eBPF Summit 2021

eBPF Library Ecosystem Overview

Go, Rust, Python, C and Other Languages

Kyle Quest

The Problem

“

Too many libraries to choose from!

The Libraries

- **C/C++:** bcc, libbpf, ebpfpub
- **Go:** iovisor/gobpf, cilium/ebpf, dropbox/goebpf, libbpfgo
- **Python:** bcc, pyebpf
- **Rust:** libbpf-rs, redbpf, aya
- **Other:** Lua (bcc), Node.js (bpf, bpfcc), Ruby (rbbcc)

A Bit of Background

- **Program Types** (loading and attaching)
- **I/O** (basic operations, additional abstractions)
- **Writing Programs** (helpers, native language support)
- **Compiling Programs** (external, clang integration, native)

Program Types

- **Tracing and Profiling** (kprobes, uprobes, tracepoints, perf events)
- **Networking**
  - **XDP**
  - **TC**
  - **Socket Control and Filtering**
- **Cgroup Resource Control** (more socket control)
- **Security** (LSM)
- Other

C/C++ Libraries

BCC

- Most popular eBPF library (direct/indirect use)
- Program creation abstractions/helpers
- I/O abstractions
- Program compiler abstractions (LLVM/clang runtime dependency)
- Lots of examples and tools
- Biggest community
- Supported program types:
  - **Tracing and profiling** programs
  - **XDP**, **TC**, **Socket Filtering** networking programs
  - **Security** (LSM) programs
- **Verdict**: Good option if you want to leverage the power of BCC and you are building system or XDP/TC-based network tracing apps
- **https://github.com/iovisor/bcc**

libbpf

- Official eBPF library (eBPF linux kernel maintainers)
- Focus on reusable eBPF programs (CO-RE)
- No I/O, program or compiler abstractions
- Template/starter project: **https://github.com/libbpf/libbpf-bootstrap**
- Supported program types (explicit attach support):
    - **Tracing and profiling** programs
    - **XDP** networking programs
    - **Security** (LSM) programs
- Support for generic program attach and link create calls
- **Verdict**: Good option if you want to use the official eBPF library and you are ok using its low level interface and you don’t need the abstractions in BCC
- **https://github.com/libbpf/libbpf**

ebpfpub

- C++ **Libbpf** wrapper library
- Tracing library created for Osquery
- Supported program types:
  - **Tracing and profiling** programs (BPF_PROG_TYPE_TRACEPOINT, BPF_PROG_TYPE_KPROBE only)
- **Verdict**: Haven’t tried it yet, but the code looks clean and it looks like a good abstraction to create tracing apps in C++
- **https://github.com/trailofbits/ebpfpub**
- **https://github.com/trailofbits/ebpf-common**
- **https://www.trailofbits.com/post/all-your-tracing-are-belong-to-bpf**

Go Libraries

iovisor/gobpf

- Official **BCC** wrapper
- Supported program types:
  - **Tracing and profiling** programs
  - **XDP** networking programs
- Partial (load only) support for TC and some Socket Control and Filtering program types
- **Verdict**: Good option if you want to leverage the power of BCC and you are building tracing apps
- **https://github.com/iovisor/gobpf**

cilium/ebpf

- **Pure Go** library
- Initial goal use case - networking (“packet wrangling in XDP and TC”)
- Tracing/profiling wasn’t the initial goal, but now it’s supported
- Strange/clever API (Collections, Specs)
- “asm” - helper library to write eBPF programs (low level instructions)
- “bpf2go” - embed compiled programs in Go code
- You are responsible for attaching many/important networking program types (e.g., XDP, TC)
- Exposes low level / raw attach program (BPF_PROG_ATTACH) and attach link (BPF_LINK_CREATE) interfaces (useful for some prog types)
- Supported program types (explicit “attach” support):
  - Most **tracing and profiling** programs
  - Few other program types (based on the vendor use cases / needs)
- **Verdict**: Interesting library if you want a pure Go library and you are ok with the library design
- **https://github.com/cilium/ebpf**

dropbox/goebpf

- **Pure Go** library
- Focus on networking
- Nice and clean
- Strange/unnecessary use of CGo in some cases
- Supported program types:
  - Basic **tracing and profiling** programs (kprobes/kretprobes only)
  - **XDP** networking programs
  - One of the **Socket filtering** program types (SOCKET_FILTER)
  - **TC** network program types (but only loading, no attach, so doesn’t count :-))
- **Verdict**: Interesting library if it supports the program types you need and if you want to a pure Go library
- **https://github.com/dropbox/goebpf**

libbpfgo

- Thin **libbpf** wrapper
- Focus on tracing and other product use cases for the library vendor
- Supported program types:
    - **Tracing and profiling** programs
    - **Security** (LSM) programs
    - **TC** network program types
- **Verdict**: Good library if it supports the program types you need and if you want to use libbpf in Go
- **https://github.com/aquasecurity/libbpfgo**

