Vulnerability Disclosure Cheat Sheet

Introduction

This cheat sheet is intended to provide guidance on the vulnerability disclosure process for both security researchers and organizations. This is an area where collaboration is extremely important, but that can often result in conflict between the two parties.

Researchers should:

- Ensure that any testing is legal and authorized.
- Respect the privacy of others.
- Make reasonable efforts to contact the security team of the organization.
- Provide sufficient details to allow the vulnerabilities to be verified and reproduced.
- Not demand payment or rewards for reporting vulnerabilities outside of an established bug bounty program.

Organizations should:

- Provide a clear method for researchers to securely report vulnerabilities.
- Clearly establish the scope and terms of any bug bounty programs.
- Respond to reports in a reasonable timeline.
- Communicate openly with researchers.
- Not threaten legal action against researchers.
- Request CVEs where appropriate.
- Publish clear security advisories and changelogs.
- Offer rewards and credit.

Methods of Disclosure

There are a number of different models that can be followed when disclosing vulnerabilities, which are listed in the sections below.

Private Disclosure

In the private disclosure model, the vulnerability is reported privately to the organization. The organization may choose to publish the details of the vulnerabilities, but this is done at the discretion of the organization, not the researcher, meaning that many vulnerabilities may never be made public. The majority of bug bounty programs require that the researcher follows this model.

The main problem with this model is that if the vendor is unresponsive, or decides not to fix the vulnerability, then the details may never be made public. Historically this has lead to researchers getting fed up with companies ignoring and trying to hide vulnerabilities, leading them to the full disclosure approach.

Full Disclosure

With the full disclosure approach, the full details of the vulnerability are made public as soon as they are identified. This means that the full details (sometimes including exploit code) are available to attackers, often before a patch is available. The full disclosure approach is primarily used in response or organizations ignoring reported vulnerabilities, in order to put pressure on them to develop and publish a fix.

This makes the full disclosure approach very controversial, and it is seen as irresponsible by many people. Generally it should only be considered as a last resort, when all other methods have failed, or when exploit code is already publicly available.

Responsible or Coordinated Disclosure

Responsible disclosure attempts to find a reasonable middle ground between these two approaches. With responsible disclosure, the initial report is made privately, but with the full details being published once a patch has been made available (sometimes with a delay to allow more time for the patches to be installed).

In many cases, the researcher also provides a deadline for the organization to respond to the report, or to provide a patch. If this deadline is not met, then the researcher may adopt the full disclosure approach, and publish the full details.

Google's Project Zero adopts a similar approach, where the full details of the vulnerability are published after 90 days regardless of whether or not the organization has published a patch.

Reporting Vulnerabilities

This section is intended to provide guidance for security researchers on how to report vulnerabilities to organizations.

Warnings and Legality

Before carrying out any security research or reporting vulnerabilities, **ensure that you know and understand the laws in your jurisdiction**. This cheat sheet **does not constitute legal advice, and should not be taken as such.**.

The following points highlight a number of areas that should be considered:

- If you are carrying out testing under a bug bounty or similar program, the organization may have established safe harbor policies, that allow you to legally carry out testing, **as long as you stay within the scope and rules of their program**. Make sure that you read the scope carefully - stepping outside of the scope and rules may be a criminal offense.
- Some countries have laws restricting reverse engineering, so testing against locally installed software may not be permitted.
- Do not demand payment or other rewards as a condition of providing information on security vulnerabilities, or in exchange for not publishing the details or reporting them to industry regulators, as this may constitute extortion.
- If you receive bug bounty payments, these are generally considered as income, meaning that they may be taxable. Reporting this income and ensuring that you pay the appropriate tax on it is **your** responsibility.
- If you find vulnerabilities as part of your work, or on equipment owned by your employer, your employer may prevent you from reporting these or claiming a bug bounty. Read your contract carefully and consider taking legal advice before doing so.

Finding Contact Details

The first step in reporting a vulnerability is finding the appropriate person to report it to. Although some organizations have clearly published disclosure policies, many do not, so it can be difficult to find the correct place to report the issue.

Where there is no clear disclosure policy, the following areas may provide contact details:

- Bug bounty programs such as BugCrowd, HackerOne, huntr.dev, Open Bug Bounty or Standoff.
- A security.txt file on the website at `/.well-known/security.txt` as per RFC 9116.
- An existing issue tracking system.
- Generic email addresses such as `security@` or `abuse@`.

