inesc id lisboa
Instituto de Engenharia de Sistemas e Computadores Investigação e Desenvolvimento em Lisboa

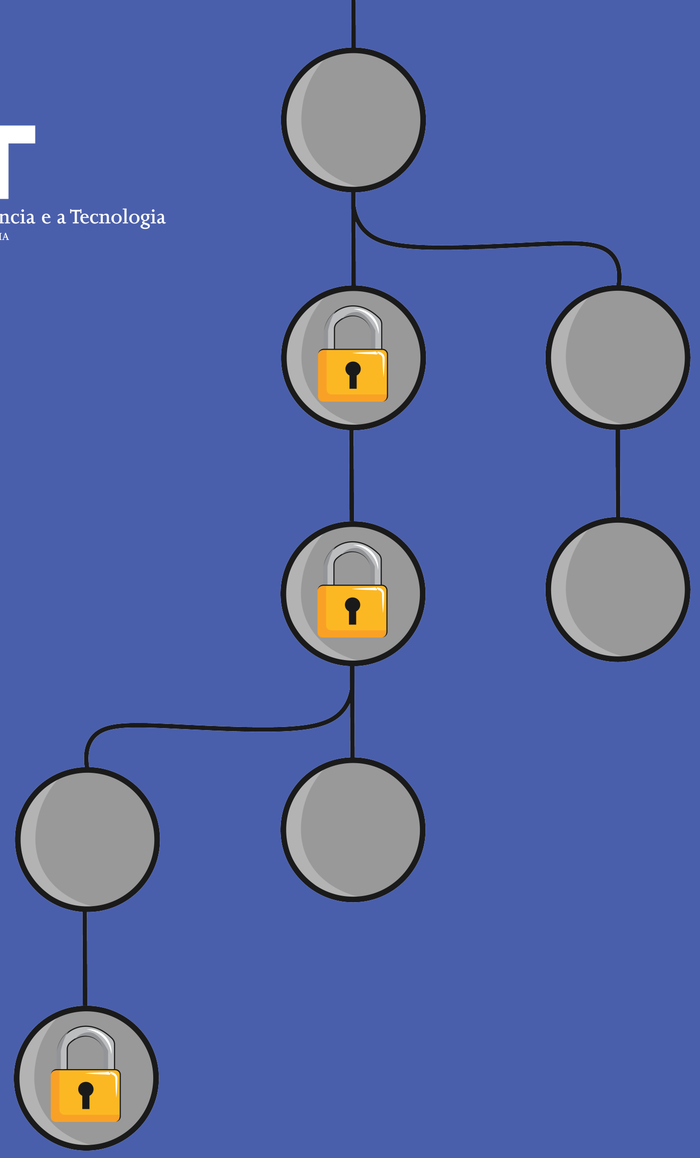
TÉCNICO LISBOA

Carnegie Mellon Portugal

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Carnegie Mellon University

FCT
Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA EDUCAÇÃO E CIÊNCIA



SECOM

Towards a convention for security commit messages

Sofia Reis, Rui Abreu, Hakan Erdogmus and Corina Păsăreanu

International Conference on Mining Software Repositories (MSR'22) | May 20, 2022

Background

💀 Detecting and assessing software vulnerabilities in machine learning is still a challenge mainly due do the scarcity and poor quality of curated data.

🏅 The chase for a gold standard dataset

😵‍💫 The detection of software vulnerabilities through commit messages is still challenging due to the **lack of structured** and **clear messages**.

⚙️

Static Analysis (Source code)	D2A [ICSE-SEIP'21]
Fuzzers (Source code)	FuzzBench [ESEC/FSE'21]
Regex / Keywords (Commit Messages)	SecBench [SecSE'17] Devign [NeurIPS'19]

✅

Verified Sources (NVD & CVE-Details)	Big-Vul [MSR'20] CVEFixes [PROMISE'21] SySeVR [TDSC'21]

👀

Manually	Pontas et. al [MSR'19]

Have you ever wondered if security commit messages are informative?

