National Cyber Security Centre

a part of GCHQ

Vulnerability Disclosure **Toolkit**

Contents

Introduction

The NCSC's Vulnerability Disclosure Toolkit is for organisations of all sizes who want to learn more about implementing a vulnerability disclosure process.

It is not intended to be a comprehensive guide to creating and implementing a vulnerability disclosure process, but instead focuses on the essential components to get you started.

About vulnerability disclosure

Security vulnerabilities are discovered all the time and people want to be able to report them directly to the organisation responsible. These reports can provide you with valuable information that you can use to improve the security of your systems. It really is in your best interest to encourage vulnerability disclosure.

Having a clearly signposted reporting process demonstrates that your organisation takes security seriously. By providing a clear process, organisations can receive the information directly so the vulnerability can be addressed, and the risk of compromise reduced. This process also reduces the reputational damage of public disclosure by providing a way to report, and a defined policy of how the organisation will respond.

Once the organisation has decided on a course of action, they should tell the 'finder' (that is, the person who's reported the vulnerability) that the issue is being managed, so they know that no further action is required.

A vulnerability disclosure process should:

- enable the reporting of found vulnerabilities
- be clear, simple, and secure
- define how the organisation will respond

The international standard for vulnerability disclosure (ISO/IEC 29147:2018) defines the techniques and policies that can be used to receive vulnerability reports and publish remediation information. The ETSI Guide to Coordinated Vulnerability Disclosure (ETSI TR 103 838) provides additional guidance on the essential steps to implementing a vulnerability disclosure process. The NCSC designed this toolkit for organisations that currently don't have a disclosure process but are looking to create one.

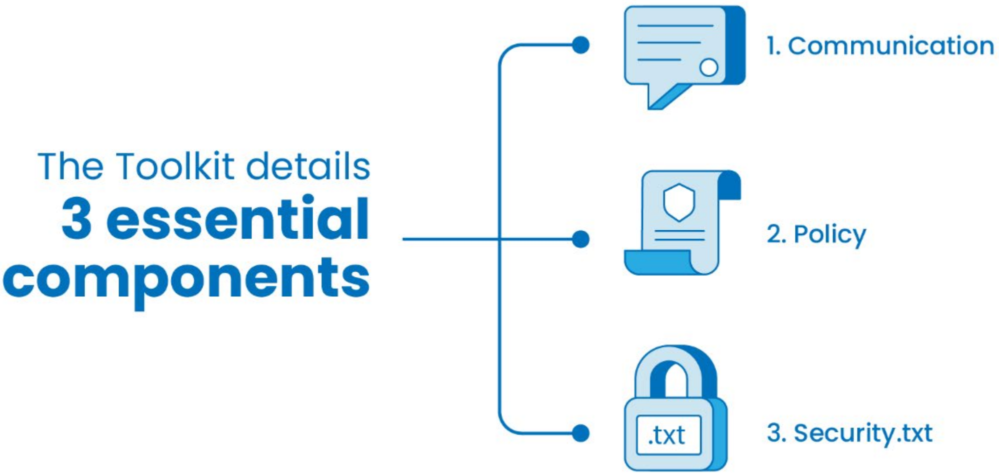
Why receive vulnerability reports?

Being able to receive, respond and ultimately fix a vulnerability report is essential to providing secure products and services. Receiving vulnerability reports helps mitigate two risks. Firstly, there is a risk that vulnerabilities are discovered by adversaries and exploited. By accepting and receiving vulnerability reports from finders, you help to reduce the number of vulnerabilities in your products or services. Secondly, if you do not provide a vulnerability disclosure route, finders who discover vulnerabilities might publicly disclose the information which will result in reputational damage, and once published could lead to a compromise. By having a predefined process, you can engage constructively with finders. This engagement means you can receive valuable information that would otherwise be missed (or require additional time and effort to discover).

Equally, going forward this requirement will be embedded into legislative frameworks. The UK government is currently developing legislation that will require manufacturers of smart devices to provide a public point of contact as part of a vulnerability disclosure policy. This is also a requirement for other international efforts on smart device security including the standard EN 303 645.

