to use disclosure deadlines as well.

We think that industry practices will improve as more researchers start to include timeline expectations in their bug reports. There are many good reasons why a security researcher might choose not to adopt a disclosure deadline policy on their bug reports, but overall, we've seen many positive outcomes from adopting disclosure deadlines and we can certainly recommend it to other security researchers.

We understand that some software vendors have chosen to prioritize these vulnerability reports at the expense of other vulnerability reports that don't have a specific disclosure timeline. As more security researchers apply deadlines, we're expecting software vendors to prioritize bug fixes based on overall impact and to invest appropriately to ensure that all important security issues can be fixed in a timely manner, and we think that would be a step in the right direction for user security.

What do you do if a vendor says a bug is invalid, or says that they cannot, or will not, fix it?

If we report an issue and the vendor indicates that they won't be issuing a patch, then we publish the technical details for discussion with a status of “wont fix” and include an additional technical assessment of the developer's response.

In essence we shift from treating the bug report as a vulnerability (where the rules of vulnerability disclosure apply) and instead begin to treat the issue as a non-security bug (where there are typically no restrictions on public discussion). We think this incentivizes vendors to perform high-quality triaging of our bug reports.

Software maintainers have been very good at assessing the security risk of the issues we report to them, and it's rare that modzero and a developer disagree about the severity of an issue.

So, is a publicly available source code patch a “fix” even if there's no build for it?

We think a public source code patch is usually equivalent to a public disclosure, even if it's not clearly marked as a security-relevant change. There's a good amount of research that supports this, such as Barth et al's *“How Open Should Open Source Be?”*[4] or Aubizzierre's *“Unearthing the World's Best Bugs”*[5]. We also have experience at modzero with analyzing security patches, so we have a good sense for what is technically feasible here, and we know that attackers have an incentive to perform this analysis against high-profile targets.

modzero researchers reported vulnerabilities in several different open-source projects, and we've noticed all projects handle security fixes in a slightly different way. Some prefer immediately releasing security patches as soon as they're ready, while others try for a more coordinated approach. Open-source projects and their user communities are in the best position to choose how to disseminate patches, but our view is that once a patch is public, we can start to discuss the vulnerability in more detail with the wider security community.

[4] https://arxiv.org/abs/1109.0507
[5] https://downloads.immunityinc.com/infiltrate-archives/infiltrate_miaubiz.pdf

Why does modzero release proof-of-concept exploit code? Doesn't this help attackers?

The primary argument against releasing proof-of-concept exploit code is that malicious parties can quickly repurpose our research into an attack that harms users. While this may occur when “full chain” exploits are released, in almost all cases our proof-of-concept code is not immediately repurposable for an attack—i.e., substantial additional research and development will be required before an exploit can be used in the wild. As discussed above *(“Doesn't disclosing a vulnerability when there's no fix endanger users?”*), releasing our proof-of-concept code doesn't appear to materially affect attacker capability.

On the flip side, we think there are some benefits to giving defenders concrete data on what an exploit might look like for any particular bug—it can assist network administrators in prioritizing patch deployment, it gives security experts the ability to validate, understand, mitigate and detect some attacks, and it provides public, real-world data to effectively drive the future of secure software development.

modzero has publicly announced the existence of a bug prior to the 90-day deadline in the past. Isn't this a type of disclosure that goes against your own policy?

From a business perspective, a disclosure at any level of detail can have a range of serious consequences. From a technical and user risk perspective however, the level of detail shared is important to factor in.

In most cases we don't think that announcing the existence of a vulnerability is equivalent to a detailed vulnerability disclosure. All software of sufficient complexity will contain vulnerabilities, so saying things like *"I just reported a vulnerability in the Android media server"* isn't materially useful information for an attacker. It's common that software vendors give early notification of upcoming advisories, and other security researchers have had good success with announcing high-level summaries of pending publications.

One concern we've heard from vendors about announcements like this: customers will often contact their software provider to inquire about the status of a fix or potential mitigations, and this can increase costs.
modzero doesn't currently announce the existence of pending vulnerability fixes, but we're keeping a close eye on how other researchers approach this, and we may experiment with early notifications again in the future if there's sufficient interest in this approach.

Are hardware vulnerabilities treated differently to software vulnerabilities in your disclosure policy?

For the time being, we intend to apply the same disclosure policy for both hardware and software issues. These cases are rare and often discussed at length, and we have historical precedence for enforcing disclosure deadlines on both hardware and software issues. Each of these discussions has been unique and valuable, and so we think it's too early to reset our expectations specifically for hardware vendors.

All of the systems that we research have different pre-existing constraints and capabilities, and we have observed legacy architectural and process issues that can make timely patch development incredibly challenging for hardware vendors. However, we don't think that resolving hardware security issues in a timely manner is impossible or infeasible, and instead, it appears that our disclosure policy has been effective at motivating increased investment in hardware security. Similar to our software vulnerability reporting, we're excited to see the results from our hardware vulnerability reporting over time.

Acknowledgments

modzero would like to thank **Google Application Security** and **Google Project Zero** for sharing thoughts, experiences, code and information on vulnerabilities. We appreciate the effort undertaken to compile the original FAQ as well as working out the Disclosure Policy which has become a de facto standard in the security research community.