How Long Do Vulnerabilities Live in the Code?

A Large-Scale Empirical Measurement Study on FOSS Vulnerability Lifetimes

Nikolaos Alexopoulos, Manuel Brack, Jan Philipp Wagner, Tim Grube, Max Mühlhäuser

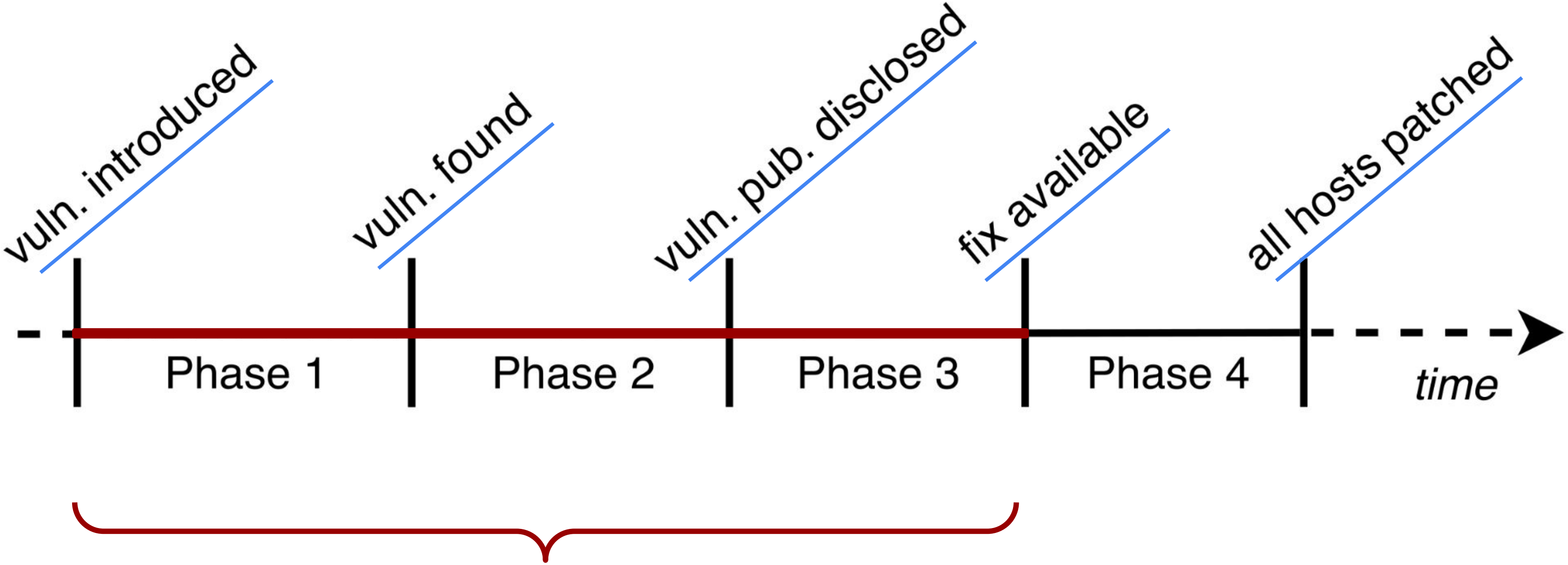
USENIX Security Symposium, Aug. 10th 2022, Boston, MA, USA

