Grading on a Curve: How Rust can Facilitate New Contributors while Decreasing Vulnerabilities

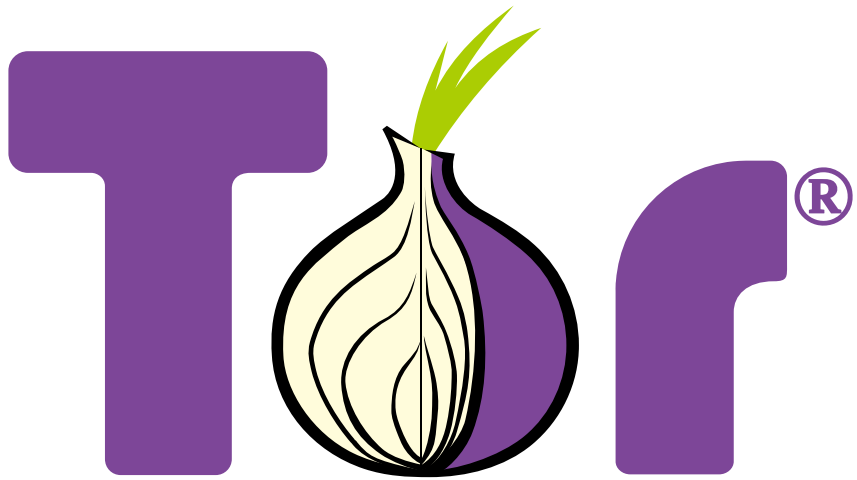
SecDev '23

Justin Tracey Ian Goldberg

University of Waterloo

October 19, 2023

What role does software serve?



TorProject.org


U.S. NAVAL
RESEARCH
LABORATORY

NATIONAL

RICO case against cop city protesters in Atlanta stirs concerns about free speech

September 21, 2023 · 4:22 PM ET

Heard on All Things Considered

Odette Yousef

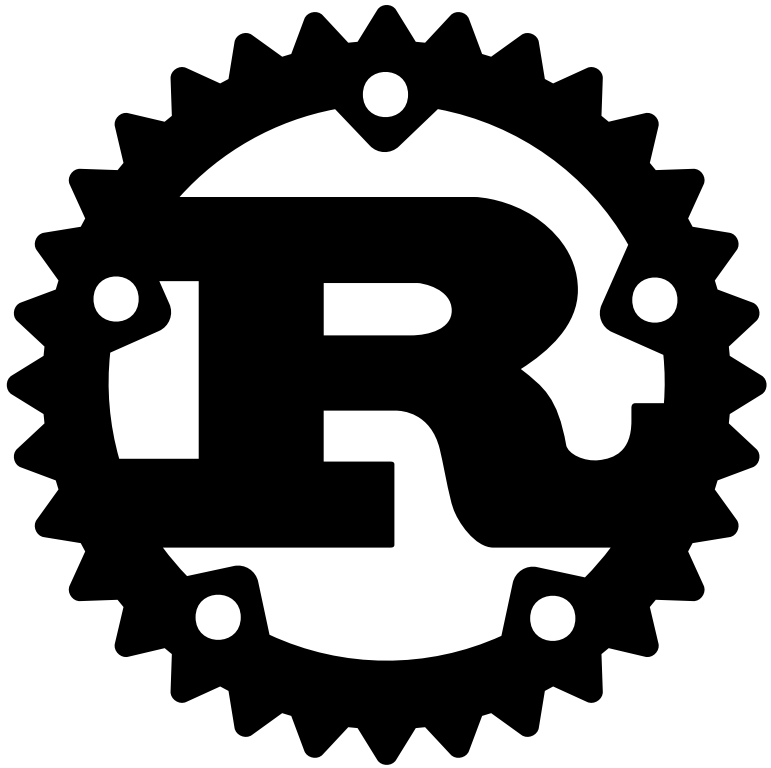
https://www.npr.org/2023/09/21/1200898062/

