Building a Commit-level Dataset of Real-World Vulnerabilities

Alexis Challande, Robin David, Guénaël Renault

Quarkslab

Biography

whoami

Alexis Challande, Ph.D. student (3rd year)
Collaboration between Quarkslab and LiX (Ecole Polytechnique/Inria) in France

Problem

Situation

Let's imagine I have a shiny new tool that detect if a patch has been applied to a binary,

how do I test my solution efficiently?

Potential Solutions

- Gather a few couples of binaries from real vulnerabilities
- Create a synthetic project where I add hand-crafted vulnerabilities
- ...

Problem

Situation

Let's imagine I have a shiny new tool that detect if a patch has been applied to a binary,

how do I test my solution efficiently?

Potential Solutions

- Gather a few couples of binaries from real vulnerabilities
- Create a synthetic project where I add hand-crafted vulnerabilities
- ...
- Use the dataset presented today

Contributions

Commit Level Dataset

A dataset of more than 1,900 vulnerabilities:

- Based on real-world issues (CVEs);
- Precise at a Commit Level.

Precompiled Dataset

Binaries for a 600 of these vulnerabilities:

- 4 architectures (`ARM`, `x86`, `x86-64`, `ARM64`);
- With debug symbols.

Available on GitHub at `https://github.com/quarkslab/aosp_dataset`.

Existing Work

- Standard Testing Suites (Juliet [1], CGC [2])
  Synthetic, Small samples

- Generated Dataset (LAVA [3], MAGMA)
  Only one bug type

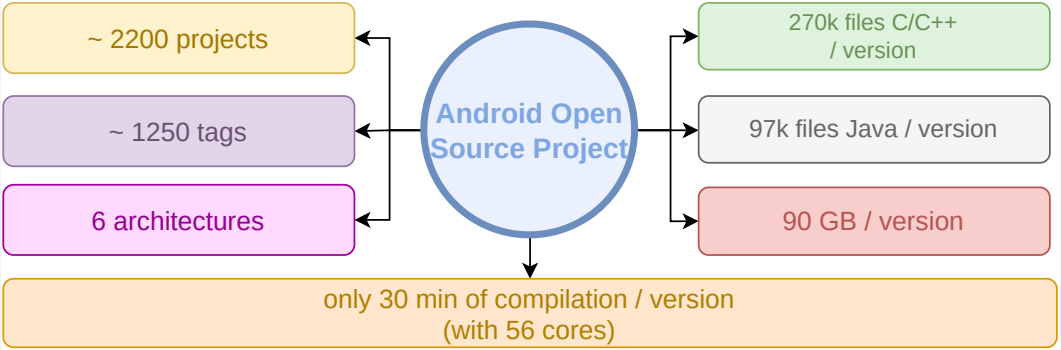
- CVE Datasets (Akram and Ping [4], Li *et al.* [5])
  Discontinued, Small dataset, Imprecise

Potential Applications

- Patch Characterization
  Draw an identity card of Patch
- Silent Fix Detection
  Detect if a commit was a security fix
- Cross Architecture Binary Diffing
  Match the same binary across different architectures
- Patch Detection
  Detect if a patch has been applied
- ...

Android Open Source Project (AOSP)

What is AOSP?



Android's security

Update process

- Multiple stakeholders (*Google*, OEM, carriers...)
- Intricated, complex, and usually not followed in time *2 years of security-update*
- *Project Treble* [6] to solve part of the problem *starting to get enforced*

Android Security Bulletins by Google

- Monthly reports of vulnerability fixes in AOSP
- Contains CVE-ID, severity and **a link to the fixing commit**

Android Security Bulletins

Framework

The most severe vulnerability in this section could lead to local escalation of privilege with no additional execution privileges needed. User interaction is not needed for exploitation.