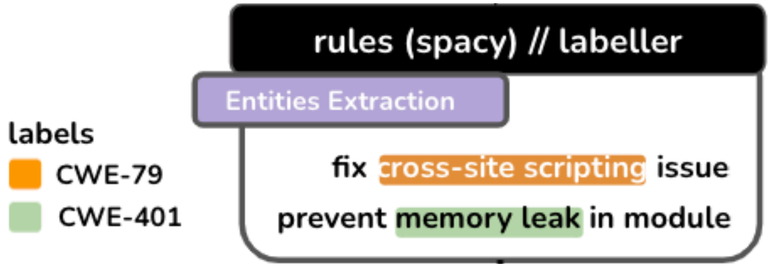
💭 *“Security experts/developers mainly provide cryptic and poor commit messages.”*

🧐 We built a rule-based tool to extract information from security commit messages.

🚀 These rules were improved based on the ground-truth.

🤔 From 2297 messages analysed, **22.81% were not classified by our rules**.

📦 Data was collected from GitHub commits found in the references of CVE reports.



* https://spacy.io/

Problem

Cryptic/poorly documented

Problem

Cryptic/poorly documented

Unclear how it relates with security

🤔 Some commit messages used to patch known vulnerabilities are either poorly documented or do not seem to be security-related.

Problem

Cryptic/poorly documented

Unclear how it relates with security

🤔 Some commit messages used to patch know vulnerabilities are either poorly documented or do not seem to be security-related.

💡 *Hooray!! We need best practices and templates to create better security commit messages.*

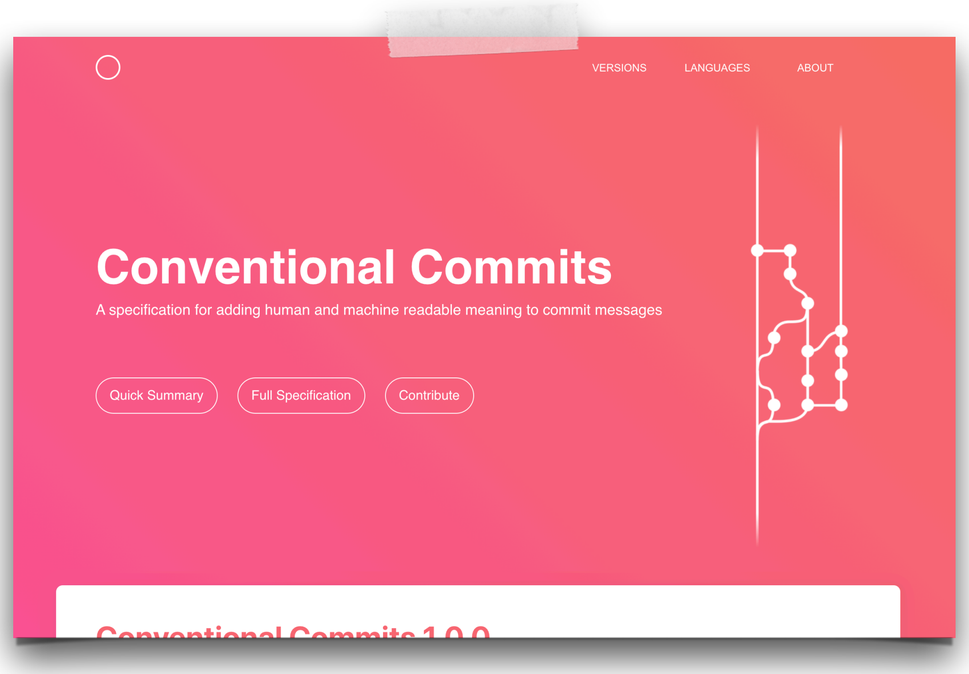
How to write a good security commit message? Any sources/guidelines available? Well, for commits, yes, but for security commits, no!