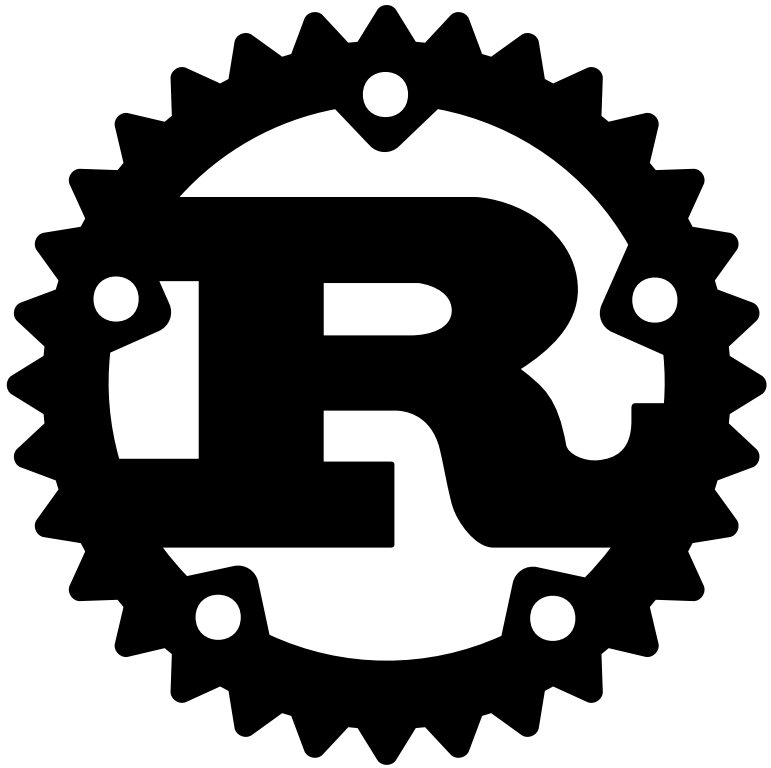

bug
smash
fund
donate.torproject.org

. . . in speaking to maintainers privately, I learned that these [new contributor] initiatives frequently cause them to seize with anxiety, because such initiatives often attract low-quality contributions. This creates more work for maintainers—all contributions, after all, must be reviewed before they are accepted. Maintainers frequently lack infrastructure to bring these contributors into a "contributor community". . .

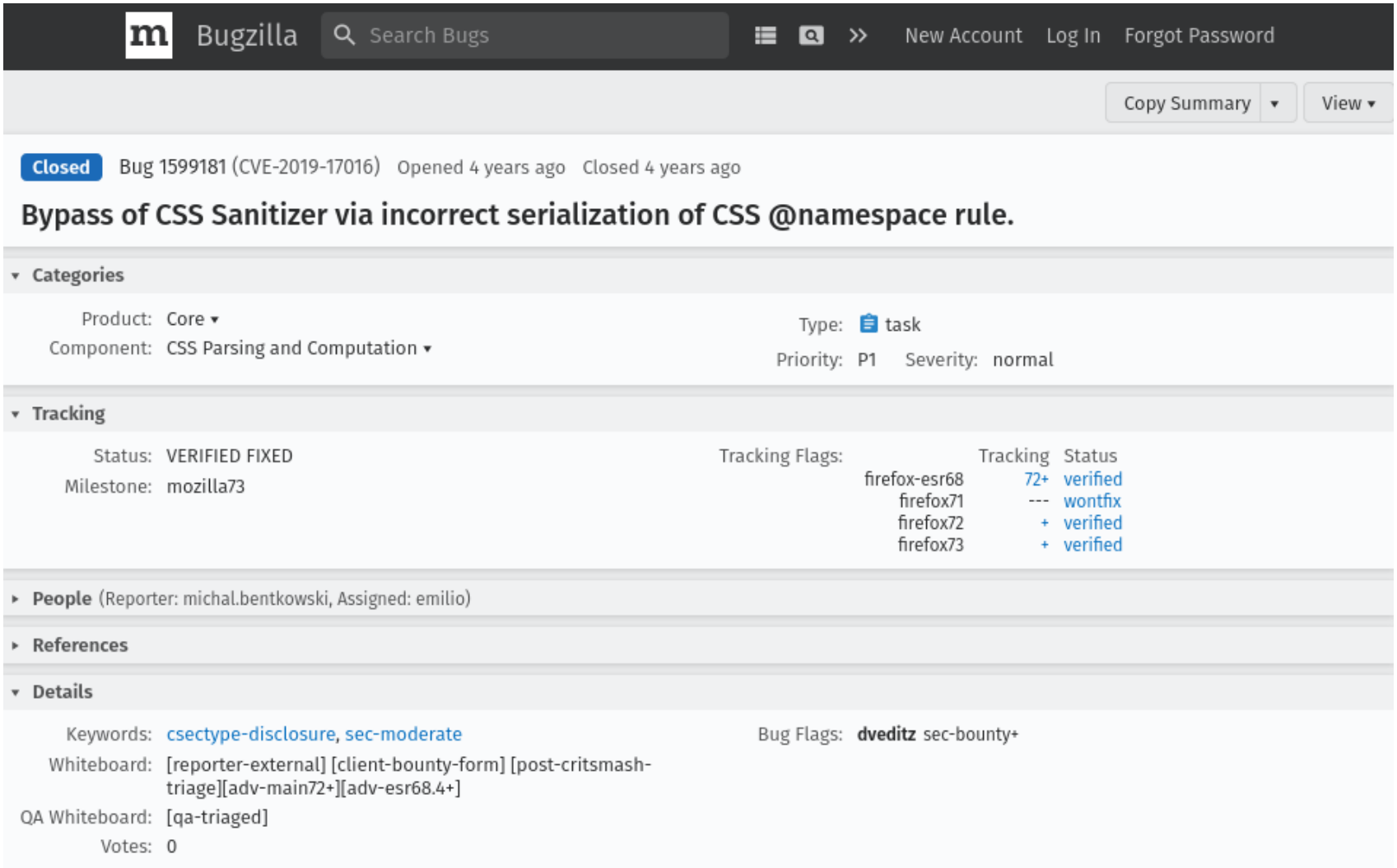
—Nadia Asparouhova[1]