Telecooperation Lab

TECHNISCHE UNIVERSITÄT DARMSTADT

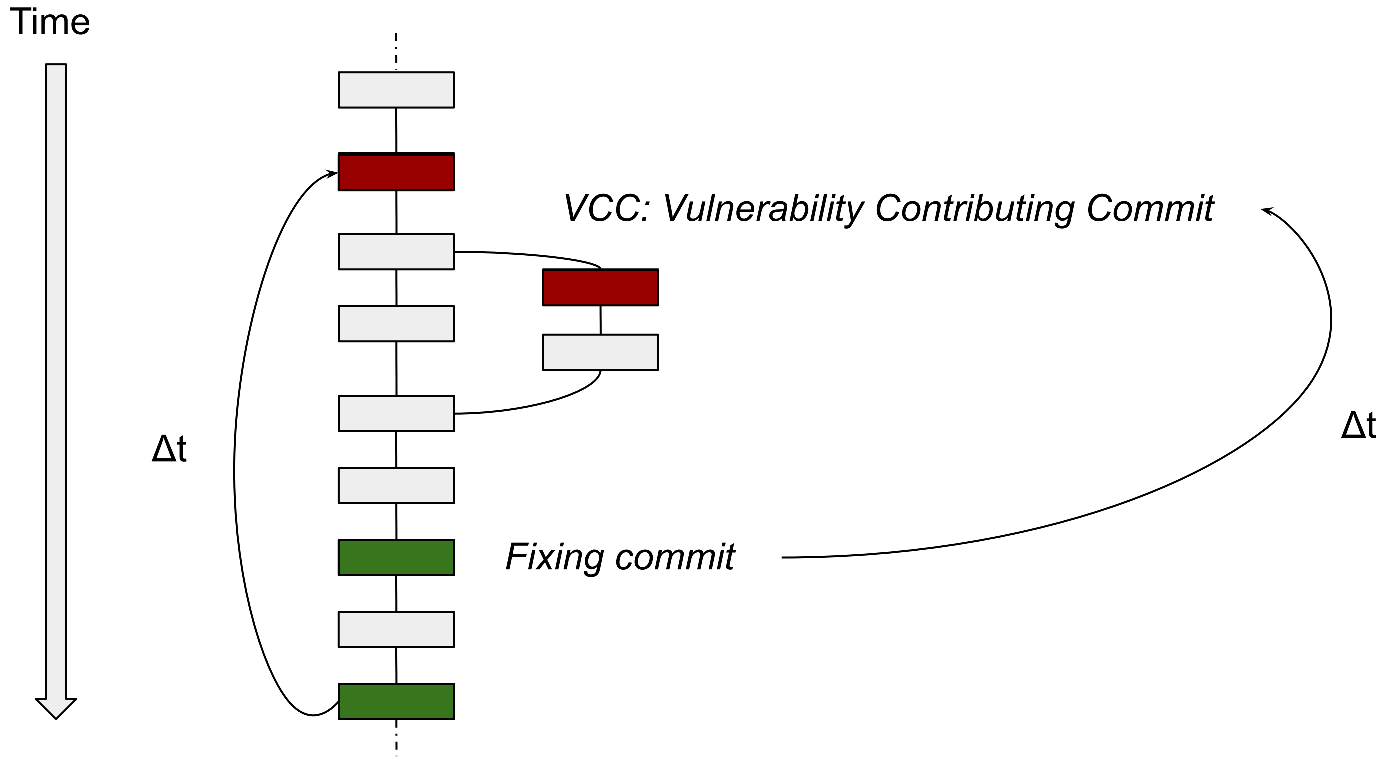
ATHENE
Nationales Forschungszentrum für angewandte Cybersicherheit

