Pyrrha & Friends

Diving into Firmware Cartography

June 4th, 2025—SSTIC 2025

Eloïse Brocas — <ebrocas@quarkslab.com>
Robin David — <rdavid@quarkslab.com>

Quarkslab

Structured Firmware Analysis

> *The term **firmware** is more generally used to describe the software that is embedded in a hardware device.*
>
> *Costin et al.*

Monolithic Firmware

- One blob of data

  (ex: small IoT like toothbrush)

Structured Firmware

- Filesystem

  (ex: router, phone...)

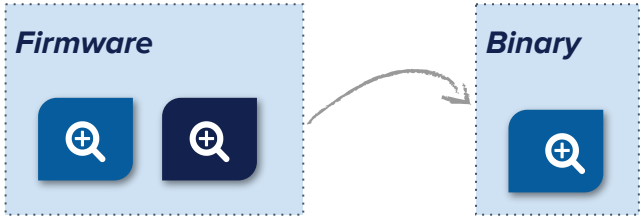
Pyrrha

Visualize firmware components and their interactions.

Quarkslab

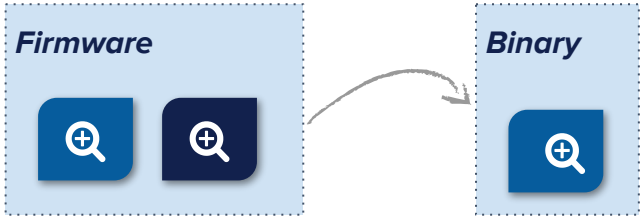
Pyrrha

A Mapper Collection



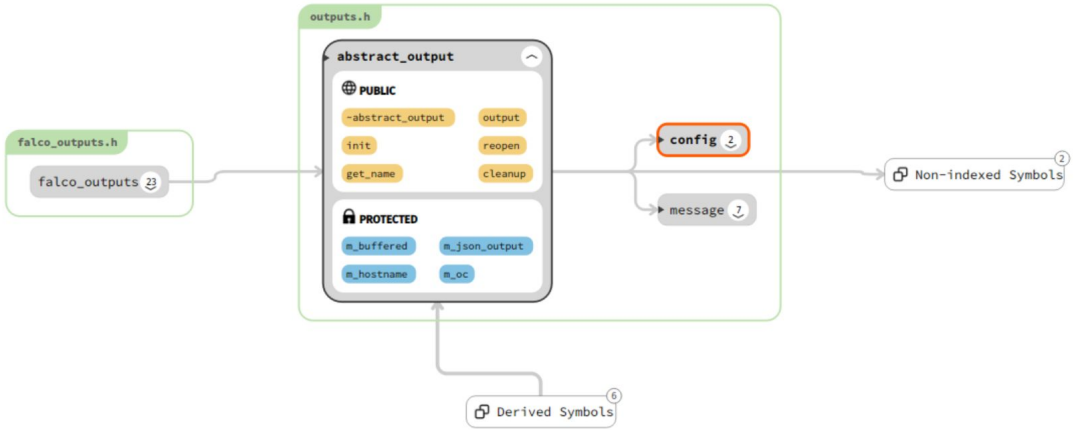
Pyrrha

A Mapper Collection



A Visualization Interface

- NumbatUI
  (Extended Sourcetrail Fork)
- Python API to create DB



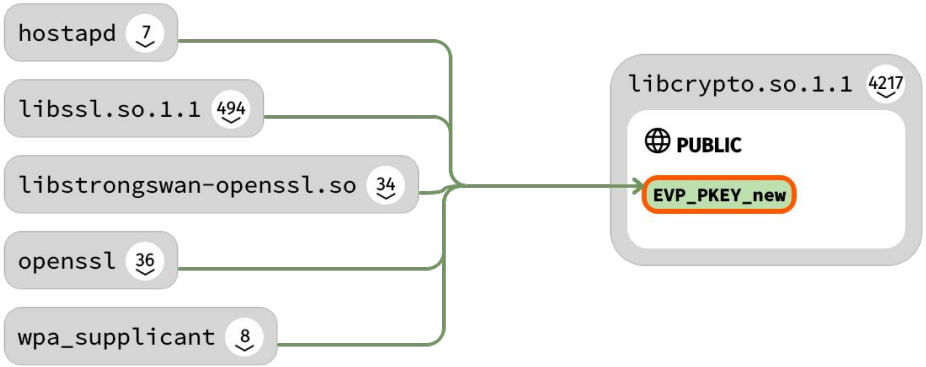
Sourcetrail: A Code Source Explorer

Mapper #1: Filesystem *(Overview)*

fs Imports/Exports

- Libraries/Executables
- Symlinks