Conventional Commits 1.0.0

4. *types* other than `fix:` and `feat:` are allowed, for example @commitlint/config-conventional (based on the the Angular convention) recommends `build:`, `chore:`, `ci:`, `docs:`, `style:`, `refactor:`, `perf:`, `test:`, and others.

The Conventional Commits source suggests adding a type to the subject/header like fix: or feat:

e.g., fix: cross-site vulnerability

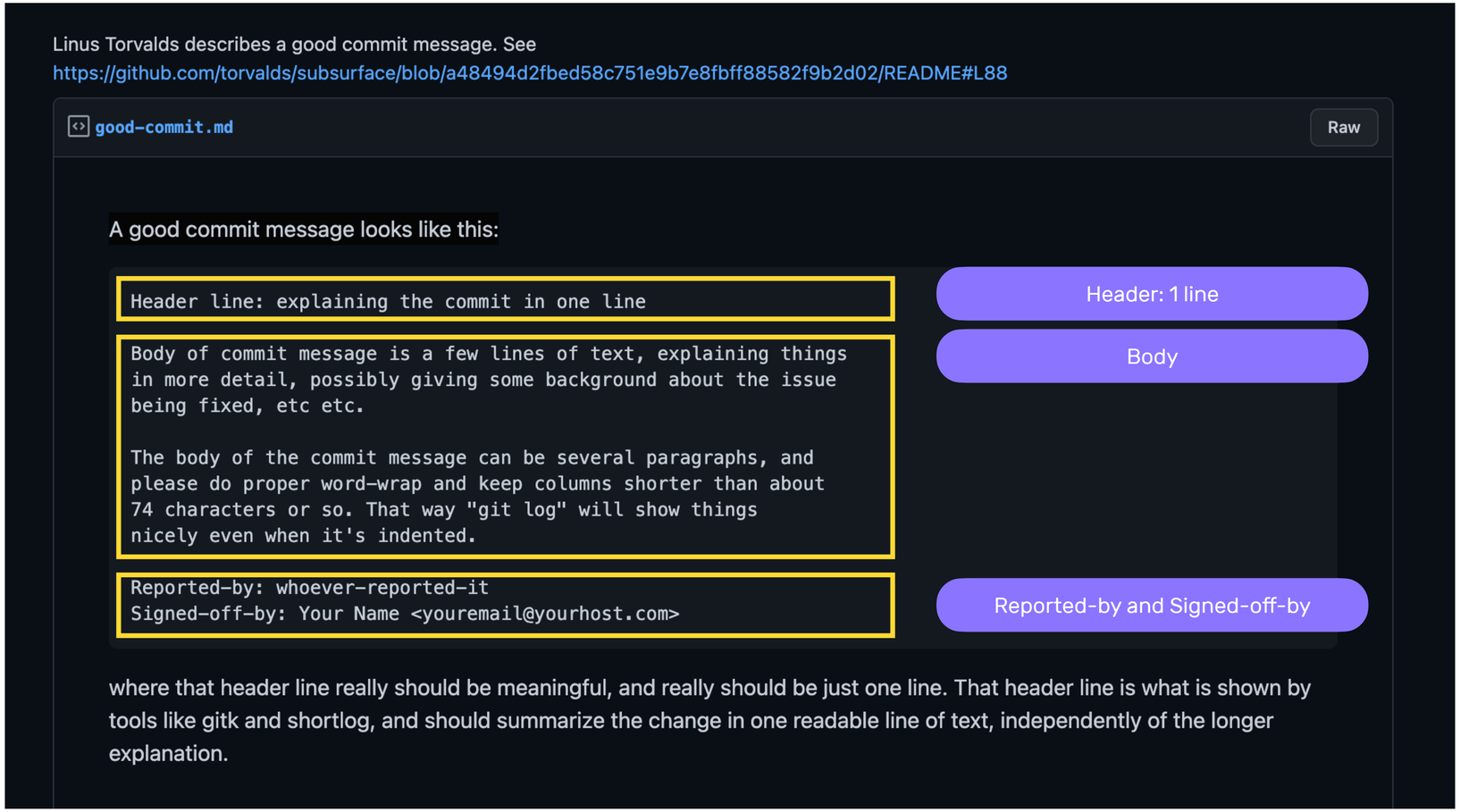


🔗 https://www.conventionalcommits.org/en/v1.0.0/

@sofiaoreis

How to write a good security commit message? Any sources/guidelines available? Well, for commits, yes, but for security commits, no!