What? The vulnerability lifecycle and lifetimes



Lifetime of a vulnerability

Lifetimes in version control systems



Time

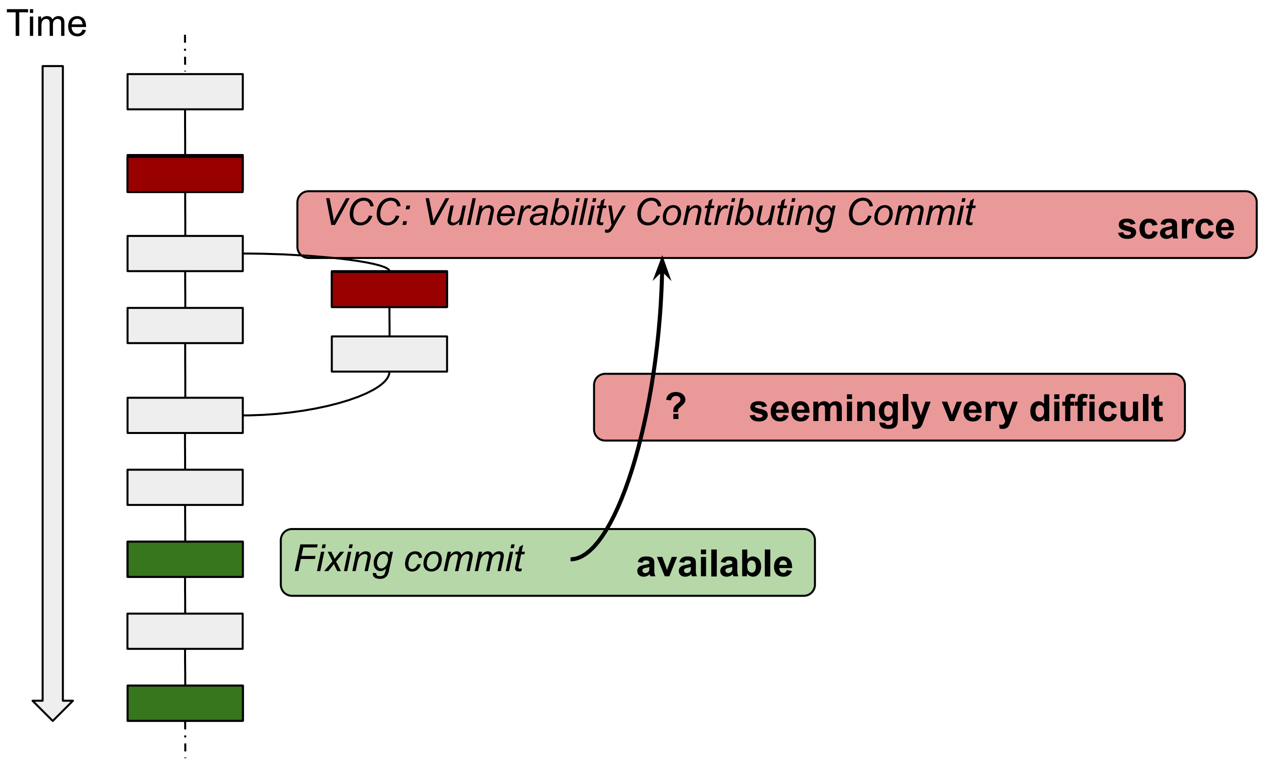
Δt

VCC: Vulnerability Contributing Commit

Δt

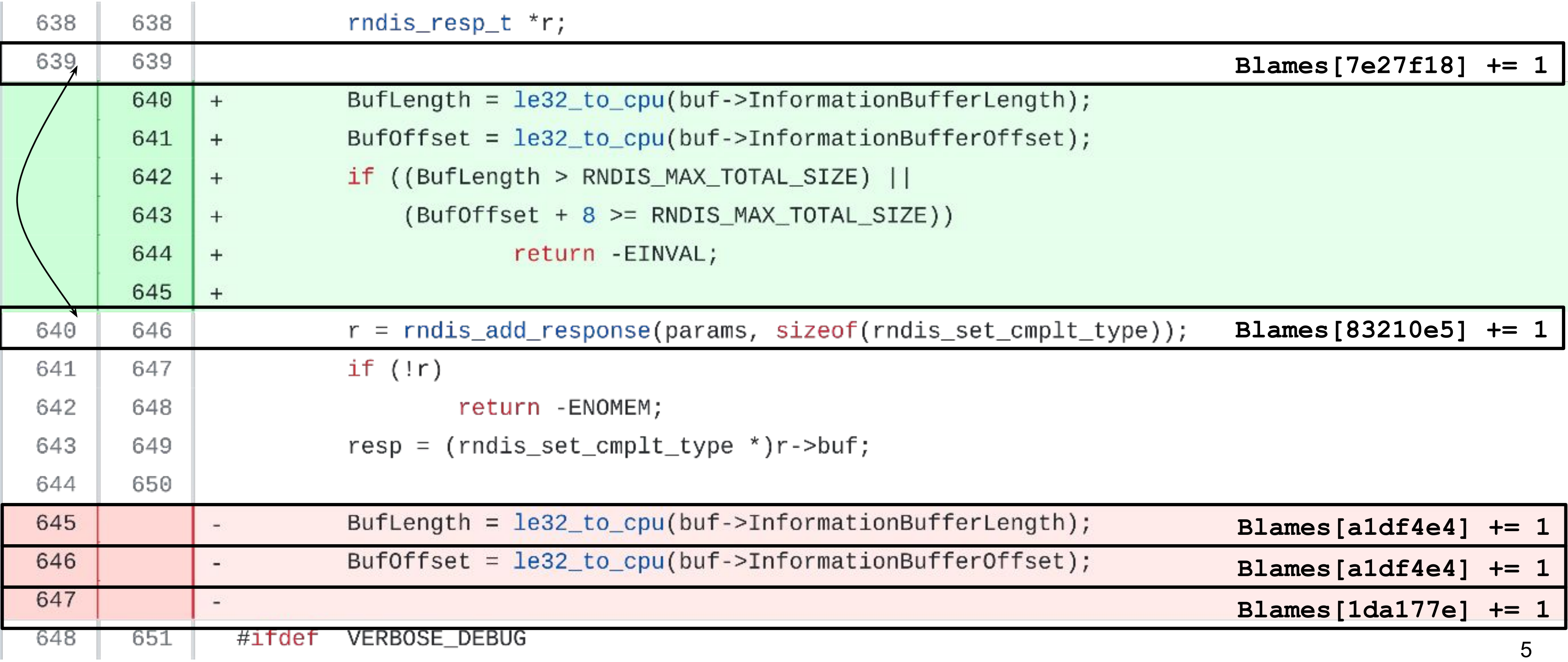
Fixing commit

Why measuring lifetimes is hard


Time
VCC: Vulnerability Contributing Commit **scarce**
? seemingly very difficult
Fixing commit **available**

VCCFinder [Perl et al. CCS 2015]