Based on lief



Speed: ++ | Precision: − | Dependency: +

Mapper #2: Inter-Binary Call Graph

`fs`

Imports/Exports

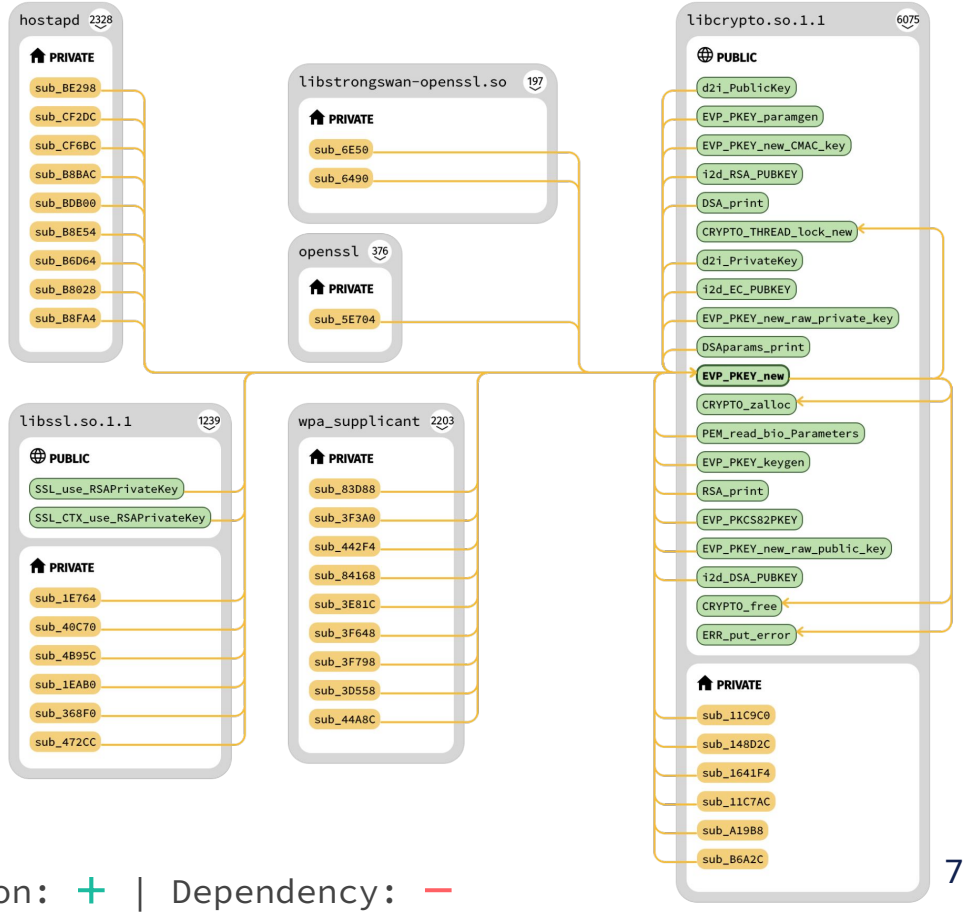
- Libraries/Executables
- Symlinks

Based on lief

+ disassembler

Inter-Binary Call Graph `fs-cg`

- Calls dependencies
- Resolve calls across binaries



Speed: − | Precision: + | Dependency: −

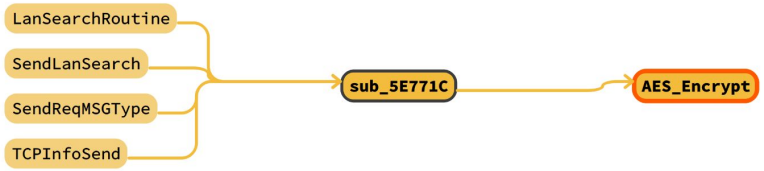
Mapper #3: Decompiled Binary

Decompiled

- Call-Graph
- Decompiled code

Based on IDA

exedecomp



5 references

sub_5E771C* 1 reference

```
1  int __fastcall sub_5E771C(char *a1)
2  {
3    return AES_Encrypt(a1, (int)&key, expandKeyLen);
4  }
```

LanSearchRoutine* 1 reference

```
..  LanSearchRoutine
46              *((_DWORD *)v3 + 1) = time(0);
47              *((_DWORD *)v4 + 2) = v2;
48              DataHton();
49              sub_5E771C(v4);
50              if ( sendto(v0, v4, 0x10u, 0, &addr, 0x10u) < 0 )
51                goto LABEL_14;
52              free(v4);
```