[1] née Eghbal, *Working in Public*





Step 1: Identify vulnerabilities



Bugzilla | Search Bugs | New Account | Log In | Forgot Password

Copy Summary | View

Closed Bug 1599181 (CVE-2019-17016) Opened 4 years ago Closed 4 years ago

Bypass of CSS Sanitizer via incorrect serialization of CSS @namespace rule.

Categories

Product: Core
Component: CSS Parsing and Computation

Type: task
Priority: P1 Severity: normal

Tracking

Status: VERIFIED FIXED
Milestone: mozilla73

Tracking Flags:

	Tracking	Status
firefox-esr68	72+	verified
firefox71	---	wontfix
firefox72	+	verified
firefox73	+	verified

People (Reporter: michal.bentkowski, Assigned: emilio)

References

Details

Keywords: csectype-disclosure, sec-moderate
Whiteboard: [reporter-external] [client-bounty-form] [post-critsmash-triage][adv-main72+][adv-esr68.4+]
QA Whiteboard: [qa-triaged]
Votes: 0

Bug Flags: **dveditz** sec-bounty+

Step 2: Identify VCCs

```
changeset:   450877:2654955bdf43
user:        Jean-Yves Avenard <jyavenard@mozilla.com>
date:        Mon Jul 27 16:25:17 2015 -0400
summary:     Bug 1186718 - Ensure ESDS have valid size. r=kentuckyfriedtakahe,

diff --git a/media/libstagefright/frameworks/av/media/libstagefright/ESDS.cpp
--- a/media/libstagefright/frameworks/av/media/libstagefright/ESDS.cpp
+++ b/media/libstagefright/frameworks/av/media/libstagefright/ESDS.cpp
@@ -138,6 +138,9 @@ status_t ESDS::parseESDescriptor(size_t

     if (streamDependenceFlag) {
         offset += 2;
+        if (size <= 2) {
+            return ERROR_MALFORMED;
+        }
         size -= 2;
     }

@@ -147,14 +150,21 @@ status_t ESDS::parseESDescriptor(size_t
         }
         unsigned URLlength = mData[offset];
         offset += URLlength + 1;
+        if (size <= URLlength + 1) {
```

```
182522:     if (streamDependenceFlag) {
182522:         offset += 2;
182522:         size -= 2;
182522:     }
182522:
182522:     if (URL_Flag) {
182522:         if (offset >= size) {
182522:             return ERROR_MALFORMED;
182522:         }
182522:         unsigned URLlength = mData[offset];
182522:         offset += URLlength + 1;
182522:         size -= URLlength + 1;
182522:     }
182522:
182522:     if (OCRstreamFlag) {
182522:         offset += 2;
182522:         size -= 2;
182522:
182522:         if ((offset >= size || mData[offset] != kTag_DecoderConfigDescriptor)
182522:                 && offset - 2 < size
182522:                 && mData[offset - 2] == kTag_DecoderConfigDescriptor) {
182522:             // Content found "in the wild" had OCRstreamFlag set but was
```