CVE-2022-25375

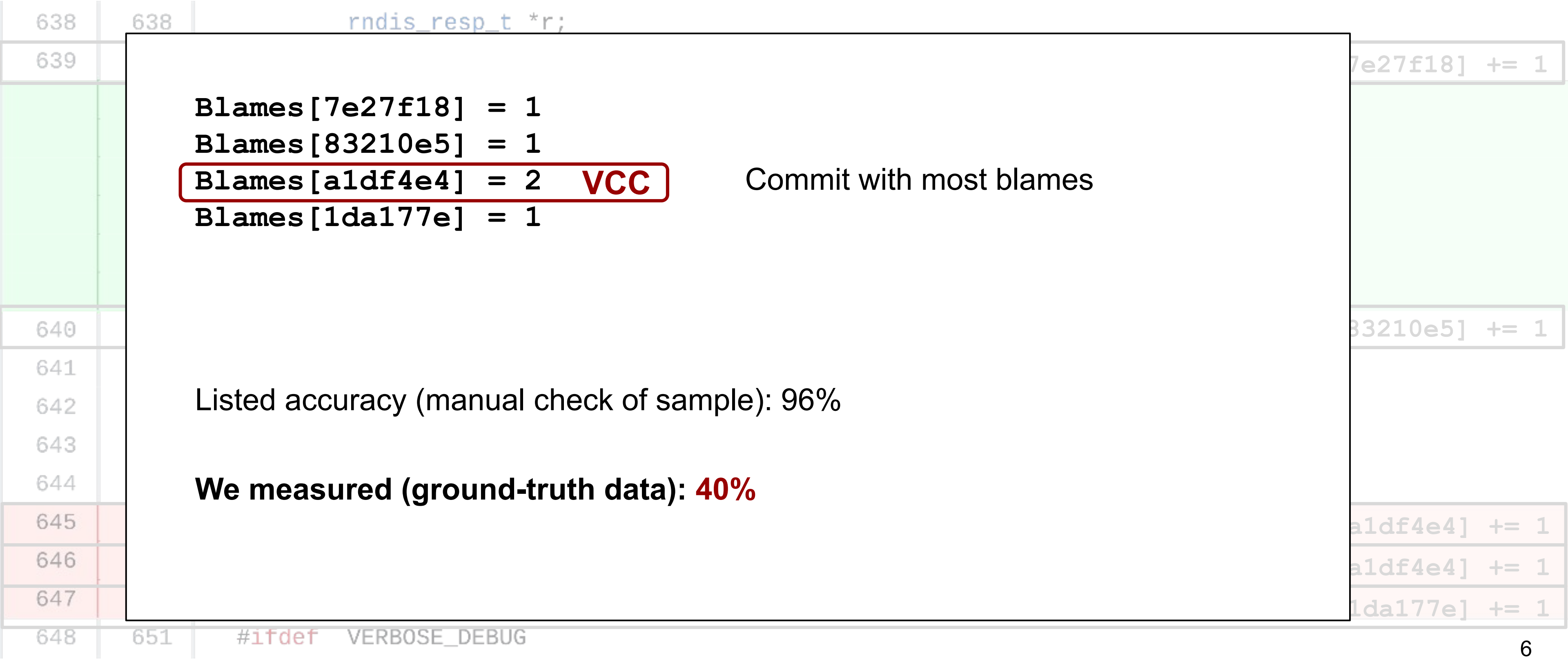


```
            rndis_resp_t *r;
                                                                          Blames[7e27f18] += 1
+           BufLength = le32_to_cpu(buf->InformationBufferLength);
+           BufOffset = le32_to_cpu(buf->InformationBufferOffset);
+           if ((BufLength > RNDIS_MAX_TOTAL_SIZE) ||
+               (BufOffset + 8 >= RNDIS_MAX_TOTAL_SIZE))
+                   return -EINVAL;
+
            r = rndis_add_response(params, sizeof(rndis_set_cmplt_type));  Blames[83210e5] += 1
            if (!r)
                    return -ENOMEM;
            resp = (rndis_set_cmplt_type *)r->buf;

-           BufLength = le32_to_cpu(buf->InformationBufferLength);        Blames[a1df4e4] += 1
-           BufOffset = le32_to_cpu(buf->InformationBufferOffset);        Blames[a1df4e4] += 1
-                                                                          Blames[1da177e] += 1
    #ifdef  VERBOSE_DEBUG
```

VCCFinder [Perl et al. CCS 2015]

CVE-2022-25375



```
            rndis_resp_t *r;
                                                            7e27f18] += 1
                                                            83210e5] += 1
                                                            a1df4e4] += 1
                                                            a1df4e4] += 1
                                                            1da177e] += 1
    #ifdef  VERBOSE_DEBUG
```

```
Blames[7e27f18] = 1
Blames[83210e5] = 1
Blames[a1df4e4] = 2   VCC
Blames[1da177e] = 1
```

Commit with most blames

Listed accuracy (manual check of sample): 96%

We measured (ground-truth data): 40%

How we did it

Key observations:

1. **We do not necessarily need to pinpoint the VCC – we just need to estimate its commit date**

How we did it (cont.)