SendLanSearch* 1 reference

SendReqMSGType* 1 reference

TCPInfoSend* 1 reference

Speed: + | Precision: ++ | Dependency: −−

Demo

Xiaomi mesh WiFi router

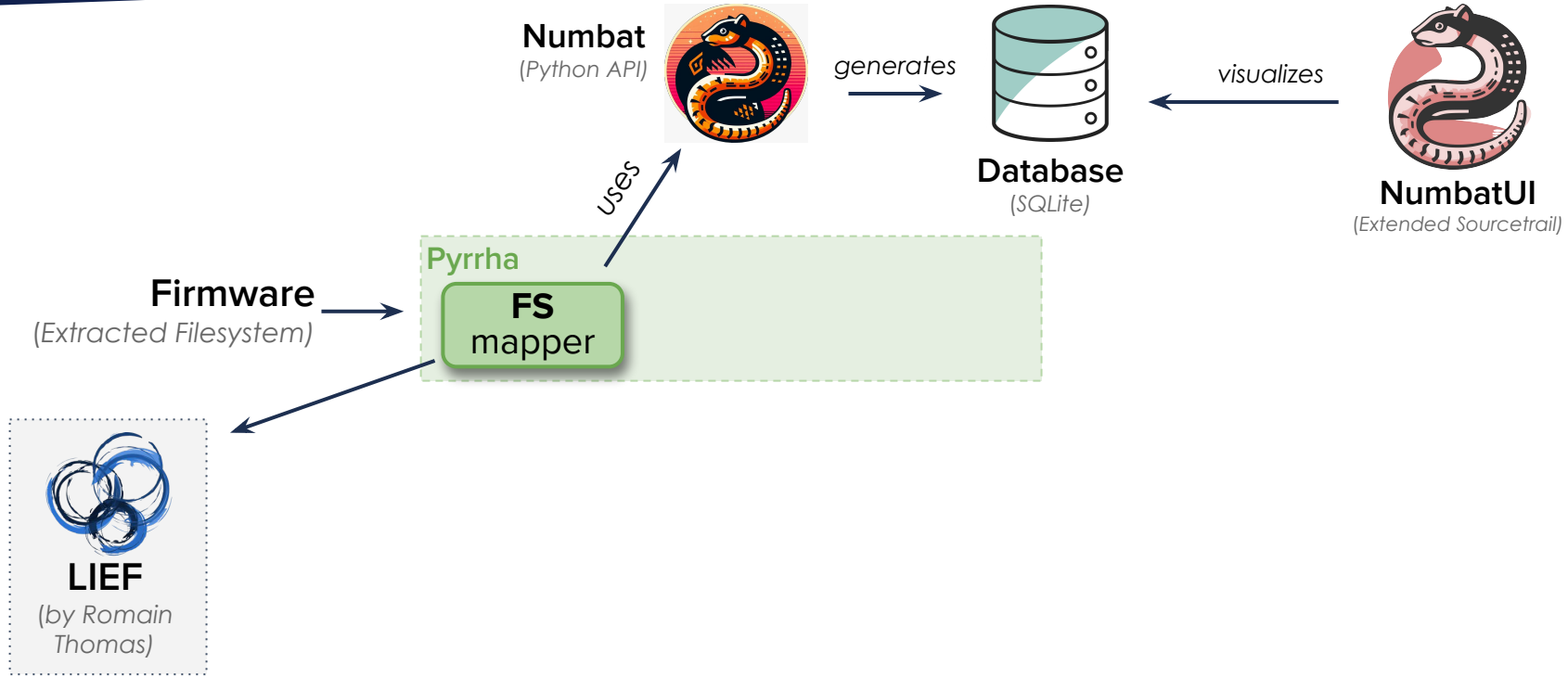
Quarkslab

Under The Hood

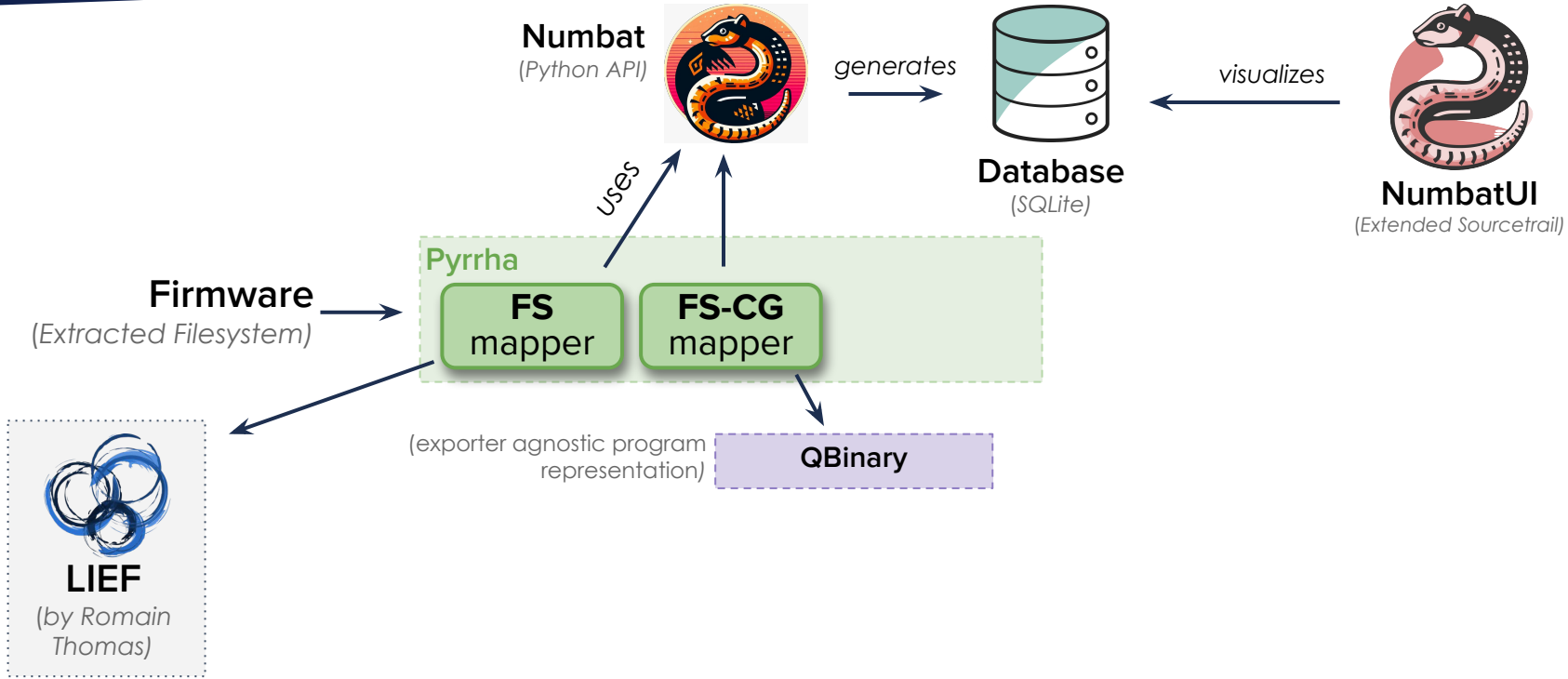
(Pyrrha & friends)