Option 1: SZZ

Step 2: Identify VCCs

```
changeset:   450877:2654955bdf43
user:        Jean-Yves Avenard <jyavenard@mozilla.com>
date:        Mon Jul 27 16:25:17 2015 -0400
summary:     Bug 1186718 - Ensure ESDS have valid size. r=kentuckyfriedtakahe,

diff --git a/media/libstagefright/frameworks/av/media/libstagefright/ESDS.cpp
--- a/media/libstagefright/frameworks/av/media/libstagefright/ESDS.cpp
+++ b/media/libstagefright/frameworks/av/media/libstagefright/ESDS.cpp
@@ -138,6 +138,9 @@ status_t ESDS::parseESDescriptor(size_t

     if (streamDependenceFlag) {
         offset += 2;
+        if (size <= 2) {
+            return ERROR_MALFORMED;
+        }
         size -= 2;
     }

@@ -147,14 +150,21 @@ status_t ESDS::parseESDescriptor(size_t
         }
         unsigned URLlength = mData[offset];
         offset += URLlength + 1;
+        if (size <= URLlength + 1) {
```

```
182522:     if (streamDependenceFlag) {
182522:         offset += 2;
182522:         size -= 2;
182522:     }
182522:
182522:     if (URL_Flag) {
182522:         if (offset >= size) {
182522:             return ERROR_MALFORMED;
182522:         }
182522:         unsigned URLlength = mData[offset];
182522:         offset += URLlength + 1;
182522:         size -= URLlength + 1;
182522:     }
182522:
182522:     if (OCRstreamFlag) {
182522:         offset += 2;
182522:         size -= 2;
182522:
182522:         if ((offset >= size || mData[offset] != kTag_DecoderConfigDescriptor)
182522:                 && offset - 2 < size
182522:                 && mData[offset - 2] == kTag_DecoderConfigDescriptor) {
182522:             // Content found "in the wild" had OCRstreamFlag set but was
```

Option 1: SZZ

Option 2: Artisanal Hand-Crafted VCC Identifications

	C++ Projects		Rust Projects	
	samples (commits)	positives (VCCs)	samples (commits)	positives (VCCs)
MP4 parser	776	17	910	0
Unicode enc.	337	1	879	0
CSS styling	3152	9	5764	5
Rendering	6177	19	7011	7
Hyphenation	164	2	50	0
MacOS audio	134	1	841	2
Total	**10740**	**49**	**15455**	**14**