➔ Use heuristic similar to VCCFinder with **weighted average** over the **blamed commits** (and some improvements introduced in Vuldigger [8])

$$d_h = \underline{d_{ref}} + \frac{1}{\sum_{i=1}^{n} \underline{b_i}} \sum_{i=1}^{n} \underline{b_i}(\underline{d_i} - \underline{d_{ref}})$$

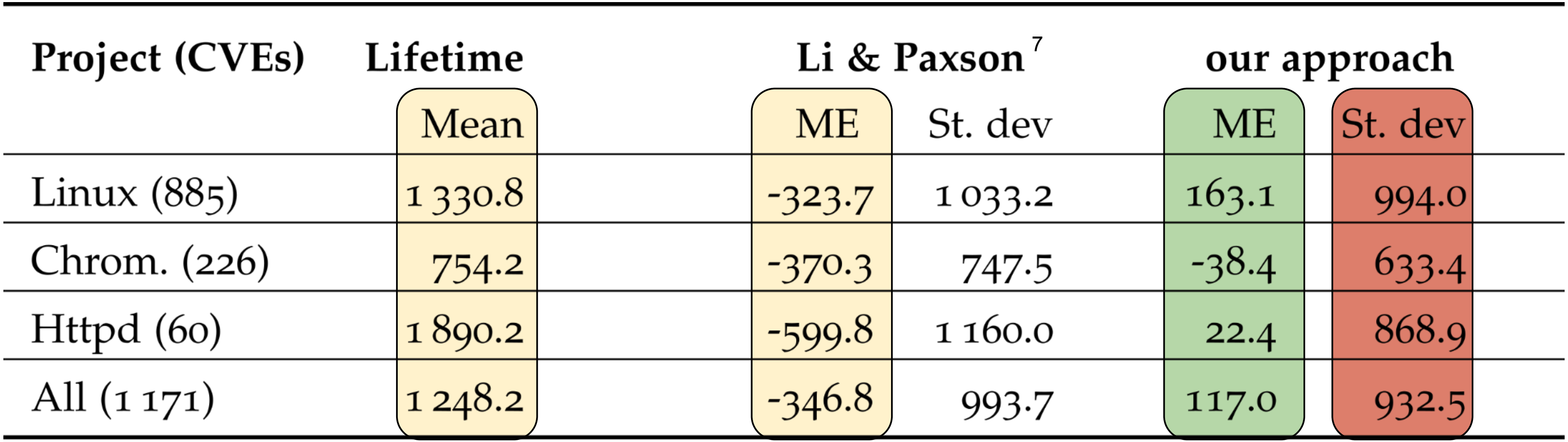
Weights: number of blames of commit i

Date of commit i

Arbitrary reference date

[8] Yang, Limin, Xiangxue Li, and Yu Yu. "Vuldigger: A just-in-time and cost-aware tool for digging vulnerability-contributing changes." GLOBECOM 2017.

Heuristic performance



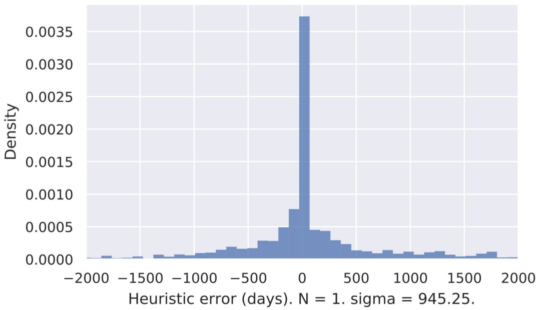
Project (CVEs)	Lifetime	Li & Paxson [7]		our approach	
	Mean	ME	St. dev	ME	St. dev
Linux (885)	1 330.8	-323.7	1 033.2	163.1	994.0
Chrom. (226)	754.2	-370.3	747.5	-38.4	633.4
Httpd (60)	1 890.2	-599.8	1 160.0	22.4	868.9
All (1 171)	1 248.2	-346.8	993.7	117.0	932.5

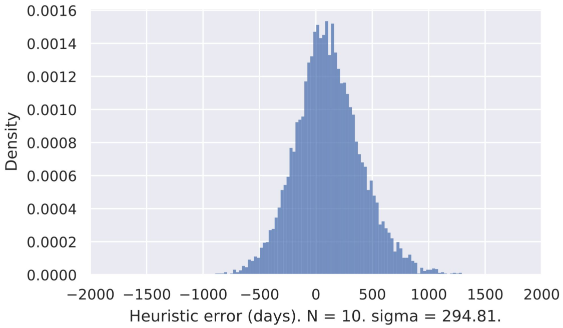
[7] Frank Li, and Vern Paxson. “A Large-Scale Empirical Study of Security Patches”. CCS 2017.