Python Libraries

BCC

- Official BCC wrapper
- Program creation helpers
- I/O abstractions
- Most widely used eBPF library
- Lots of examples and python-based tools
- **Verdict**: Use this library if you want to leverage the power of BCC and its community
- **https://github.com/iovisor/bcc/tree/master/src/python/bcc**

pyebpf

- BCC wrapper (with extras)
- Lets you write kprobes and I/O handlers in Python
- Dated (python2 only) and requires extra work
- Supported program types:
    - Only kprobe **tracing and profiling** programs
- **Verdict**: Good library if you are looking for a project to contribute and you want to write kprobe eBPF programs in Python
- **https://pypi.org/project/pyebpf**

Rust Libraries

libbpf-rs

- Lightweight libbpf wrapper (almost official rust library for libbpf :-))
- Needs good examples
- Leverages “auto-attach” from libbpf
- Explicitly supported program/attach types:
  - Most **tracing and profiling** programs (no raw tracepoint support)
  - **XDP** networking programs
  - Security/LSM programs
  - One of the **Socket filtering** program types (SK_SKB/sockmap/streamparser)
- **Verdict**: Good option if you just want to use libbpf directly from Rust.
- **https://github.com/libbpf/libbpf-rs**
- **https://github.com/libbpf/libbpf-bootstrap/tree/master/examples/rust**

redbpf

- libbpf wrapper (partial wrapper, with extras)
- Focus on networking and other product use cases for the library creators
- Supported program types:
  - Some **tracing and profiling** programs (no raw tracepoint support)
  - **XDP** networking programs
  - Two **Socket filtering** program types (SOCKET_FILTER, SK_SKB/sockmap)
- "redbpf-probes" - helper library to generate eBPF programs
- **Verdict**: Interesting library if it supports the program types you need
- **https://github.com/foniod/redbpf**

aya

- **Pure Rust** library
- Supported program types:
  - Most **tracing and profiling** programs (raw tracepoint support is WIP)
  - **XDP** networking programs
  - **TC** classifier programs
  - Several **socket control and filtering** programs (SOCK_FILTER, SK_SKB, SOCK_OPS, SK_MSG)
  - Security/LSM (WIP)
  - Others
- Planned support for rust-based eBPF programs (no clang)
- **Verdict**: Early, but pretty impressive
- **https://github.com/alessandrod/aya**

Other Languages

Lua

- Official bcc wrapper library
- Quite a few examples (some work as-is, some don’t)
- Doesn't get enough attention
- **Verdict**: Be ready to do extra work
- **https://github.com/iovisor/bcc/tree/master/src/lua**
- **https://github.com/iovisor/bcc/tree/master/examples/lua**

Ruby

- BCC wrapper
- Supports most **tracing and profiling** programs
- Quite a few examples
- Requires a specific libbcc version
- **Verdict**: Be ready to do extra work to make it work
- **https://github.com/udzura/rbbcc**

Node.js

- node_bpf - libbpf wrapper
  - Experimental / only a few MAP related functions
- node_bpfcc - bcc wrapper
  - Supports most **tracing and profiling** programs
  - No raw tracepoint support
- Doesn't get enough attention.
- **Verdict**: Cool experiment, but you are on your own if you try to use it.
- **https://github.com/mildsunrise/node_bpf**
- **https://github.com/mildsunrise/node_bpfcc**

Key Takeaways

Thank You

https://twitter.com/kcqon

https://github.com/kcq

Program Type Categories

Tracing and Profiling

- **BPF_PROG_TYPE_KPROBE** (kprobe/kretprobe/uprobe/uretprobe)
- **BPF_PROG_TYPE_PERF_EVENT**
- **BPF_PROG_TYPE_TRACEPOINT**
- **BPF_PROG_TYPE_RAW_TRACEPOINT**
- **BPF_PROG_TYPE_RAW_TRACEPOINT_WRITABLE**

Networking

XDP

- BPF_PROG_TYPE_XDP

Traffic Control (TC)

- BPF_PROG_TYPE_SCHED_CLS
- BPF_PROG_TYPE_SCHED_ACT

Socket Control and Filtering

- BPF_PROG_TYPE_SOCKET_FILTER
- BPF_PROG_TYPE_SOCK_OPS
- BPF_PROG_TYPE_SK_SKB
- BPF_PROG_TYPE_SK_MSG
- BPF_PROG_TYPE_SK_LOOKUP
- BPF_PROG_TYPE_SK_REUSEPORT
- BPF_PROG_TYPE_STRUCT_OPS

Flow Disector

- **BPF_PROG_TYPE_FLOW_DISSECTOR**

Lightweight Tunnel

- BPF_PROG_TYPE_LWT_IN
- BPF_PROG_TYPE_LWT_OUT
- BPF_PROG_TYPE_LWT_XMIT
- BPF_PROG_TYPE_LWT_SEG6LOCAL

Cgroup Resource Control

- BPF_PROG_TYPE_CGROUP_SKB
- BPF_PROG_TYPE_CGROUP_SOCK
- BPF_PROG_TYPE_CGROUP_SOCKOPT
- BPF_PROG_TYPE_CGROUP_SOCK_ADDR
- BPF_PROG_TYPE_CGROUP_SYSCTL
- BPF_PROG_TYPE_CGROUP_DEVICE

Security

- **BPF_PROG_TYPE_LSM**