Toolkit components

This toolkit contains three components your organisation can implement to make it easier for you to create a vulnerability disclosure process.



1. Communication

Having a dedicated email address (such as security@example.com) or contact web form ensures that the vulnerability information gets to the right person (or team) who can help fix the issue. The NCSC recommend vulnerability information is protected, and one of the easiest ways to achieve this is to use a secure web form.

You should make sure this contact route is easy to find. You can add it to your 'Contact Us' web page (or privacy or security pages) and publish a security.txt. If you don't want to highlight it on a web page, then you should publish the contact route in a security.txt.

The secure web form should allow for anonymous vulnerability reports but should also allow the finder to provide contact details so the triage team can ask for more information and provide any updates to the disclosure. Each report should be automatically assigned a unique reference number to allow both finder and organisation to track the report.

2. Policy

By providing a clear policy, organisations define what they expect from someone reporting a vulnerability, as well as what they will do in response. This enables the organisation and the finder to confidently work within an agreed framework.

The ISO standard defines the minimum requirements for a vulnerability disclosure policy. In its basic form, a vulnerability disclosure policy should contain the following information:

- how you want to be contacted
- secure communication options (for example, a secure web form)
- what information to include in the report
- what the finder should expect to happen
- guidance on what is in and out of scope for the finder to do in finding vulnerabilities

If you are a UK government department, you can use our GitHub hosted vulnerability disclosure policy by linking directly to this policy on your website or in your security.txt file, or both.

An example of a basic vulnerability disclosure policy is included on Appendix 1.

3. Security.txt

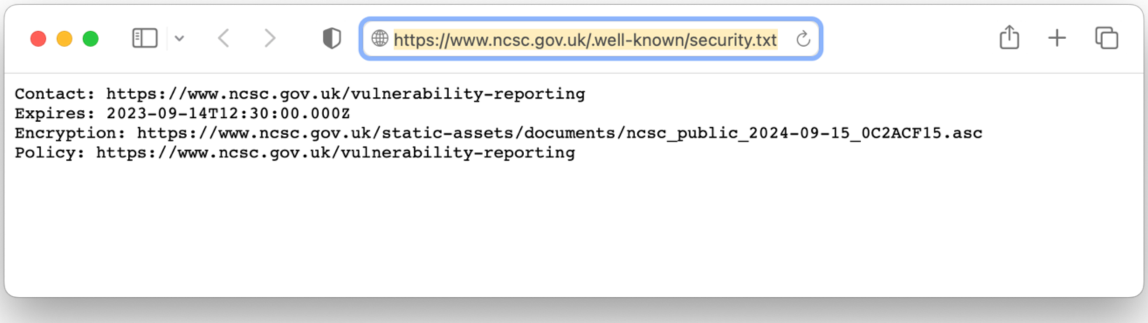
One of the most important elements of vulnerability disclosure, and a challenge for the finder, is understanding who to contact. Security.txt is an IETF Internet informational specification (RFC 9116) and it describes a text file that webmasters can host in the "/.well-known" directory of the domain root. It advertises the organisation's vulnerability disclosure process so that someone can quickly find all of the information needed to report a vulnerability.

The file contains three key fields:

- `CONTACT`: How finders should report vulnerabilities. For example, email or secure web form.
- `POLICY`: A link to the organisation's vulnerability disclosure policy.
- `EXPIRES`: Indicates the date and time after which the data contained in the "security.txt" file is considered stale and should not be used. The value of this field is formatted according to the Internet profile of [ISO.8601] as defined in [RFC3339]. It is recommended that the value of this field be less than a year into the future to avoid staleness.

The ENCRYPTION field is optional and should link to the PGP public key you wish to be used for encrypted communication.