Step 3: Derive a learning curve

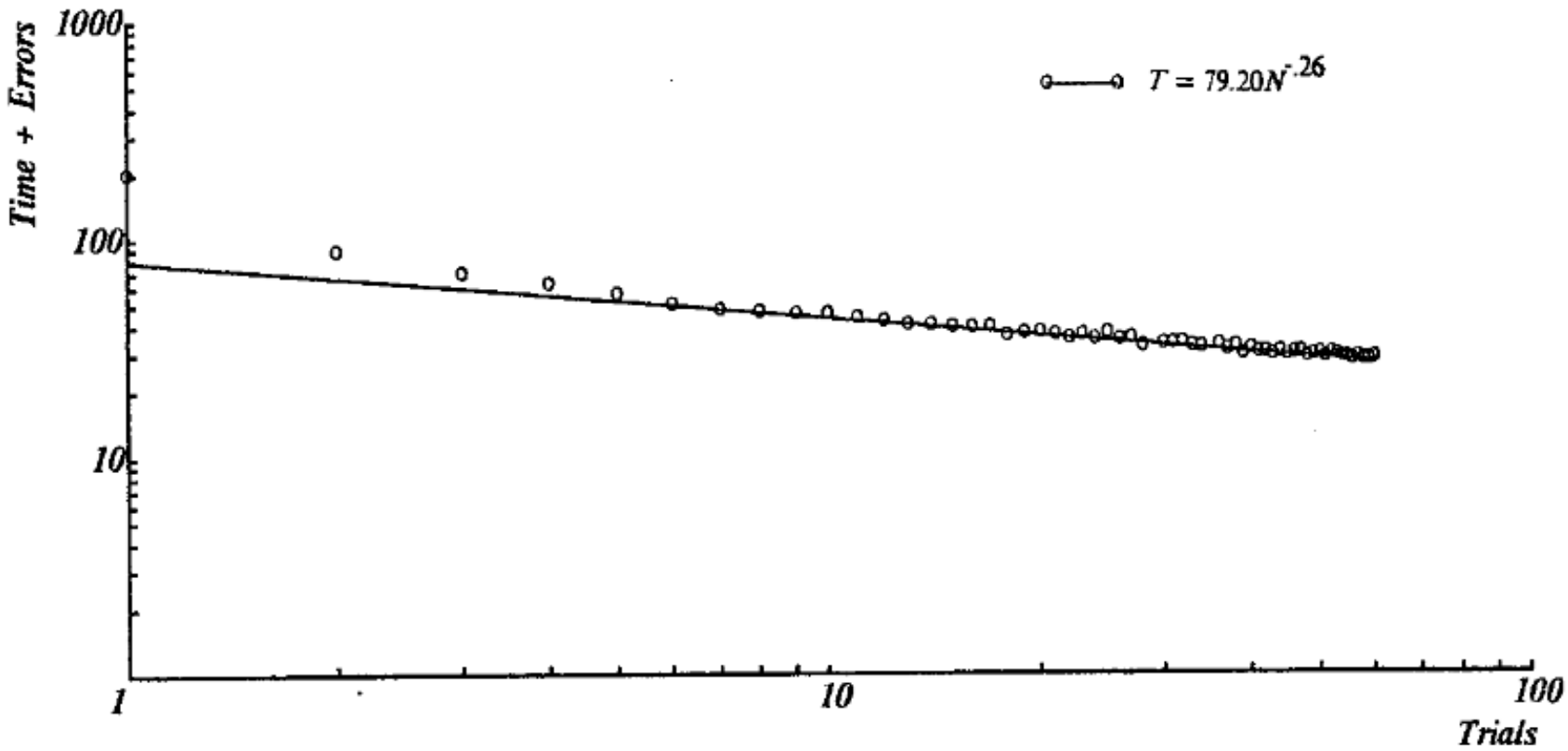
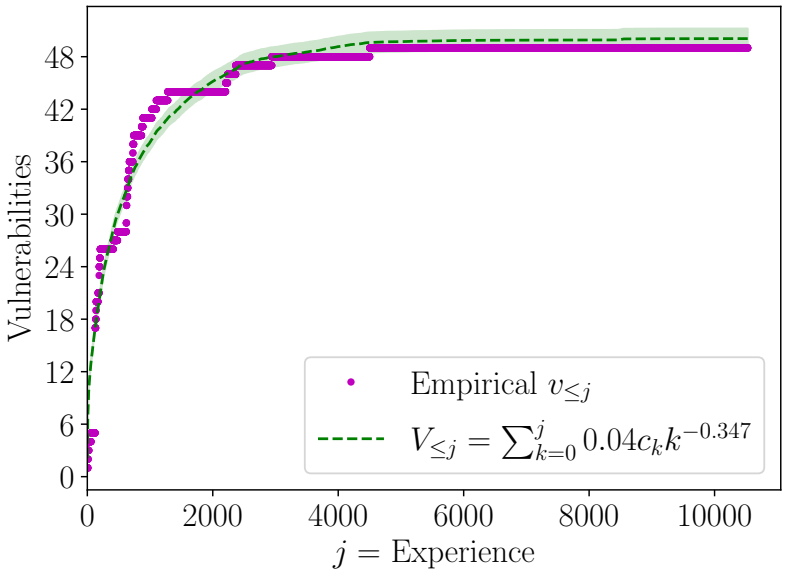
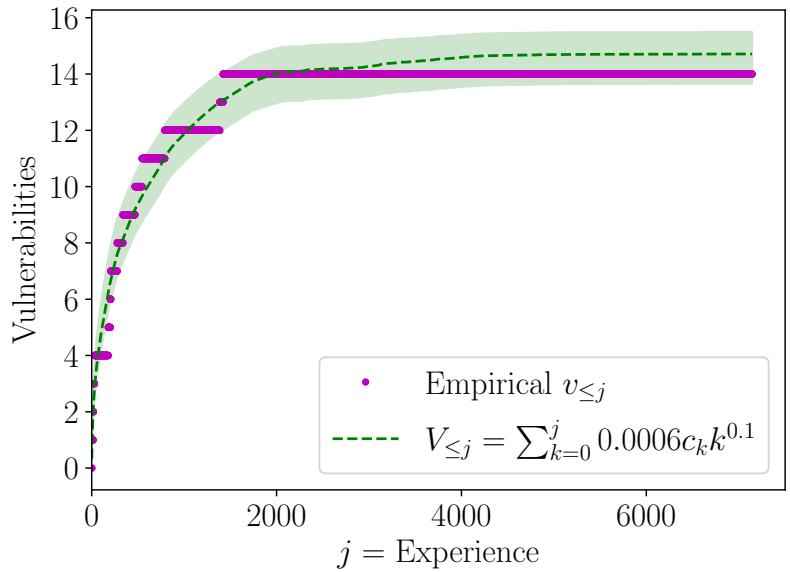


Figure 1: Learning in a Mirror Tracing Task (Log-Log Coordinates).
Replotted from Snoddy (1926).