“A good commit message looks like this” by Linus Torvalds



🔗 https://gist.github.com/matthewhudson/1475276

@sofiaoreis

How to write a good security commit message? Any sources/guidelines available? Well, for commits, yes, but for security commits, no!

"How to Write a Git Commit Message" by Chris Beams

```
Summarize changes in around 50 characters or less

More detailed explanatory text, if necessary. Wrap it to about 72
characters or so. In some contexts, the first line is treated as the
subject of the commit and the rest of the text as the body. The
blank line separating the summary from the body is critical (unless
you omit the body entirely); various tools like `log`, `shortlog`
and `rebase` can get confused if you run the two together.

Explain the problem that this commit is solving. Focus on why you
are making this change as opposed to how (the code explains that).
Are there side effects or other unintuitive consequences of this
change? Here's the place to explain them.

Further paragraphs come after blank lines.

 - Bullet points are okay, too

 - Typically a hyphen or asterisk is used for the bullet, preceded
   by a single space, with blank lines in between, but conventions
   vary here

If you use an issue tracker, put references to them at the bottom,
like this:

Resolves: #123
See also: #456, #789
```

Keep commits atomic (1 task -> 1 fix)

Subject

- 50 chars (max)
- Capitalized
- No period in the end
- Imperative form

Body

- 72 chars
- What, why and how

Bug-trackers // PRs & Issues mentions

🔗 https://chris.beams.io/posts/git-commit/

@sofiaoreis

NEW CONVENTION > SECOM

Atomic changes: Commit each patch as a separate change. — *ref. Chris Beams*

```
vuln-fix: subject/header containing summary of changes in ~50 characters (Vuln-ID,)

Detailed explanation of the subject/header in ~75 words.
(what) Explain the security issue(s) that this commit is patching.
(why) Focus on why this patch is important and its impact.
(how) Describe how the issue is patched.

[For Each Weakness in Weaknesses:]
Weakness: weakness identification or CWE-ID.
Severity: severity of the issue (Low, Medium, High, Critical).
CVSS: numerical representation (0-10) of the vulnerability severity.
Detection: method used to detect the issue (Tool, Manual, Exploit).
Report: http://link-to-report/
Introduced in: commit hash.
[End]

Reported-by: reporter name 1 <reporter-email-1@host.com>
Reported-by: reporter name 2 <reporter-email-2@host.com>
Signed-off-by: your name <your-email@yourhost.com>

[If you use an issue tracker, add reference to it here:]
[if external issue tracker:]
Bug-tracker: https://link-to-bug-tracker/id

[if github used as issue tracker:]
Resolves: #123
See also: #456, #789
```

🔗 https://tqrg.github.io/secom/

@sofiaoreis

NEW CONVENTION > SECOM

Header should start with a **type** — following the Conventional Commits specification.

⚡ The default type was set to **vuln-fix** upon discussions with the OSV team.

```
vuln-fix: subject/header containing summary of changes in ~50 characters (Vuln-ID,)

Detailed explanation of the subject/header in ~75 words.
(what) Explain the security issue(s) that this commit is patching.
(why) Focus on why this patch is important and its impact.
(how) Describe how the issue is patched.

[For Each Weakness in Weaknesses:]
Weakness: weakness identification or CWE-ID.
Severity: severity of the issue (Low, Medium, High, Critical).
CVSS: numerical representation (0-10) of the vulnerability severity.
Detection: method used to detect the issue (Tool, Manual, Exploit).
Report: http://link-to-report/
Introduced in: commit hash.
[End]

Reported-by: reporter name 1 <reporter-email-1@host.com>
Reported-by: reporter name 2 <reporter-email-2@host.com>
Signed-off-by: your name <your-email@yourhost.com>

[If you use an issue tracker, add reference to it here:]
[if external issue tracker:]
Bug-tracker: https://link-to-bug-tracker/id

[if github used as issue tracker:]
Resolves: #123
See also: #456, #789
```