- The generic "Contact Us" page on the website.
- Social media platforms.
- Phoning the organization.
- Reaching out to the community.

When reaching out to people who are not dedicated security contacts, request the details for a relevant member of staff, rather than disclosing the vulnerability details to whoever accepts the initial contact (especially over social media).

If it is not possible to contact the organization directly, a national or sector-based CERT may be able to assist.

Initial Report

Once a security contact has been identified, an initial report should be made of the details of the vulnerability. Ideally this should be done over an encrypted channel (such as the use of PGP keys), although many organizations do not support this.

The initial report should include:

- Sufficient details of the vulnerability to allow it to be understood and reproduced.
- HTTP requests and responses, HTML snippets, screenshots or any other supporting evidence.
  - Redact any personal data before reporting.
  - Some organizations may try and claim vulnerabilities never existed, so ensure you have sufficient evidence to prove that they did.
- Proof of concept code (if available).
- The impact of the vulnerability.
- Any references or further reading that may be appropriate.

In many cases, especially in smaller organizations, the security reports may be handled by developers or IT staff who do not have a security background. This means that they may not be familiar with many security concepts or terminology, so reports should be written in clear and simple terms.

It may also be beneficial to provide a recommendation on how the issue could be mitigated or resolved. However, unless the details of the system or application are known, or you are very confident in the recommendation then it may be better to point the developers to some more general guidance (such as an OWASP cheat sheet).

If you are planning to publish the details of the vulnerability after a period of time (as per some responsible disclosure policies), then this should be clearly communicated in the initial email - but try to do so in a tone that doesn't sound threatening to the recipient.

If the organization does not have an established bug bounty program, then avoid asking about payments or rewards in the initial contact - leave it until the issue has been acknowledged (or ideally fixed). In particular, **do not demand payment before revealing the details of the vulnerability**. At best this will look like an attempt to scam the company, at worst it may constitute extortion.

Ongoing Communication

While simpler vulnerabilities might be resolved solely from the initial report, in many cases there will be a number of emails back and forth between the researcher and the organization. Especially for more complex vulnerabilities, the developers or administrators may ask for additional information or recommendations on how to resolve the issue. They may also ask for assistance in retesting the issue once a fix has been implemented. Although there is no obligation to carry out this retesting, as long as the request is reasonable then and providing feedback on the fixes is very beneficial.

It may also be necessary to chase up the organization if they become unresponsive, or if the established deadline for publicly disclosing the vulnerability is approaching. Ensure that this communication stays professional and positive - if the disclosure process becomes hostile then neither party will benefit.

Be patient if it's taking a while for the issue to be resolved. The developers may be under significant pressure from different people within the organization, and may not be able to be fully open in their communication. Triaging, developing, reviewing, testing and deploying a fix within in an enterprise environment takes **significantly** more time than most researchers expect, and being constantly hassled for updates just adds another level of pressure on the developers.

When to Give Up

Despite every effort that you make, some organizations are not interested in security, are impossible to contact, or may be actively hostile to researchers disclosing vulnerabilities. In some cases they may even threaten to take legal action against researchers. When this happens it is very disheartening for the researcher - it is important not to take this personally. When this happens, there are a number of options that can be taken.

- Publicly disclose the vulnerability, and deal with any negative reaction and potentially even a lawsuit. Whether or not they have a strong legal case is irrelevant - they have expensive lawyers and fighting any kind of legal action is expensive and time consuming. Before going down this route, ask yourself *is it really worth it?*

- Anonymously disclose the vulnerability. However, if you've already made contact with the organization and tried to report the vulnerability to them, it may be pretty obvious who's responsible behind the disclosure. If you are going to take this approach, ensure that you have taken sufficient operational security measures to protect yourself.
- Report the vulnerability to a third party, such as an industry regulator or data protection authority.
- Move on.

There are many organizations who have a genuine interest in security, and are very open and co-operative with security researchers. Unless the vulnerability is extremely serious, **it is not worth burning yourself out, or risking your career and livelihood over an organization who doesn't care**.

Publishing

Once a vulnerability has been patched (or not), then a decision needs to be made about publishing the details. This should ideally be done through discussion with the vendor, and at a minimum the vendor should be notified that you intend to publish, and provided with a link to the published details. The disclosure would typically include:

- A high level summary of the vulnerability and its impact.
- Details of which version(s) are vulnerable, and which are fixed.
- Technical details or potentially proof of concept code.
- Mitigations or workarounds.
- Links to the vendor's published advisory.
- The timeline for the discovery, vendor communication and release.