You can see the NCSC's security.txt file at https://www.ncsc.gov.uk/.well-known/security.txt:



```
Contact: https://www.ncsc.gov.uk/vulnerability-reporting
Expires: 2023-09-14T12:30:00.000Z
Encryption: https://www.ncsc.gov.uk/static-assets/documents/ncsc_public_2024-09-15_0C2ACF15.asc
Policy: https://www.ncsc.gov.uk/vulnerability-reporting
```

The security.txt file should be published to all your domains and subdomains in the standard location of /.well-known/

For example:

```
# Only applies to example.com:
https://example.com/.well-known/security.txt


# Only applies to subdomain.example.com:
https://subdomain.example.com/.well-known/security.txt
```

We believe the IETF specification provides the best mix of ease of implementation and a simple way to advertise your vulnerability disclosure process. More information on security.txt and how to implement it can be found at https://securitytxt.org and in the IETF specification.

Implementation

Vulnerability disclosure report

The table below provides an example of the information that should be submitted in a vulnerability report. Implementing the report as a web form can enable you to set mandatory (M) fields for completion.

Report Section	Description
Title	Concise summary categorising the vulnerability, and the site/application where it can be found E.g. Reflected XSS on the XYZ website
Asset (M)	Web address, IP address, product, service name, etc.
Weakness	Such as a CWE
Severity	Such as low, medium, high, critical, and the calculated via CVSS
Description of the Vulnerability (M)	• A summary of the vulnerability • Supporting files (e.g. screenshot or video) • Any mitigations or recommendations
Steps to reproduce (M)	• Clear and descriptive steps to reproduce the vulnerability • Proof of concept code if available
Impact	The effects of successfully exploiting the vulnerability.
Contact details	• Name • Email Address (These details are optional to enable anonymous reporting)

(M) denotes mandatory field

How to respond to vulnerability disclosure

An unsolicited message from someone telling you about a vulnerability may sound scary at first, but don't panic. You are **not** under attack, and this is **not** a cyber incident. However, you should remember to follow the NCSC's Phishing Guidance. To effectively respond to a vulnerability report, take the following steps.

1. Don't ignore the report. Respond promptly to the finder and thank them. Feedback encourages engagement and they'll be more inclined to help you again in the future.
2. Pass the report to someone in your organisation who is responsible for the affected product or service. If it is managed by a third party, discuss the report with them.
3. Avoid forcing the finder to sign documents such as non-disclosure agreement as the individual is simply looking to ensure the vulnerability is fixed.
4. If you need more information to confirm and fix the issue, you should politely request that additional information from the finder.
5. Once you have decided on a course of action, let the finder know that the issue is being managed. You don't need to provide lots of technical information or commit to timescales.
6. If the issue takes time to fix, you should send periodic updates back to the finder.
7. Once the issue is fixed, let the finder know. They might be able to retest the issue to confirm the fix.
8. Consider publicly acknowledging and thanking the finder as this creates a sense of trust and transparency.

Validation and Triage

Validation

A vulnerability report is a documented account that describes a set of conditions exposing a vulnerability which can then be acted upon within the system. The report might not be fully complete or accurate so therefore it should always be checked and validated as the first part of the process. The validation step should follow the steps to reproduce and/or proof of concept provided in the vulnerability report. If you are unable to reproduce and validate the vulnerability, you should contact the finder to seek further clarification. They may be able to provide additional information. If, after validation, the report is assessed to not be a vulnerability, the finder should be informed.

Triage

It is necessary to triage all vulnerabilities using a consistent methodology. This should include checking the severity and impact assessment provided in the vulnerability report. Different triage methodologies can be used to come up with a consistent rating. One example is the Common Vulnerability Scoring System (CVSS) can be used as one example.

Report lifecycle

Once a report has been triaged it's a key point in the process and this is often a good point to communicate this back to the finder. A report will go through a few different states during a typical lifecycle, and these are outlined in the next section.

The table below illustrates the typical lifecycle of a vulnerability report. The changing report state will describe where it is in the lifecycle process. It is useful to assign each vulnerability report a unique reference number.