🔗 https://tqrg.github.io/secom/

@sofiaoreis

NEW CONVENTION > SECOM

Subject should contain a brief summary of the patch. — *ref. Chris Beams and Linus Torvalds.*

🧐 We observed the inclusion of the **Vuln-ID** on the subject/header in some security commits.

💡 This pattern makes it easier to locate the patch when using shortlog/git log —pretty=oneline.

```
vuln-fix: subject/header containing summary of changes in ~50 characters (Vuln-ID,)

Detailed explanation of the subject/header in ~75 words.
(what) Explain the security issue(s) that this commit is patching.
(why) Focus on why this patch is important and its impact.
(how) Describe how the issue is patched.

[For Each Weakness in Weaknesses:]
Weakness: weakness identification or CWE-ID.
Severity: severity of the issue (Low, Medium, High, Critical).
CVSS: numerical representation (0-10) of the vulnerability severity.
Detection: method used to detect the issue (Tool, Manual, Exploit).
Report: http://link-to-report/
Introduced in: commit hash.
[End]

Reported-by: reporter name 1 <reporter-email-1@host.com>
Reported-by: reporter name 2 <reporter-email-2@host.com>
Signed-off-by: your name <your-email@yourhost.com>

[If you use an issue tracker, add reference to it here:]
[if external issue tracker:]
Bug-tracker: https://link-to-bug-tracker/id

[if github used as issue tracker:]
Resolves: #123
See also: #456, #789
```

NEW CONVENTION > SECOM

The **Body** should clearly explain the header in ~75 words; and, follow the *what*, *why* and *how* structure from Chris Beams. — *ref. Chris Beams and Linus Torvalds*

```
1   vuln-fix: subject/header containing summary of changes in ~50 characters (Vuln-ID,)
2
3   Detailed explanation of the subject/header in ~75 words.
4   (what) Explain the security issue(s) that this commit is patching.
5   (why) Focus on why this patch is important and its impact.
6   (how) Describe how the issue is patched.
7
8   [For Each Weakness in Weaknesses:]
9   Weakness: weakness identification or CWE-ID.
10  Severity: severity of the issue (Low, Medium, High, Critical).
11  CVSS: numerical representation (0-10) of the vulnerability severity.
12  Detection: method used to detect the issue (Tool, Manual, Exploit).
13  Report: http://link-to-report/
14  Introduced in: commit hash.
15  [End]
16
17  Reported-by: reporter name 1 <reporter-email-1@host.com>
18  Reported-by: reporter name 2 <reporter-email-2@host.com>
19  Signed-off-by: your name <your-email@yourhost.com>
20
21  [If you use an issue tracker, add reference to it here:]
22  [if external issue tracker:]
23  Bug-tracker: https://link-to-bug-tracker/id
24
25  [if github used as issue tracker:]
26  Resolves: #123
27  See also: #456, #789
```