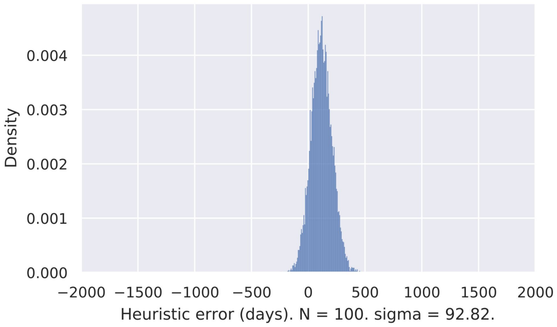
How we did it

Key observations:

1. We do not necessarily need to pinpoint the VCC – just estimate commit date
2. **We do not necessarily care about individual vulnerabilities**





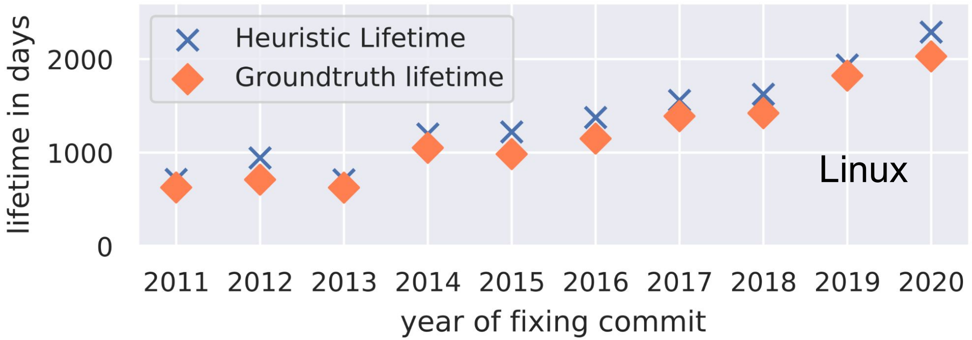


sigma ~ 1 / sqrt(N) → 10 samples 95% CI ~ ±585 days
20 samples ~ ±395 days
100 samples ~ ±176 days

Validating the heuristic

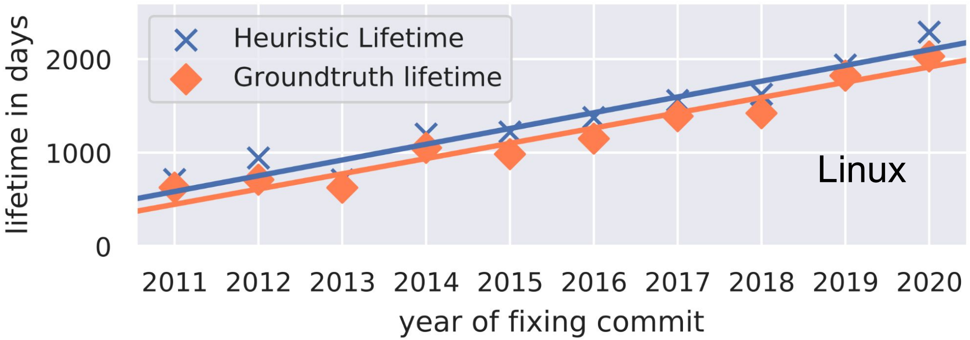
- Is the heuristic *good enough*? → We need to see how the heuristic performs in tasks similar to what we want to do

Heuristic performance (over time)