Some organizations may request that you do not publish the details at all, or that you delay publication to allow more time to their users to install security patches. In the interest of maintaining a positive relationship with the organization, it is worth trying to find a compromise position on this.

Whether to publish working proof of concept (or functional exploit code) is a subject of debate. Some people will view this as an unethical move, and will argue that by doing so you are directly helping criminals compromise their users. On the other hand, the code can be used to both system administrators and penetration testers to test their systems, and attackers will be able to develop or reverse engineering working exploit code if the vulnerability is sufficiently valuable.

If you are publishing the details in hostile circumstances (such as an unresponsive organization, or after a stated period of time has elapsed) then you may face threats and even legal action. Whether there is any legal basis for this will depend on your jurisdiction, and whether you signed any form of non-disclosure agreement with the organization. Make sure you understand your legal position before doing so.

Note that many bug bounty programs forbid researchers from publishing the details without the agreement of the organization. If you choose to do so, you may forfeit the bounty or be banned from the platform - so read the rules of the program before publishing.

Receiving Vulnerability Reports

This section is intended to provide guidance for organizations on how to accept and receive vulnerability reports.

Bug Bounty Programs

Bug bounty programs incentivize researchers to identify and report vulnerabilities to organizations by offering rewards. These are usually monetary, but can also be physical items (swag). The process is often managed through a third party such as BugCrowd or HackerOne, who provide mediation between researchers and organizations.

When implementing a bug bounty program, the following areas need to be clearly defined:

- Which systems and applications are in scope.
  - Live systems or a staging/UAT environment?
  - Excluding systems managed or owned by third parties.
- Which types of vulnerabilities are eligible for bounties (SSL/TLS issues? Missing HTTP security headers? Version disclosure?)
- Legal provisions such as safe harbor policies.
  - The disclose.io project provides some example policies.
  - Take legal advice from lawyers, **not from this cheat sheet**.
- How much to offer for bounties, and how is the decision made.
  - The program could get very expensive if a large number of vulnerabilities are identified.
  - Too little and researchers may not bother with the program.
- The timeline for the initial response, confirmation, payout and issue resolution.

When to Implement a Bug Bounty Program

Bug bounty have been adopted by many large organizations such as Microsoft, and are starting to be used outside of the commercial sector, including the US Department of Defense. However, for smaller organizations they can bring significant challenges, and require a substantial investment of time and resources. These challenges can include:

- Having sufficient time and resources to respond to reports.
- Having sufficiently skilled staff to effectively triage reports.
  - Reports may include a large number of junk or false positives.
  - Managed bug bounty programs may help by performing initial triage (at a cost).
- Dealing with large numbers of false positives and junk reports.
- The impact of individuals testing live systems (including unskilled attackers running automated tools they don't understand).
- Being unable to differentiate between legitimate testing traffic and malicious attacks.
- Researchers going out of scope and testing systems that they shouldn't.
- The financial cost of running the program (some companies pay out hundreds of thousands of dollars a year in bounties).
- Dealing with researchers who are unhappy with how the program is run (such as disputing bounty amounts, or being angry when reported issues are duplicates or out of scope).

Despite these potential issues, bug bounty programs are a great way to identify vulnerabilities in applications and systems. However, they should only be used by organizations that already have a mature vulnerability disclosure process, supported by strong internal processes to resolve vulnerabilities.

Publishing Contact Details

The most important step in the process is providing a way for security researchers to contact your organization. The easier it is for them to do so, the more likely it is that you'll receive security reports. The following list includes some of the common mechanisms that are used for this - the more of these that you can implement the better:

- A dedicated security contact on the "Contact Us" page.
- Dedicated instructions for reporting security issues on a bug tracker.
- The common `security@` email address.
- A security.txt file on the website at `/.well-known/security.txt` as per RFC 9116.

- Using third party bug bounty program.

It is also important to ensure that frontline staff (such as those who monitor the main contact address, web chat and phone lines) are aware of how to handle reports of security issues, and who to escalate these reports to within the organization.

