Romain Thomas - rthomas@quarkslab.com

LIEF: Library to Instrument Executable Formats

quarkslab

SECURING EVERY BIT OF YOUR DATA

Table of Contents

Introduction

Architecture

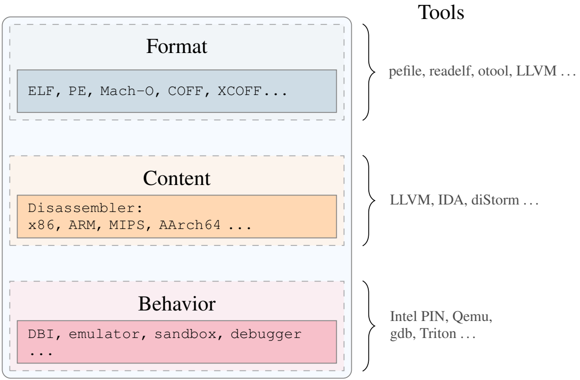
Demo

Conclusion

About

- Romain Thomas - Security engineer at Quarkslab
- Working on obfuscation and software protection, reverse engineering
- Contributor to the Triton project (`https://triton.quarkslab.com`)

Layers of information



Tools

Format

ELF, PE, Mach-O, COFF, XCOFF...

pefile, readelf, otool, LLVM ...

Content

Disassembler:
x86, ARM, MIPS, AArch64 ...

LLVM, IDA, diStorm ...

Behavior

DBI, emulator, sandbox, debugger ...

Intel PIN, Qemu,
gdb, Triton ...

Figure: Layer of information in an executable

Howto?

- Get assembly code?
- Get symbols?
- Get imported functions?

Executable File Formats in a Nutshell

Executable File Formats in a Nutshell

Executable file format gives information such as:

- First instruction address to execute.
- Libraries used
- Target architecture (`x86`, `ARM` . . .)

Executable File Formats in a Nutshell

The three mainstream formats:

- **ELF**: Linux, Android . . .
- **PE**: Windows
- **Mach-O**: OS-X, iOS, . . .

Modification

Format modifications can be a starting point to:

- Packing
- Watermarking
- Hooking: Perform interposition on functions
- Persistent code injection
- Malware analysis (static unpacking . . .)

Purpose of LIEF

- Provide a **cross-platform** library to parse ELF, PE and Mach-O formats
- Abstract common features from the different formats (section, header, entry point, symbols . . .)
- Enable format modifications
- Provide an API for different languages (Python, C++, C . . .)

Howto? (answers)

Get assembly code?

Howto? (answers)

Get assembly code?

```python
import lief
binary = lief.parse("C:\\Windows\\explorer.exe") # PE
asm = binary.get_section(".text")
```

Howto? (answers)

Get symbols?

Howto? (answers)

Get symbols?

```
import lief
binary = lief.parse("/bin/ls") # ELF
for symbol in binary.symbols:
  print(symbols)
```

Howto? (answers)

Get imported functions?

Howto? (answers)

Get imported functions?

```python
import lief
binary = lief.parse("/usr/lib/libc++abi.dylib") # Mach-O
for function in binary.imported_functions:
  print(function)
```

Table of Contents

Introduction

Architecture

Demo

Conclusion

Overview

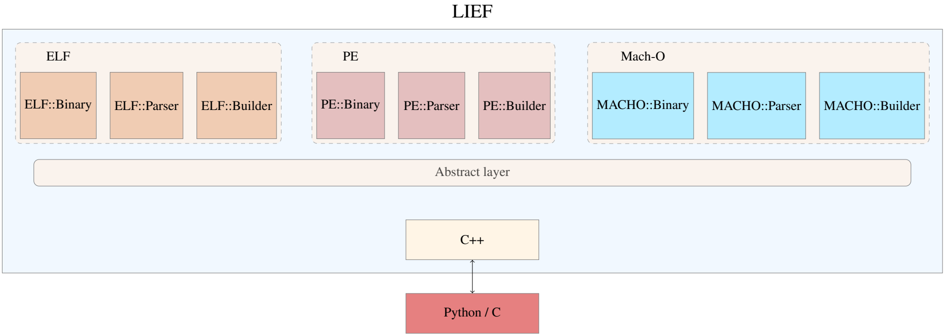


Figure: Global architecture

Modification process

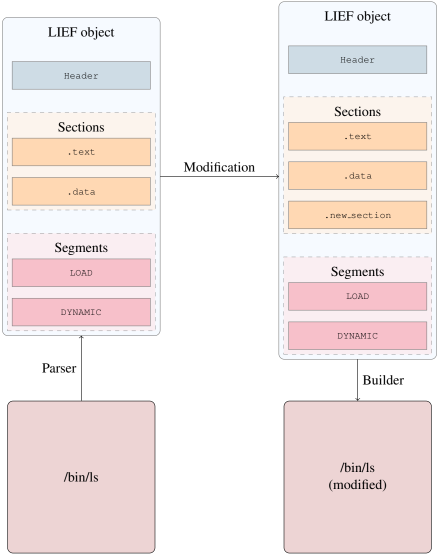


Table of Contents

Introduction

Architecture

Demo

Conclusion

Demo!

Table of Contents

Introduction

Architecture

Demo

Conclusion

Roadmap

Some ideas for next versions:

- Graphical User Interface (Work in progress)
- Handle the `OAT` format (subset of the `ELF` format)
- `PE` API to hook functions
- `PE/Mach-O` fuzzer
- Handle the Dwarf format

- Source code is available on GitHub: `https://github.com/lief-project` (**Apache 2.0** license)
- Website: `https://lief.quarkslab.com`

- Source code is available on GitHub: `https://github.com/lief-project` (**Apache 2.0** license)
- Website: `https://lief.quarkslab.com`

Missing feature or bug?

- Source code is available on GitHub: `https://github.com/lief-project` (**Apache 2.0** license)
- Website: `https://lief.quarkslab.com`

Missing feature or bug?

lief@quarkslab.com
or
Open an issue / pull request

Thank you!

quarkslab

SECURING EVERY BIT OF YOUR DATA