Linux: Years with >20 vulnerabilities in ground truth dataset

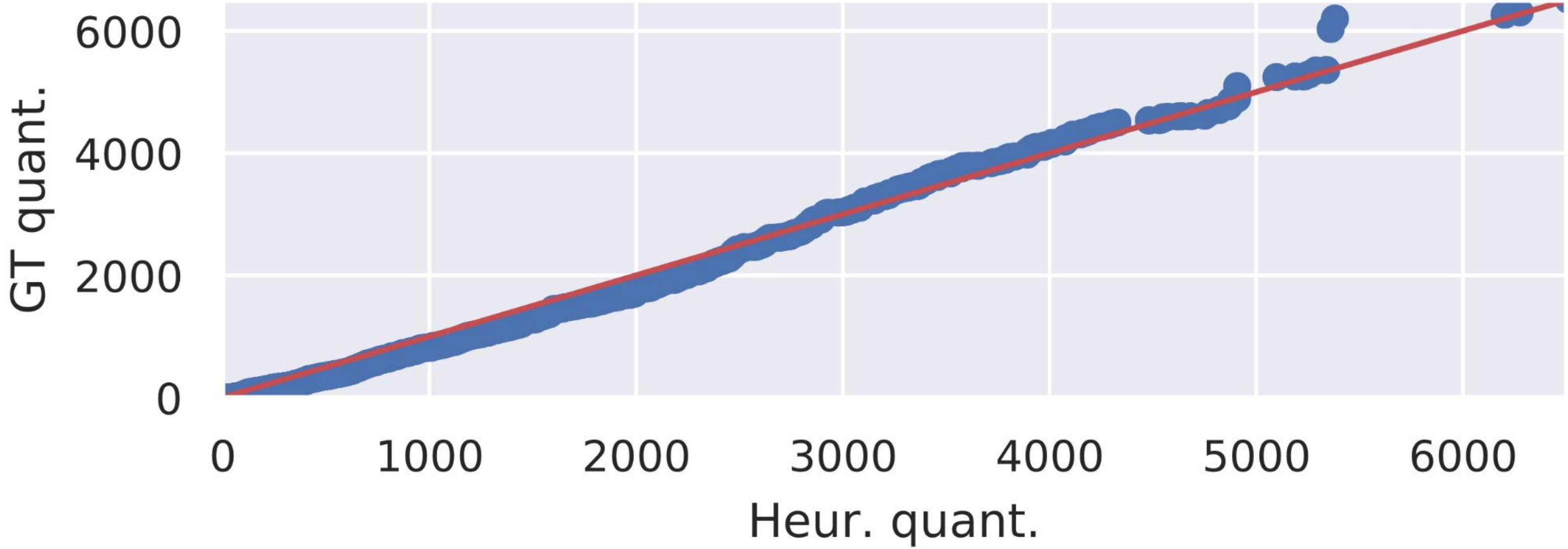
Heuristic performance (over time)



Linux: Years with >20 vulnerabilities in ground truth dataset

Heuristic performs well over time and in estimating trends

Heuristic performance (distributions)



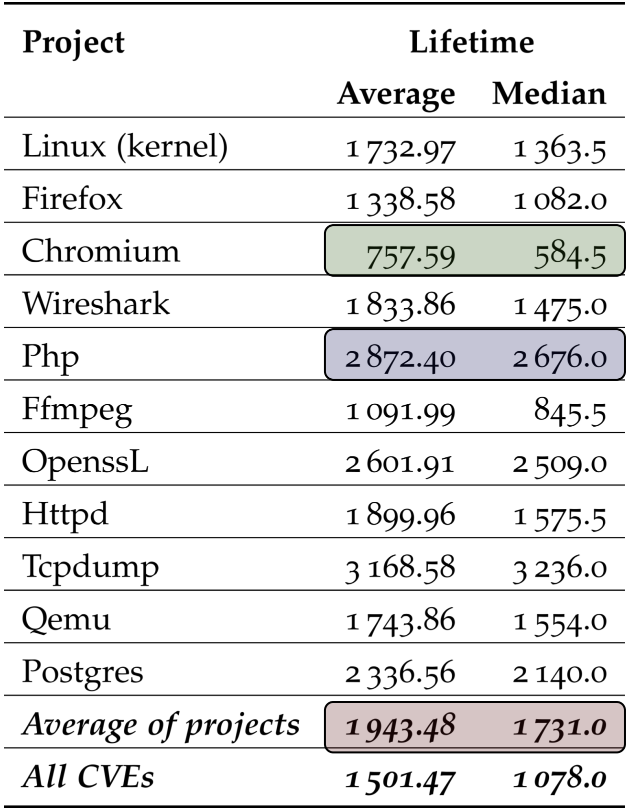
Heuristic performs well in estimating the distribution of lifetimes

Dataset

- 11 big popular FLOSS projects – multiple sources
- 1.193 CVEs with known VCC (ground truth)
- ~6.000 CVEs with known fixing commit