Report State	Description
Open: New	The report has not yet been validated and triaged. As part of the initial validation check there may be a need to ask the finder for additional information
Open: Triaged	The report has been triaged and assessed as a valid vulnerability. It would normally then be assigned to a person/team for remediation
Closed	There are a number of reasons why the report may be closed: • N/A: For example, the report could be spam; not a security vulnerability; or out of scope. • Duplicate: The vulnerability has already been reported. • Informative: The report contained useful security information but does not warrant remediation e.g. could be a configuration issue or is not going to be fixed. • Remediated: The vulnerability has been fixed.

Recognising finder contributions

When the vulnerability has been remediated, the organisation can consider acknowledging the finder for their work. This could be implemented through a thank you web page with a list of finders who have contributed reports. Alternatives are a letter of appreciation thanking them for their contribution and possibly the inclusion of merchandise.

Before public recognition, the finder should be consulted to ensure they agree with being acknowledged and agree what contact details should be shared.

When reports have been remediated you can inform the finder that the report has been closed with a simple preformatted standard reply.

Example:

```
Hello and thank you very much for using our vulnerability reporting service. We hope it
was easy to use but please do give any feedback you may have.

We believe this issue has now been resolved.

Many thanks again.
```

Response plans

From our experience of running the NCSC Vulnerability Reporting Service, two of the most frequent submissions are concerning cross-site scripting (XSS) vulnerabilities and subdomain takeover (otherwise known as dangling domains). In order to help in response to these two classes of vulnerabilities, the toolkit contains response plans for both.

Response plan: cross-site scripting vulnerability

Reflected cross-site scripting (XSS) is typically used to launch site impersonation or phishing attacks, in which unsuspecting users are lured to malicious sites via links that appear legitimate. The attacker is then free to present the user with what appears to be genuine content, in an attempt, for example, to capture authentication credentials.

This type of vulnerability occurs when data provided by a web client is used immediately by server-side scripts to generate a page of results for the user. If user-supplied data is included in the resulting page without proper HTML escaping or validation, client-side executable code may be injected into the dynamic page. This in turn, allows an attacker to craft a URL containing arbitrary JavaScript that, when accessed by a user, will result in said JavaScript code to be executed against the user.

In general, an attacker may exploit cross-site scripting to steal a victim's session tokens, log their keystrokes, steal private data, or perform privileged actions in the context of a victim's session. As an example, cross-site scripting could be exploited to perform an extremely effective phishing campaign - the attacker's malicious script will run in the same page as trusted content from the application, making it likely for users to enter sensitive credentials or data.

Recommendations

This issue will require changes to the website content. You should perform both context specific input validation and output encoding to all user-controlled content throughout the site to help mitigate cross-site scripting attacks.

Input validation

Only allow the characters you expect for the type of input you're receiving. For example:

- if you expect an unsigned integer, only accept unsigned integers within the expected range
- if you expect the names of UK counties, make sure the input is from a "known-good" list
- input which fails validation should be rejected, and not sanitised

Output encoding

When including user-submitted data in responses to end users, encode the output based on the appropriate context of where the output appears in the resulting page. Content placed into HTML needs to be HTML-encoded. HTML encoding functions should be sure to encode the following characters: single (') and double quotes ("), backticks (`), angle brackets (<, >), forward (/) and backslashes (\), equals signs (=), and ampersands (&).

More information is available in the OWASP XSS Prevention Cheat Sheet:

https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.md

Response plan: subdomain takeover vulnerability

A ‘subdomain takeover’ or ‘dangling domain’ vulnerability is where someone other than the system owner can take ownership of a subdomain.

This occurs when a subdomain is configured to display content hosted on a third-party site. This is indicated by the existence of a CNAME (Canonical Name) DNS record.

A Canonical Name, or CNAME record is a type of DNS record that maps an alias name to a true, (or canonical), domain name. CNAME records are typically used to map a subdomain such as www or mail to a third-party hosting provider, hosting that subdomain’s content. This third party is typically a cloud service provider such as AWS or Azure.

The issue occurs when the third-party subdomain appears to have expired or is otherwise no longer pointing to the intended content. This means that an attacker could create an account with the service provider, register the expired subdomain and point it to content that they control.