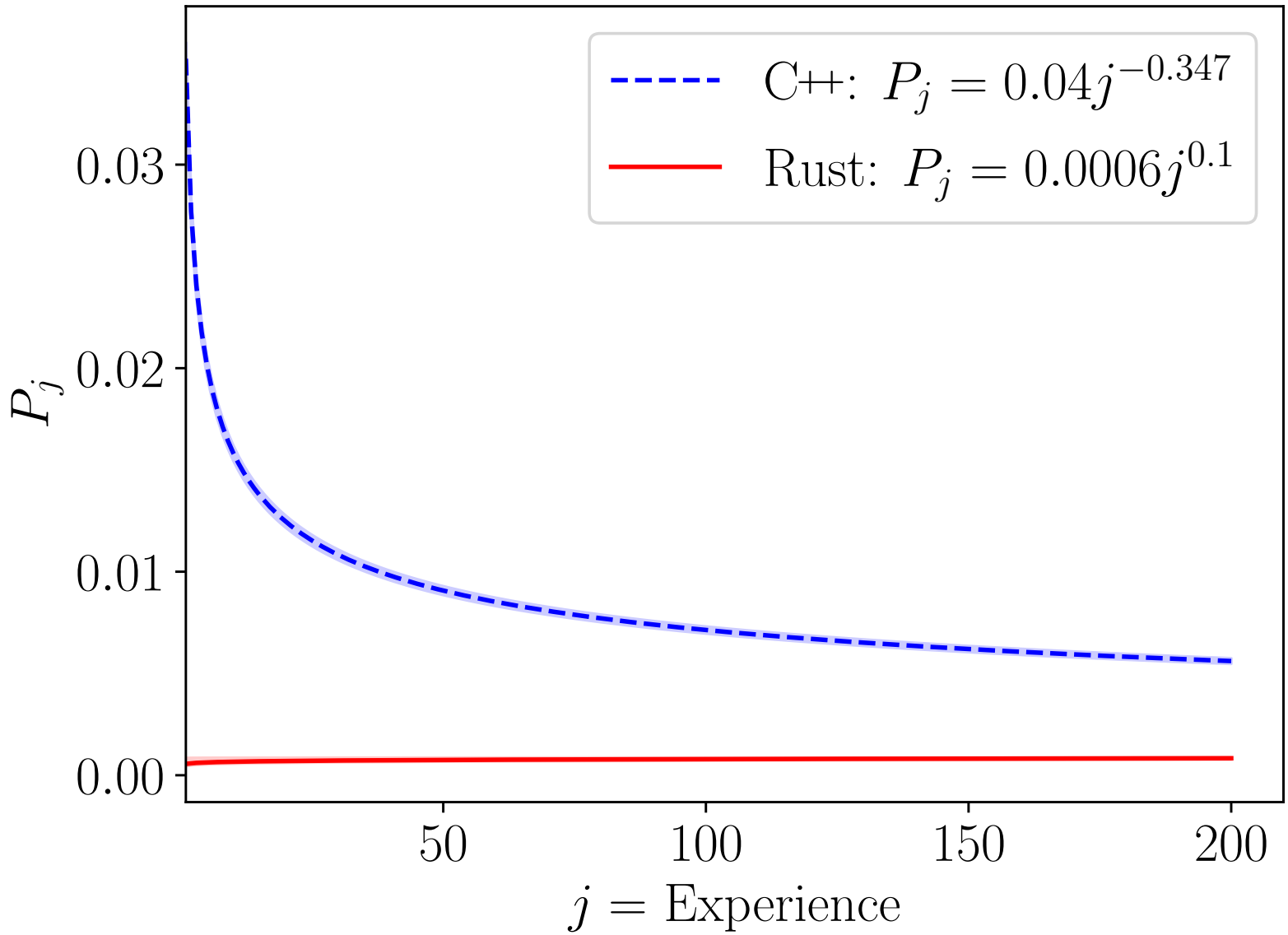
$$T = T_1 j^{-\ell}$$

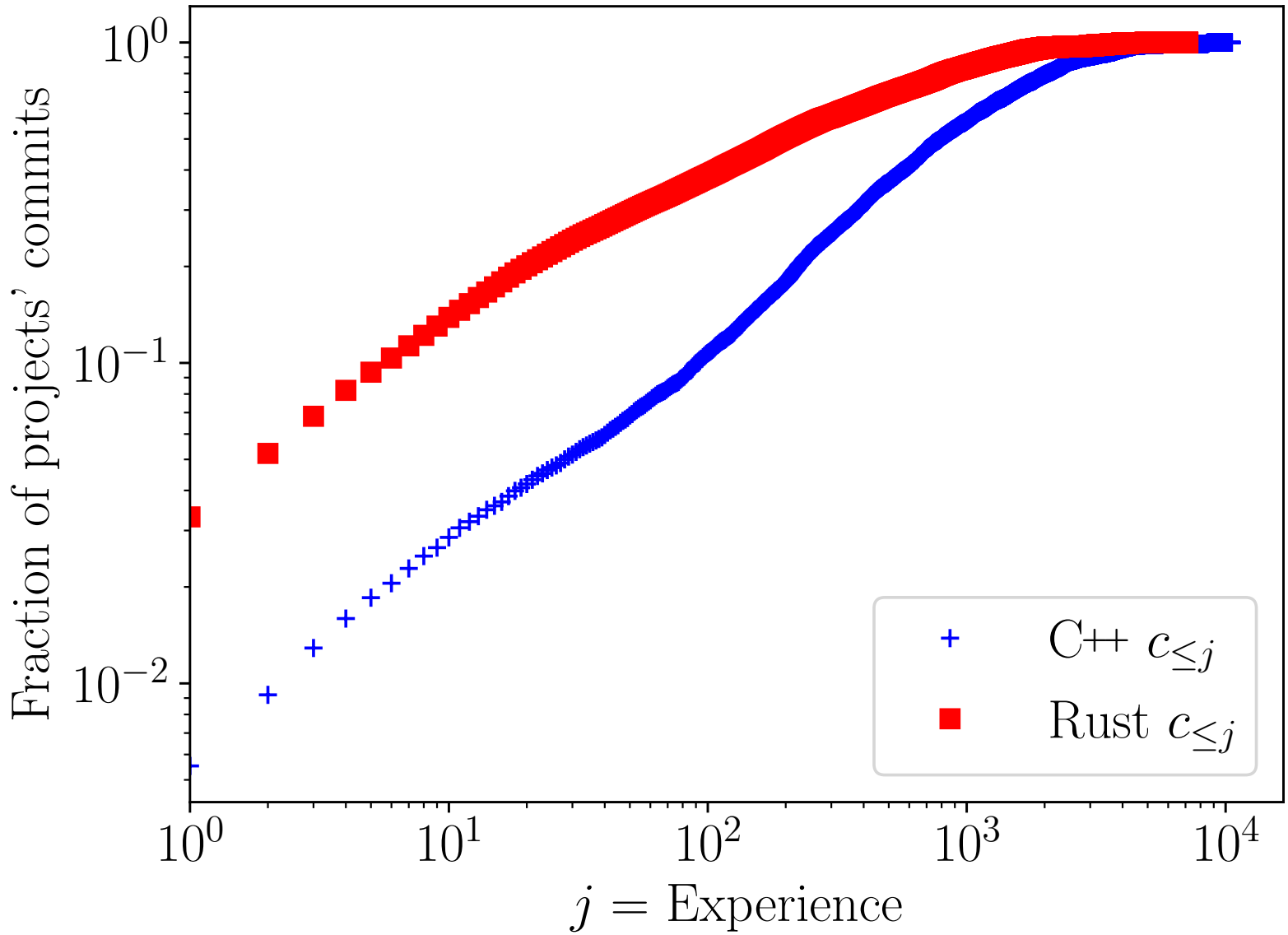

Vulnerabilities
0
6
12
18
24
30
36
42
48
0
2000
4000
6000
8000
10000
j = Experience
Empirical $v_{\leq j}$
$V_{\leq j} = \sum_{k=0}^{j} 0.04 c_k k^{-0.347}$

C++


Vulnerabilities
0
2
4
6
8
10
12
14
16
0
2000
4000
6000
j = Experience
Empirical $v_{\leq j}$
$V_{\leq j} = \sum_{k=0}^{j} 0.0006 c_k k^{0.1}$