CVE	References	Type	Severity	Updated AOSP versions
CVE-2021-39619	A-197399948	EoP	High	11, 12
CVE-2021-39663	A-200682135*	EoP	High	10
CVE-2021-39676	A-197228210*	EoP	High	11
CVE-2021-39664	A-203938029	ID	High	12

Extract of March 2022 Bulletin

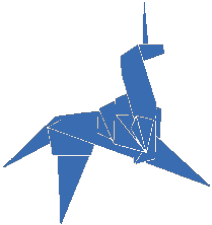
Roy: Tool Overview

Objective

Based on the *Security Patch Level*, find CVEs affecting a device.

How does it work?

1. Crawl Android Security Bulletins
2. Parse results
3. Store them in a database



Towards Binary Artefacts

Motivations

Not every vulnerability is found in Open Source.
There exists a need for binary only solutions.

Why using AOSP?

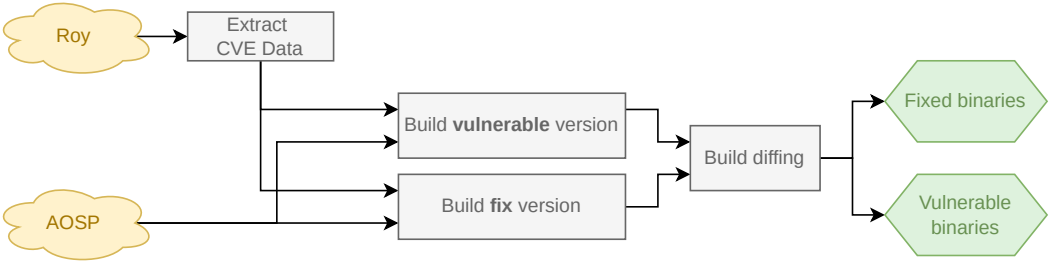
AOSP is a perfect target for building binary artefacts:

- Cross architecture Operating System
- Documented and working build system
- Information precise at a commit level (Roy)

AOSPBuilder: Overview

Process

1. Reuse Roy results for vulnerabilities
2. Prepare a build environment for AOSP
3. (Build-magic)



Dataset Overview: At Source Level

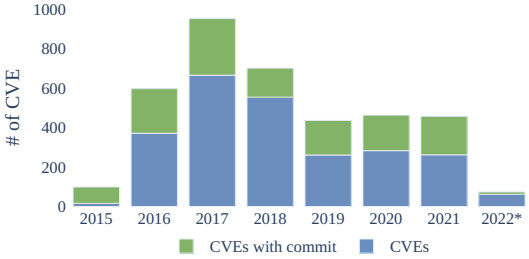
Results

- Huge set of vulnerabilities (3400+ and more than 1900 with commits)
- Ever increasing *but parser often needs to be updated*

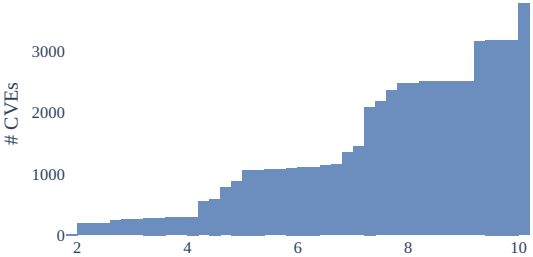
Limitations

- Only Open Source components
- Rely on Google's commitment to publish bulletins

Dataset Overview: At Source Level



CVE Evolution over time



CVSS scores

Dataset Overview: At Binary Level

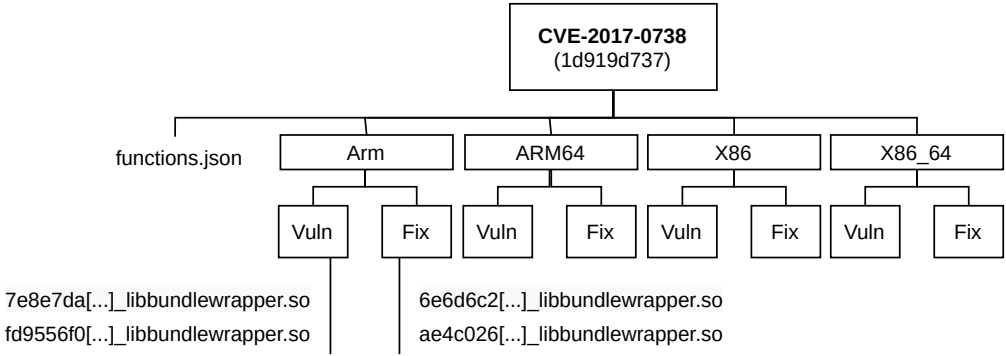
Results

- 600 vulnerabilities compiled
- 120 GB *links to download on GitHub*
- 4 architectures *the same binaries in various architectures*