NEW CONVENTION > SECOM

As seen on several reports, sometimes a CVE integrates different types of weaknesses. Therefore, SECOM also considers that multiple CWEs can be fixed by changes in the same piece of code.

```
1   vuln-fix: subject/header containing summary of changes in ~50 characters (Vuln-ID,)
2
3   Detailed explanation of the subject/header in ~75 words.
4   (what) Explain the security issue(s) that this commit is patching.
5   (why) Focus on why this patch is important and its impact.
6   (how) Describe how the issue is patched.
7
8   [For Each Weakness in Weaknesses:]
9   Weakness: weakness identification or CWE-ID.
10  Severity: severity of the issue (Low, Medium, High, Critical).
11  CVSS: numerical representation (0-10) of the vulnerability severity.
12  Detection: method used to detect the issue (Tool, Manual, Exploit).
13  Report: http://link-to-report/
14  Introduced in: commit hash.
15  [End]
16
17  Reported-by: reporter name 1 <reporter-email-1@host.com>
18  Reported-by: reporter name 2 <reporter-email-2@host.com>
19  Signed-off-by: your name <your-email@yourhost.com>
20
21  [If you use an issue tracker, add reference to it here:]
22  [if external issue tracker:]
23  Bug-tracker: https://link-to-bug-tracker/id
24
25  [if github used as issue tracker:]
26  Resolves: #123
27  See also: #456, #789
```

NEW CONVENTION > SECOM

For each CWE, the message should include the weakness metadata such as the weakness id or name, severity, CVSS, detection method, and, finally, a link to the report.

👀 Many of these fields were observed in some of the evaluated security commit messages.

```
vuln-fix: subject/header containing summary of changes in ~50 characters (Vuln-ID,)

Detailed explanation of the subject/header in ~75 words.
(what) Explain the security issue(s) that this commit is patching.
(why) Focus on why this patch is important and its impact.
(how) Describe how the issue is patched.

[For Each Weakness in Weaknesses:]
Weakness: weakness identification or CWE-ID.
Severity: severity of the issue (Low, Medium, High, Critical).
CVSS: numerical representation (0-10) of the vulnerability severity.
Detection: method used to detect the issue (Tool, Manual, Exploit).
Report: http://link-to-report/
Introduced in: commit hash.
[End]

Reported-by: reporter name 1 <reporter-email-1@host.com>
Reported-by: reporter name 2 <reporter-email-2@host.com>
Signed-off-by: your name <your-email@yourhost.com>

[If you use an issue tracker, add reference to it here:]
[if external issue tracker:]
Bug-tracker: https://link-to-bug-tracker/id

[if github used as issue tracker:]
Resolves: #123
See also: #456, #789
```

🔗 https://tqrg.github.io/secom/

@sofiaoreis

NEW CONVENTION > SECOM

For each CWE, the message should include the weakness metadata such as the weakness id or name, severity, CVSS, detection method, and, finally, a link to the report.

👀 Many of these fields were observed in some of the evaluated security commit messages.

When possible, the hash of the commit that introduced the vulnerability should be included in the message. — OSV team request to comply with their data schema.

```
1   vuln-fix: subject/header containing summary of changes in ~50 characters (Vuln-ID,)
2
3   Detailed explanation of the subject/header in ~75 words.
4   (what) Explain the security issue(s) that this commit is patching.
5   (why) Focus on why this patch is important and its impact.
6   (how) Describe how the issue is patched.
7
8   [For Each Weakness in Weaknesses:]
9   Weakness: weakness identification or CWE-ID.
10  Severity: severity of the issue (Low, Medium, High, Critical).
11  CVSS: numerical representation (0-10) of the vulnerability severity.
12  Detection: method used to detect the issue (Tool, Manual, Exploit).
13  Report: http://link-to-report/
14  Introduced in: commit hash.
15  [End]
16
17  Reported-by: reporter name 1 <reporter-email-1@host.com>
18  Reported-by: reporter name 2 <reporter-email-2@host.com>
19  Signed-off-by: your name <your-email@yourhost.com>
20
21  [If you use an issue tracker, add reference to it here:]
22  [if external issue tracker:]
23  Bug-tracker: https://link-to-bug-tracker/id
24
25  [if github used as issue tracker:]
26  Resolves: #123
27  See also: #456, #789
```