Results

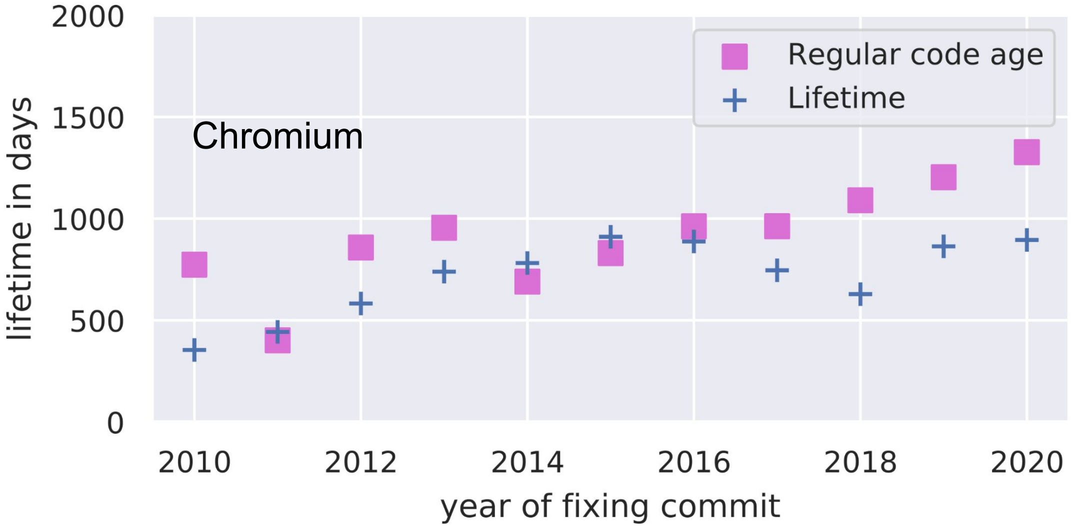
Results: lifetimes per project



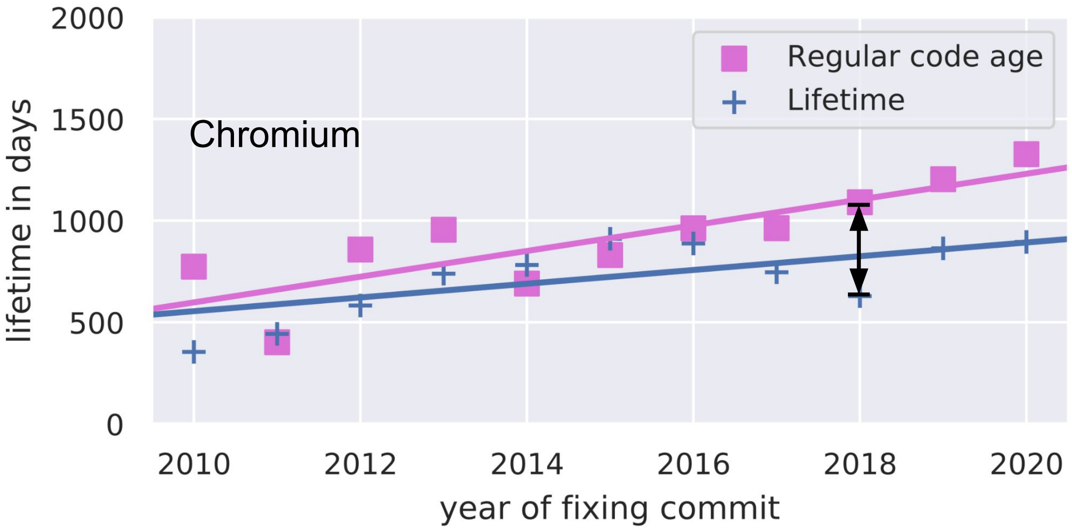
Project	Lifetime	
	Average	Median
Linux (kernel)	1 732.97	1 363.5
Firefox	1 338.58	1 082.0
Chromium	757.59	584.5
Wireshark	1 833.86	1 475.0
Php	2 872.40	2 676.0
Ffmpeg	1 091.99	845.5
OpenssL	2 601.91	2 509.0
Httpd	1 899.96	1 575.5
Tcpdump	3 168.58	3 236.0
Qemu	1 743.86	1 554.0
Postgres	2 336.56	2 140.0
Average of projects	***1 943.48***	***1 731.0***
All CVEs	***1 501.47***	***1 078.0***

- Mean: 1943 days → 5,3 years
- Median: 1731 days → 4,7 years
- Median < Mean generally
- Great variations between projects → Do shorter lifetimes mean better security?

Results: the effect of code age



Results: the effect of code age



- Vulnerability lifetime ~ age of the code at time of fix
- Identified metrics:
  - Spread
  - Rate of change of spread

Lifetimes: Implications

- Practical considerations (e.g. LTS duration, tool effectiveness)
- Theoretical insights (e.g. distribution, VDMs)
- Interesting metrics:
  - Spread between average lifetime and code age
  - Rate of change of this spread
- Enables further research

How Long Do Vulnerabilities Live in the Code?

A Large-Scale Empirical Measurement Study on FOSS Vulnerability Lifetimes

Nikolaos Alexopoulos, Manuel Brack, Jan Philipp Wagner, Tim Grube, Max Mühlhäuser

USENIX Security Symposium 2022, Boston, MA, USA

Nikolaos Alexopoulos: alexopoulos@tk.tu-darmstadt.de @nikanta0

Reproduced Artifact: https://github.com/manuelbrack/VulnerabilityLifetimes/tree/usenix_ae

Telecooperation Lab

TECHNISCHE
UNIVERSITÄT
DARMSTADT

ATHENE
Nationales Forschungszentrum
für angewandte Cybersicherheit