Providing Reporting Guidelines

Alongside the contact details, it is also good to provide some guidelines for researchers to follow when reporting vulnerabilities. These could include:

- Requesting specific information that may help in confirming and resolving the issue.
- Using specific categories or marking the issue as confidential on a bug tracker.
- Providing PGP keys for encrypted communication.
- Establishing a timeline for an initial response and triage.
- Establishing safe harbor provisions.

Communicating With Researchers

Communication between researchers and organizations is often one of the hardest points of the vulnerability disclosure process, and can easily leave both sides frustrated and unhappy with the process.

The outline below provides an example of the ideal communication process:

- Respond to the initial request for contact details with a clear mechanism for the researcher to provide additional information.
- Acknowledge the vulnerability details and provide a timeline to carry out triage.
- Request additional clarification or details if required.
- Confirm the vulnerability and provide a timeline for implementing a fix.
    - Confirm the details of any reward or bounty offered.
- If required, request the researcher to retest the vulnerability.
- Confirm that the vulnerability has been resolved.

Throughout the process, provide regular updates of the current status, and the expected timeline to triage and fix the vulnerability. Even if there is no firm timeline for these, the ongoing communication provides some reassurance that the vulnerability hasn't been forgotten about.

Researchers Demanding Payment

Some individuals may approach an organization claiming to have found a vulnerability, and demanding payment before sharing the details. Although these requests *may* be legitimate, in many cases they are simply scams.

One option is to request that they carry out the disclosure through a mediated bug bounty platform, which can provide a level of protection for both sides, as scammers are unlikely to be willing to use these platforms.

Disclosure

Commercial and Open Source Software

Once the vulnerability has been resolved (and retested), the details should be published in a security advisory for the software. It is important to remember that publishing the details of security issues **does not make the vendor look bad**. All software has security vulnerabilities, and demonstrating a clear and established process for handling and disclosing them gives far more confidence in the security of the software than trying to hide the issues.

At a minimum, the security advisory must contain:

- A high level summary of the vulnerability, including the impact.
- A clear list of vulnerable versions.
- A clear list of patch versions.
- Any caveats on when the software is vulnerable (for example, if only certain configurations are affected).
- Any workarounds or mitigation that can be implemented as a temporary fix.
- A CVE for the vulnerability.

Where possible it is also good to include:

- The timeline of the vulnerability disclosure process.
- Credit for the researcher who identified the vulnerability.
- Technical details of the vulnerability.
- IDS/IPS signatures or other indicators of compromise.

Security advisories should be easy for developers and system administrators to find. Common ways to publish them include:

- A dedicated "security" or "security advisories" page on the website.
- A security mailing list or forum.

- Linked from the main changelogs and release notes.

Some researchers may publish their own technical write ups of the vulnerability, which will usually include the full details required to exploit it (and sometimes even working exploit code). For more serious vulnerabilities, it may be sensible to ask the researcher to delay publishing the full details for a period of time (such as a week), in order to give system administrators more time to install the patches before exploit code is available. However, once the patch has been releases, attackers will be able to reverse engineer the vulnerability and develop their own exploit code, so there is limited value to delaying the full release.

Private Systems

For vulnerabilities in private systems, a decision needs to be made about whether the details should be published once the vulnerability has been resolved. Most bug bounty programs give organizations the option about whether to disclose the details once the issue has been resolved, although it is not typically required.

Publishing these details helps to demonstrate that the organization is taking proactive and transparent approach to security, but can also result in potentially embarrassing omissions and misconfigurations being made public. In the event of a future compromise or data breach, they could also potentially be used as evidence of a weak security culture within the organization. Additionally, they may expose technical details about internal, and could help attackers identify other similar issues. As such, this decision should be carefully evaluated, and it may be wise to take legal advice.

Rewarding Researchers

Where researchers have identified and reported vulnerabilities outside of a bug bounty program (essentially providing free security testing), and have acted professionally and helpfully throughout the vulnerability disclosure process, it is good to offer them some kind of reward to encourage this kind of positive interaction in future. If monetary rewards are not possible then a number of other options should be considered, such as:

- Discounts or credit for services or products offered by the organization.
- Virtual rewards (such as special in-game items, custom avatars, etc).
- T-shirts, stickers and other branded items (swag).
- Credit in a "hall of fame", or other similar acknowledgement.

Further Reading

- The CERT Guide to Coordinated Vulnerability Disclosure
- HackerOne's Vulnerability Disclosure Guidelines
- Disclose.io's Vulnerability Disclosure Terms