🔗 https://tqrg.github.io/secom/

@sofiaoreis

NEW CONVENTION > SECOM

```
1   vuln-fix: subject/header containing summary of changes in ~50 characters (Vuln-ID,)
2
3   Detailed explanation of the subject/header in ~75 words.
4   (what) Explain the security issue(s) that this commit is patching.
5   (why) Focus on why this patch is important and its impact.
6   (how) Describe how the issue is patched.
7
8   [For Each Weakness in Weaknesses:]
9   Weakness: weakness identification or CWE-ID.
10  Severity: severity of the issue (Low, Medium, High, Critical).
11  CVSS: numerical representation (0-10) of the vulnerability severity.
12  Detection: method used to detect the issue (Tool, Manual, Exploit).
13  Report: http://link-to-report/
14  Introduced in: commit hash.
15  [End]
16
17  Reported-by: reporter name 1 <reporter-email-1@host.com>
18  Reported-by: reporter name 2 <reporter-email-2@host.com>
19  Signed-off-by: your name <your-email@yourhost.com>
20
21  [If you use an issue tracker, add reference to it here:]
22  [if external issue tracker:]
23  Bug-tracker: https://link-to-bug-tracker/id
24
25  [if github used as issue tracker:]
26  Resolves: #123
27  See also: #456, #789
```

Reported-by and Signed-off-by fields — *ref. Chris Beams and Linus Torvalds.*

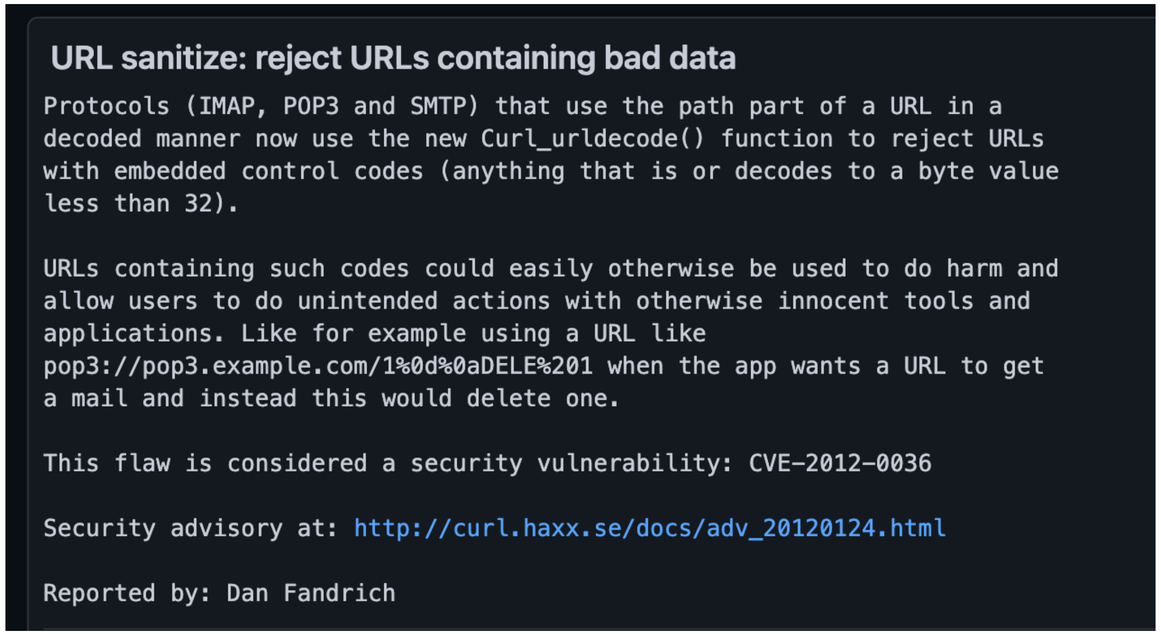
References to Bug and Issue trackers — *ref. Chris Beams and Linus Torvalds.*

🔗 https://tqrg.github.io/secom/

@sofiaoreis

Convention > Example

before



[curl/curl/75ca568]

Convention > Example

before

URL sanitize: reject URLs containing bad data

Protocols (IMAP, POP3 and SMTP) that use the path part of a URL in a decoded manner now use the new Curl_urldecode() function to reject URLs with embedded control codes (anything that is or decodes to a byte value less than 32).