Quarkslab

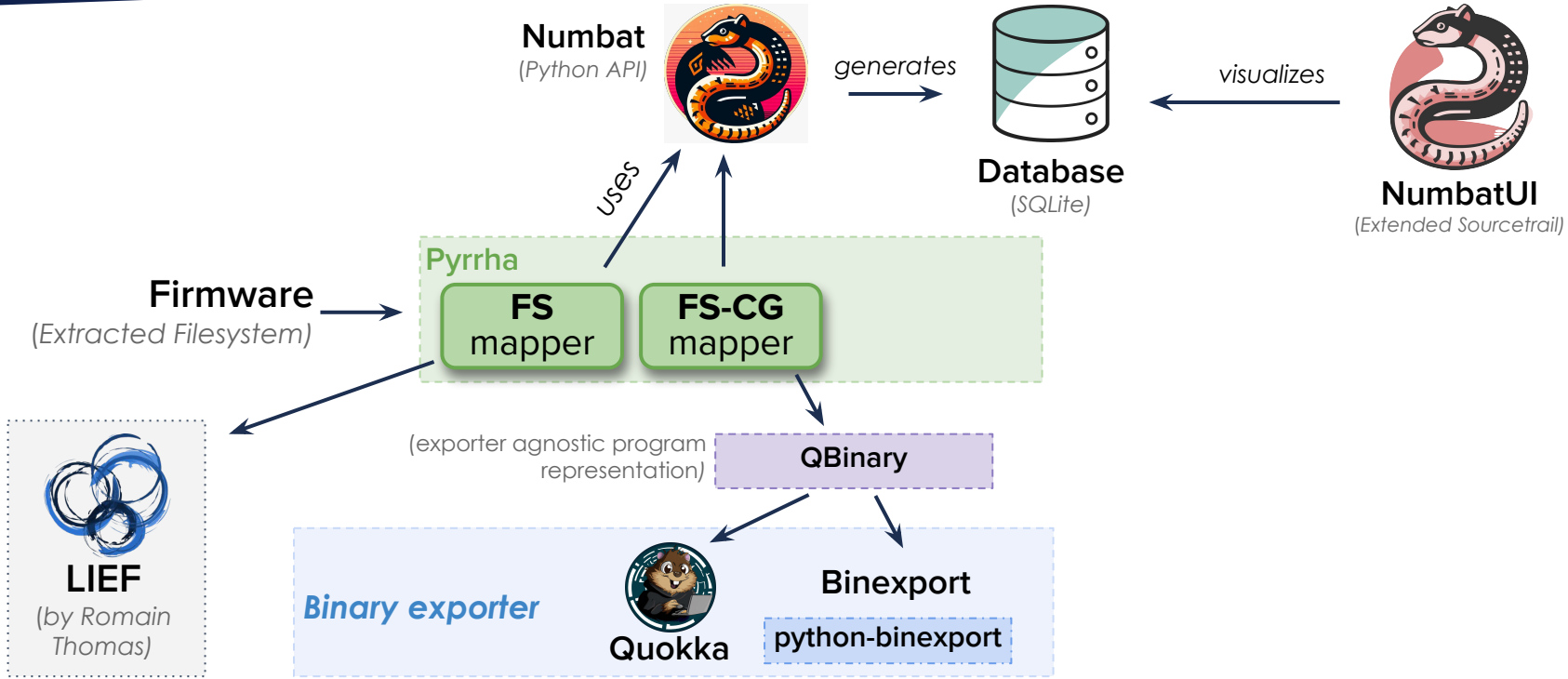
Pyrrha & Friends



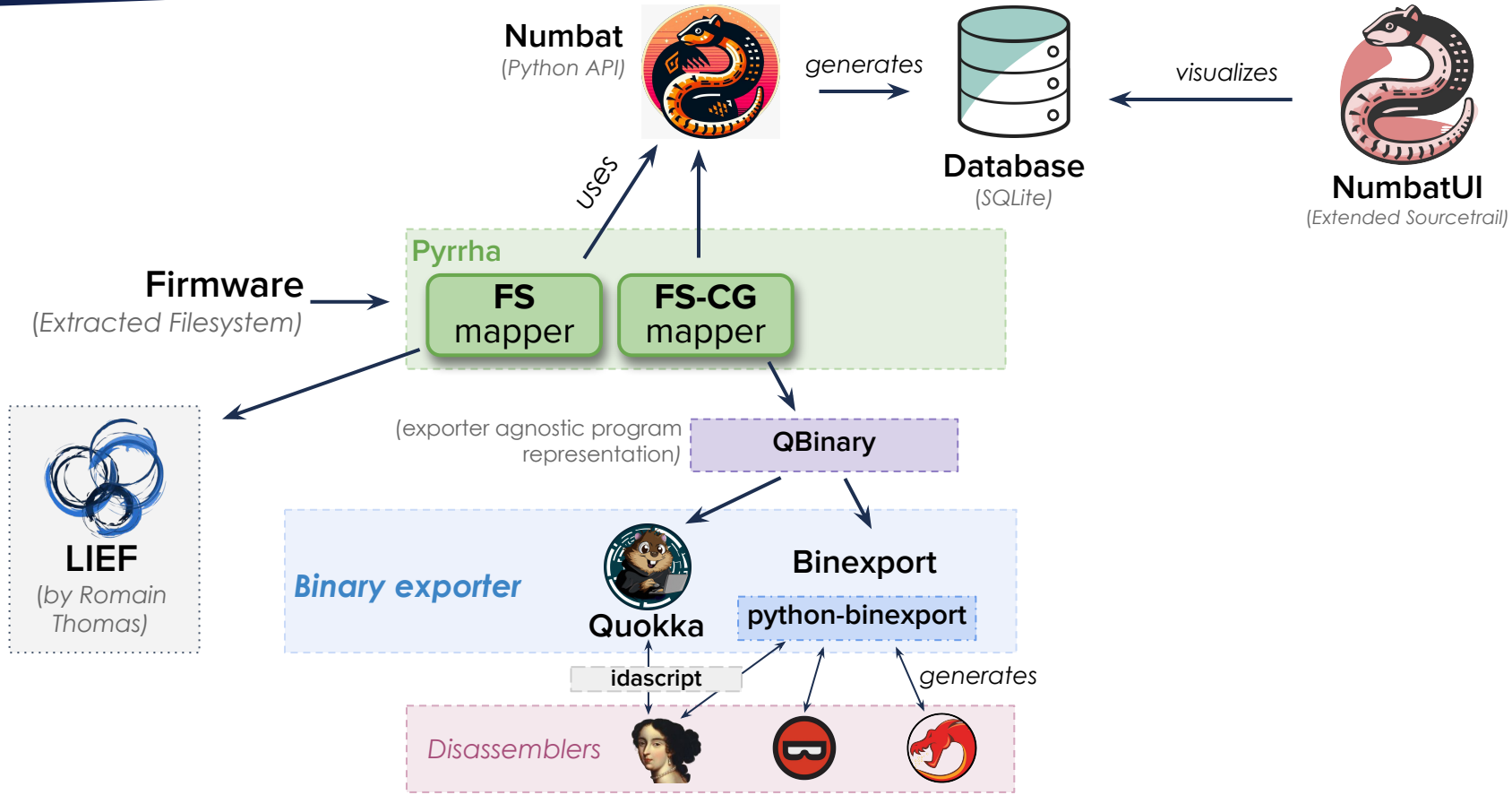
Pyrrha & Friends



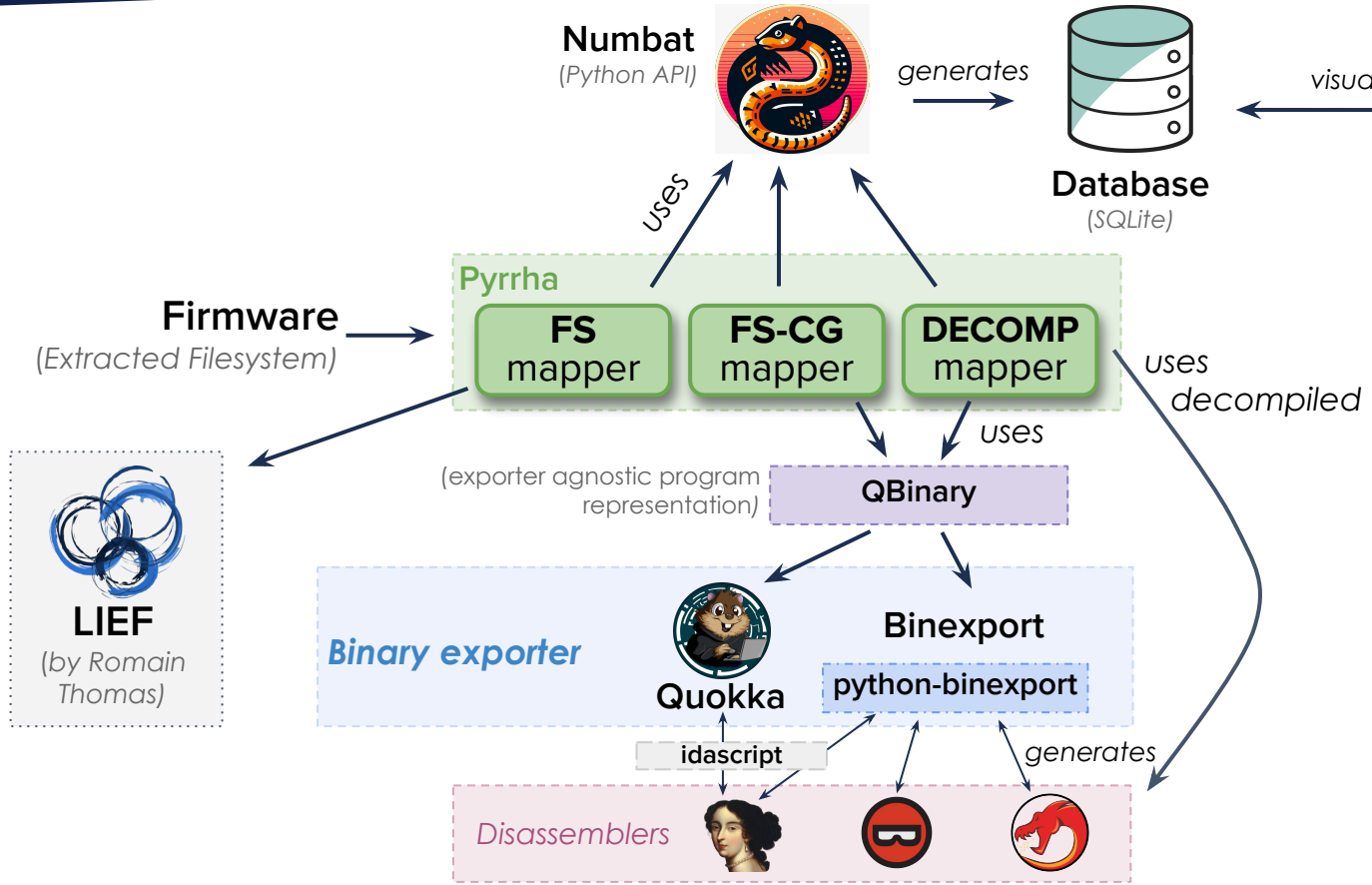
Pyrrha & Friends



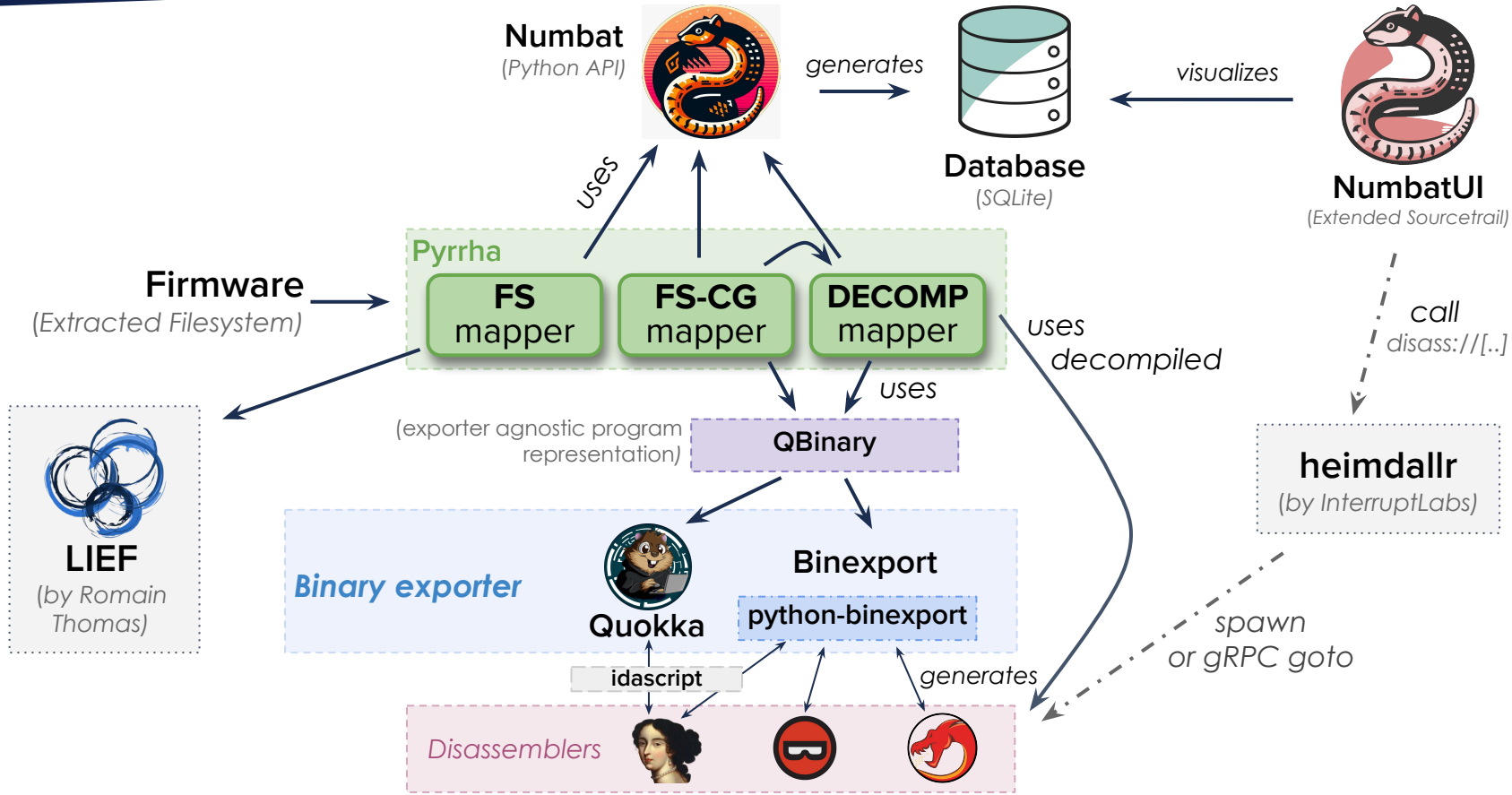
Pyrrha & Friends



Pyrrha & Friends



Pyrrha & Friends



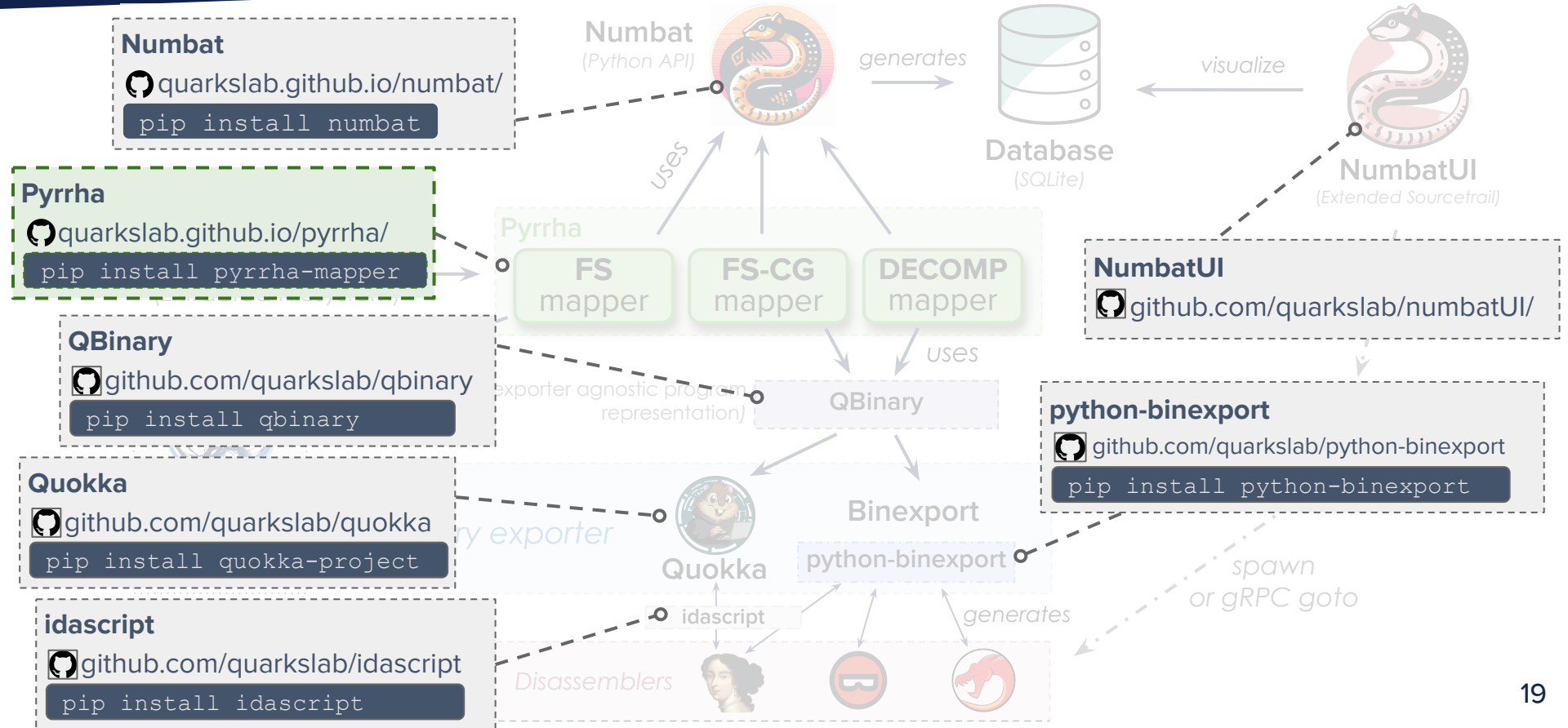
Conclusion / Outro

⇒ **Productivity** tooling for firmware `RECONNAISSANCE` phase