As the CNAME record pointing to the cloud subdomain is still present, an attacker can create content that appears to be hosted on the subdomain.

For example:

```
beta.example.com.  86400   IN  CNAME   beta-apps.cloudprovider.net.
```

The provision of beta-apps.cloudprovider.net has expired. Someone can therefore register and provision this. They can then add their own content to beta-apps.cloudprovider.net.

As beta.example.com points to beta-apps.cloudprovider.net, they appear to have taken over the subdomain.

Recommendation

If the subdomain is no longer needed, or no longer needs to be hosted by the cloud service provider, the CNAME record can safely be removed (likely routing requests to the subdomain to a 404 page not found error).

If this subdomain still needs to be hosted on the same subdomain, reclaim the subdomain from the finder. Verify that no other CNAMEs point to unclaimed or expired apps, or other hosting providers.

Appendix 1: Example of a basic vulnerability disclosure policy

You can use this policy as a basis for your organisational policy.

```
# Introduction

(Add an introduction to your organisation.)

This vulnerability disclosure policy applies to any vulnerabilities you are considering
reporting to us (the "Organisation").  We recommend reading this vulnerability
disclosure policy fully before you report a vulnerability and always acting in
compliance with it.

We value those who take the time and effort to report security vulnerabilities
according to this policy. However, we do not offer monetary rewards for vulnerability
disclosures.


# Reporting

If you believe you have found a security vulnerability, please submit your report to us
using the following link/email:

[insert link/email]

In your report please include details of:

* The website, IP or page where the vulnerability can be observed.

* A brief description of the type of vulnerability, for example; “XSS vulnerability”.

* Steps to reproduce. These should be a benign, non-destructive, proof of concept. This
helps to ensure that the report can be triaged quickly and accurately. It also reduces
the likelihood of duplicate reports, or malicious exploitation of some vulnerabilities,
such as sub-domain takeovers.


# What to expect

After you have submitted your report, we will respond to your report within 5 working
days and aim to triage your report within 10 working days. We’ll also aim to keep you
informed of our progress.

Priority for remediation is assessed by looking at the impact, severity and exploit
complexity. Vulnerability reports might take some time to triage or address. You are
welcome to enquire on the status but should avoid doing so more than once every 14
days. This allows our teams to focus on the remediation.

We will notify you when the reported vulnerability is remediated, and you may be
invited to confirm that the solution covers the vulnerability adequately.

Once your vulnerability has been resolved, we welcome requests to disclose your report.
We’d like to unify guidance to affected users, so please do continue to coordinate
public release with us.


# Guidance

You must NOT:

* Break any applicable law or regulations.

* Access unnecessary, excessive or significant amounts of data.

* Modify data in the Organisation's systems or services.

* Use high-intensity invasive or destructive scanning tools to find vulnerabilities.

* Attempt or report any form of denial of service, e.g. overwhelming a service with a
high volume of requests.

* Disrupt the Organisation's services or systems.

* Submit reports detailing non-exploitable vulnerabilities, or reports indicating that
the services do not fully align with “best practice”, for example missing security
headers.
```

```
* Submit reports detailing TLS configuration weaknesses, for example “weak” cipher
suite support or the presence of TLS1.0 support.

* Communicate any vulnerabilities or associated details other than by means described
in the published security.txt.

* Social engineer, ‘phish’ or physically attack the Organisation's staff or
infrastructure.

* Demand financial compensation in order to disclose any vulnerabilities.


You must:

* Always comply with data protection rules and must not violate the privacy of the
Organisation’s users, staff, contractors, services or systems. You must not, for
example, share, redistribute or fail to properly secure data retrieved from the systems
or services.

* Securely delete all data retrieved during your research as soon as it is no longer
required or within 1 month of the vulnerability being resolved, whichever occurs first
(or as otherwise required by data protection law).


# Legalities

This policy is designed to be compatible with common vulnerability disclosure good
practice. It does not give you permission to act in any manner that is inconsistent
with the law, or which might cause the Organisation or partner organisations to be in
breach of any legal obligations.
```