URLs containing such codes could easily otherwise be used to do harm and allow users to do unintended actions with otherwise innocent tools and applications. Like for example using a URL like pop3://pop3.example.com/1%0d%0aDELE%201 when the app wants a URL to get a mail and instead this would delete one.

This flaw is considered a security vulnerability: CVE-2012-0036

Security advisory at: http://curl.haxx.se/docs/adv_20120124.html

Reported by: Dan Fandrich

[curl/curl/75ca568]

after

vuln-fix: Sanitize URLs to reject malicious data (CVE-2012-0036)

Protocols (IMAP, POP3 and SMTP) that use the path part of a URL in a decoded manner now use the new Curl_urldecode() function to reject URLs with embedded control codes (anything that is or decodes to a byte value less than 32).

URLs containing such codes could easily otherwise be used to do harm and allow users to do unintended actions with otherwise innocent tools and applications. Like for example using a URL like pop3://pop3.example.com/1%0d%0aDELE%201 when the app wants a URL to get a mail and instead this would delete one.

Weakness: CWE-89

Severity: High

Detection: Manual

Report: https://curl.se/docs/CVE-2012-0036.html

Reported-by: Dan Fandrich

Signed-off-by: Daniel Stenberg (daniel@haxx.se)

Resolves: #17940

See also: #17937

Feedback (OpenSSF)

Working groups are open to everyone! If you are interested or doing research in security, you should join!

We presented SECOM in two different OpenSSF working groups:
“Best practices” and “Vulnerability disclosure”.

Open Source Vulnerability (OSV) Database Team

GitHub Security Lab (Semmle/CodeQL)

Product Assurance and Security Team

Feedback (OpenSSF)

We presented SECOM in two different OpenSSF working groups:
“Best practices” and “Vulnerability disclosure”.

Open Source Vulnerability (OSV) Database Team

GitHub Security Lab (Semmle/CodeQL)

Product Assurance and Security Team

💯 Feedback was very positive! — One member of the OSV team provided a full review of the convention.

👍 Members of OpenSSF showed interest to apply and advocate for the convention inside their companies.

Feedback (OpenSSF)

We presented SECOM in two different OpenSSF working groups:
“Best practices” and “Vulnerability disclosure”.

Open Source Vulnerability (OSV) Database Team

GitHub Security Lab (Semmle/CodeQL)

Product Assurance and Security Team

💯 Feedback was very positive! — One member of the OSV team provided a full review of the convention.

👍 Members of OpenSSF showed interest to apply and advocate for the convention inside their companies.

🤔 Some concerns regarding adoption were raised. OpenSSF is keen to help us here!
Scoreboards; Educational programs/tutorials; and, conference talks.

Feedback (OpenSSF)

Working groups are open to everyone! If you are interested or doing research in security, you should join!

We presented SECOM in two different OpenSSF working groups: “Best practices” and “Vulnerability disclosure”.

Open Source Vulnerability (OSV) Database Team

GitHub Security Lab (Semmle/CodeQL)

Product Assurance and Security Team

💯 Feedback was very positive! — One member of the OSV team provided a full review of the convention.

👍 Members of OpenSSF showed interest to apply and advocate for the convention inside their companies.

🤔 Some concerns regarding adoption were raised. OpenSSF is keen to help us here!
Scoreboards; Educational programs/tutorials; and, conference talks.

🤩 But, in general, the OSS security community suggested that they would like to see SECOM evolve into a standard practice.

🔗 https://tqrg.github.io/secom/

@sofiaoreis

WHAT'S UP NEXT?

✅ Compliance checkers.

📍 Transform SECOM into a standard practice in the security community with the help of OpenSSF.

🥣 Auto-completion.

💭 Recommendations.

👀 Don't forget to visit our website!

Any idea on how to improve the convention?
Let us know. ⬇️

🔗 https://tqrg.github.io/secom/

@sofiaoreis