Rust

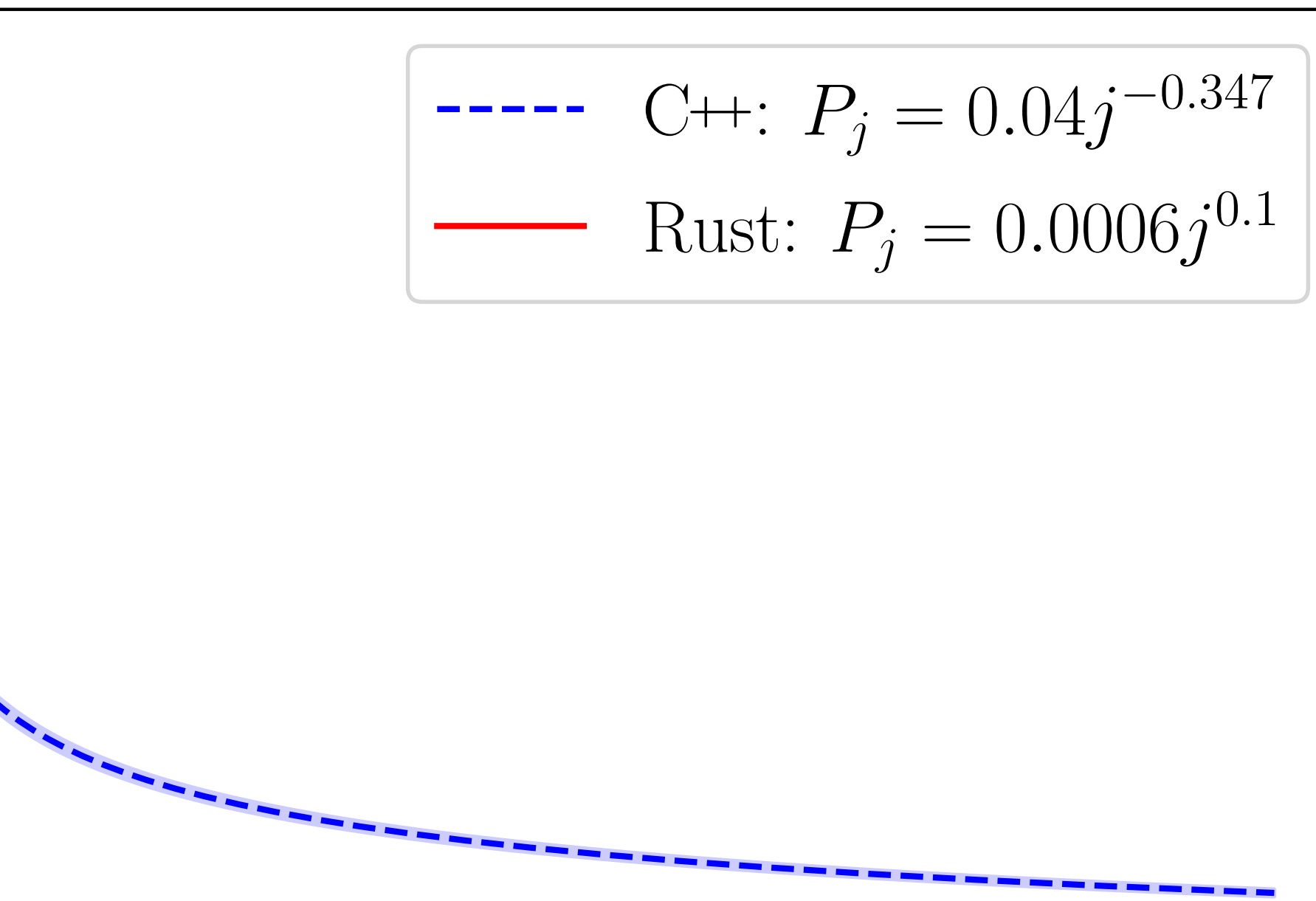

C++: $P_j = 0.04j^{-0.347}$
Rust: $P_j = 0.0006j^{0.1}$
P_j
0.03
0.02
0.01
0.00
50
100
150
200
j = Experience


10^0
10^{-1}
10^{-2}
Fraction of projects' commits
10^0
10^1
10^2
10^3
10^4
j = Experience
C++ $c_{\leq j}$
Rust $c_{\leq j}$

Conclusions

- We can use learning curves to measure experience vs. vulnerability rates
- Security of going from memory-unsafe to memory-safe: outsize benefit for newbies makes it good for maintainers
- Rust seems to attract a larger base of new contributors

`crysp.org/software`


C++: $P_j = 0.04j^{-0.347}$
Rust: $P_j = 0.0006j^{0.1}$