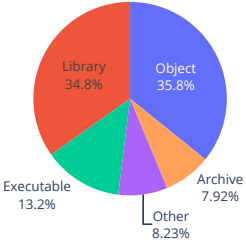
Limitations

- Only vulnerabilities on C/C++
- Build automation is hard: lots of failures
- Only vulnerabilities after Android 6

Dataset Overview: At Binary Level



Dataset Overview: At Binary Level



Binary Files Types

Name	Count
libbluetooth.so	953
bluetooth.default.so	748
libnfc-nci.so	650
libstagefright.so	421
net_test_btif	417

Most Common Binaries

Usage

- Information is stored in JSON Files
- A helper module in Python provided to ease usage

```
aosp-dataset/
├── cves
│   ├── CVE-2012-6701.json
│   ├── CVE-2012-6702.json
│   ├── ....
├── LICENSE
├── precompiled
│   └── links.json
├── pyproject.toml
├── README.md
├── schemas
│   ├── AospCve.schema.json
│   ├── precompiled.schema.json
└── src
    └── aosp_dataset
```

CVE Details

Links towards precompiled vulnerabilities

Schemas definitions

Python module

Conclusion

Dataset

- +1,900 Commit-precise real-world vulnerabilities
- +600 Precompiled artefacts for four architectures

Useful for: Patch characterization, Silent fix detection, Cross architecture diffing ...

`https://github.com/quarkslab/aosp_dataset`

Thank you

Contact information:

achallande@quarkslab.com

+33 1 58 30 81 51

https://www.quarkslab.com

Quarkslab

References I

Frederick Boland and Paul Black.
The juliet 1.1 c/c++ and java test suite.
(45).
Publisher: Computer (IEEE Computer).

DARPA.
Cyber grand challenge.

Brendan Dolan-Gavitt, Patrick Hulin, Engin Kirda, Tim Leek, Andrea Mambretti, Wil Robertson, Frederick Ulrich, and Ryan Whelan.
LAVA: Large-scale automated vulnerability addition.
In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 110–121. IEEE.

Junaid Akram and Luo Ping.
How to build a vulnerability benchmark to overcome cyber security attacks.
14(1):60–71.

References II

Frank Li and Vern Paxson.
A large-scale empirical study of security patches.
In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2201–2215. ACM.

Iliyan Malchev.
Here comes treble: A modular base for android.

Xiaojun Xu, Chang Liu, Qian Feng, Heng Yin, Le Song, and Dawn Song.
Neural network-based graph embedding for cross-platform binary code similarity detection.
pages 363–376.

Bingchang Liu, Wei Huo, Chao Zhang, Wenchao Li, Feng Li, Aihua Piao, and Wei Zou.
αdiff: cross-version binary code similarity detection with DNN.
In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering - ASE 2018*, pages 667–678. ACM Press.

References III

Jiyong Jang, Abeer Agrawal, and David Brumley.
ReDeBug: Finding unpatched code clones in entire OS distributions.
In *2012 IEEE Symposium on Security and Privacy*, pages 48–62. IEEE.

Henning Perl, Sergej Dechand, Matthew Smith, Daniel Arp, Fabian Yamaguchi, Konrad Rieck, Sascha Fahl, and Yasemin Acar.
VCCFinder: Finding potential vulnerabilities in open-source projects to assist code audits.
In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS '15*, pages 426–437. ACM Press.

Zhen Liu, Qiang Wei, and Yan Cao.
VFDETECT: A vulnerable code clone detection system based on vulnerability fingerprint.
In *2017 IEEE 3rd Information Technology and Mechatronics Engineering Conference (ITOEC)*, pages 548–553. IEEE.