⇒ Can represent **any kind** of data relationship as **nested graphs**
(network interactions, threat-intelligence, interprocess comms)

⇒ *Just scratched the surface of possibilities*
(only focused on executables)

Pyrrha & Friends



Tool

quarkslab.github.io/pyrrha/

Setup

- Install a binary exporter plugin [1↗] [2↗]
- `pip install pyrrha-mapper`
- Install NumbatUI visualizer [↗]

Limitations

- Mostly “Linux” based firmware
- Most suited for C/C++ code
- Heuristics on decompilation xrefs

All Features (branch `dev`)

- Binexport, Ghidra/Binary Ninja
- Sync with disassembler (Heimdallr)

```
pip install
  git+https://github.com/quarkslab/pyrrha.git@dev
```

Special thanks to the contributors:
Sami Babigeon, Riccardo Mori, Pascal Wu

Thank you

Contact information:

Email: contact@quarkslab.com

Phone: +33 1 58 30 81 51

Website: quarkslab.com

@quarkslab

Quarkslab

Benchmarks *(WiFi router)*

Infos

Size	161 Mb
#files	1746
#Executables	111
#Libraries	318
#kernel modules	136

565

	Time			Size	
	Mean	Total *(1 cpu)*	Total *(8 cpu)*	Mean	Total
Disassembly *(.i64)*	25s	4h10m	29min18s	1.8 Mb	1.1 Gb
Quokka	0.77s	438s	68s	460 Kb	255 Mb
Binexport	0.85s	481s	72s	661 Kb	366 Mb
Decompilation	19s	3h03	37m50s	387 Kb	214 Mb
Indexing	1.9s	18m30s	3m28s	1.5 Mb	831 Mb
	Total:	7h38m	1h9m	**Total*:**	1.27 Gb

Favor on-demand creation rather than systematic DB creation