References IV

Zhen Li, Deqing Zou, Shouhuai Xu, Hai Jin, Hanchao Qi, and Jie Hu.
VulPecker: an automated vulnerability detection system based on code similarity analysis.
In *Proceedings of the 32nd Annual Conference on Computer Security Applications*, pages 201–213. ACM.

Yaqin Zhou and Asankhaya Sharma.
Automated identification of security issues from commit messages and bug reports.
In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering - ESEC/FSE 2017*, pages 914–919. ACM Press.

Antonio Nappa, Richard Johnson, Leyla Bilge, Juan Caballero, and Tudor Dumitras.
The attack of the clones: A study of the impact of shared code on vulnerability patching.
In *2015 IEEE Symposium on Security and Privacy*, pages 692–708. IEEE.

References V

Google.
Android security bulletins.

MITRE Corporation.
MITRE.

Alexandre Dulaunoy and Pieter-Jan Moreels.
cve-search - a free software to collect, search and analyse common vulnerabilities and exposures in software.

Serena Elisa Ponta, Henrik Plate, Antonino Sabetta, Michele Bezzi, and Cedric Dangremont.
A manually-curated dataset of fixes to vulnerabilities of open-source software.
In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, pages 383–387. IEEE.

References VI

Sadegh Farhang, Mehmet Bahadir Kirdan, Aron Laszka, and Jens Grossklags.
Hey google, what exactly do your security patches tell us? a large-scale empirical study on android patched vulnerabilities.

Yifei Xu, Zhengzi Xu, Bihuan Chen, Fu Song, Yang Liu, and Ting Liu.
BinXRay: Patch based vulnerability matching for binary programs.
In *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pages 376–387. ACM.

Bas van Schaik and Kevin Backhouse.
FPs are cheap. show me the CVEs!

CVE-2020-0471.

Seulbae Kim, Seunghoon Woo, Heejo Lee, and Hakjoo Oh.
VUDDY: A scalable approach for vulnerable code clone discovery.
In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 595–614. IEEE.

References VII

Yue Duan, Xuezixiang Li, Jinghan Wang, and Heng Yin.
DeepBinDiff: Learning program-wide code representations for binary diffing.
In *Proceedings 2020 Network and Distributed System Security Symposium*. Internet Society.

Guru Prasad Bhandari, Amara Naseer, and Leon Moonen.
CVEfixes: Automated collection of vulnerabilities and their fixes from open-source software.

Hang Zhang and Zhiyun Qian.
Precise and accurate patch presence test for binaries.
In *27th USENIX Security Symposium (USENIX Security 18)*, pages 887–902. USENIX Association.

Yinxing Xue, Zhengzi Xu, Mahinthan Chandramohan, and Yang Liu.
Accurate and scalable cross-architecture cross-OS binary code search with emulation.
45(11):1125–1149.

References VIII

Min-je Choi, Sehun Jeong, Hakjoo Oh, and Jaegul Choo.
End-to-end prediction of buffer overruns from raw source code via neural memory networks.

Patrick Morrison, Kim Herzig, Brendan Murphy, and Laurie Williams.
Challenges with applying vulnerability prediction models.
In *Proceedings of the 2015 Symposium and Bootcamp on the Science of Security*, pages 1–9. ACM.

Zhen Li, Deqing Zou, Shouhuai Xu, Xinyu Ou, Hai Jin, Sujuan Wang, Zhijun Deng, and Yuyi Zhong.
VulDeePecker: A deep learning-based system for vulnerability detection.

References IX

Xinda Wang, Kun Sun, Archer Batcheller, and Sushil Jajodia.
Detecting ”0-day” vulnerability: An empirical study of secret security patch in OSS.
In *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 485–492.
ISSN: 1530-0889.

Paul E. Black.
A software assurance reference dataset: Thousands of programs with known bugs.
123:123005.

Universal ctags.
original-date: 2010-03-25T10:43:13Z.

Google.
gitiles - git at google.