KUBERMATIC

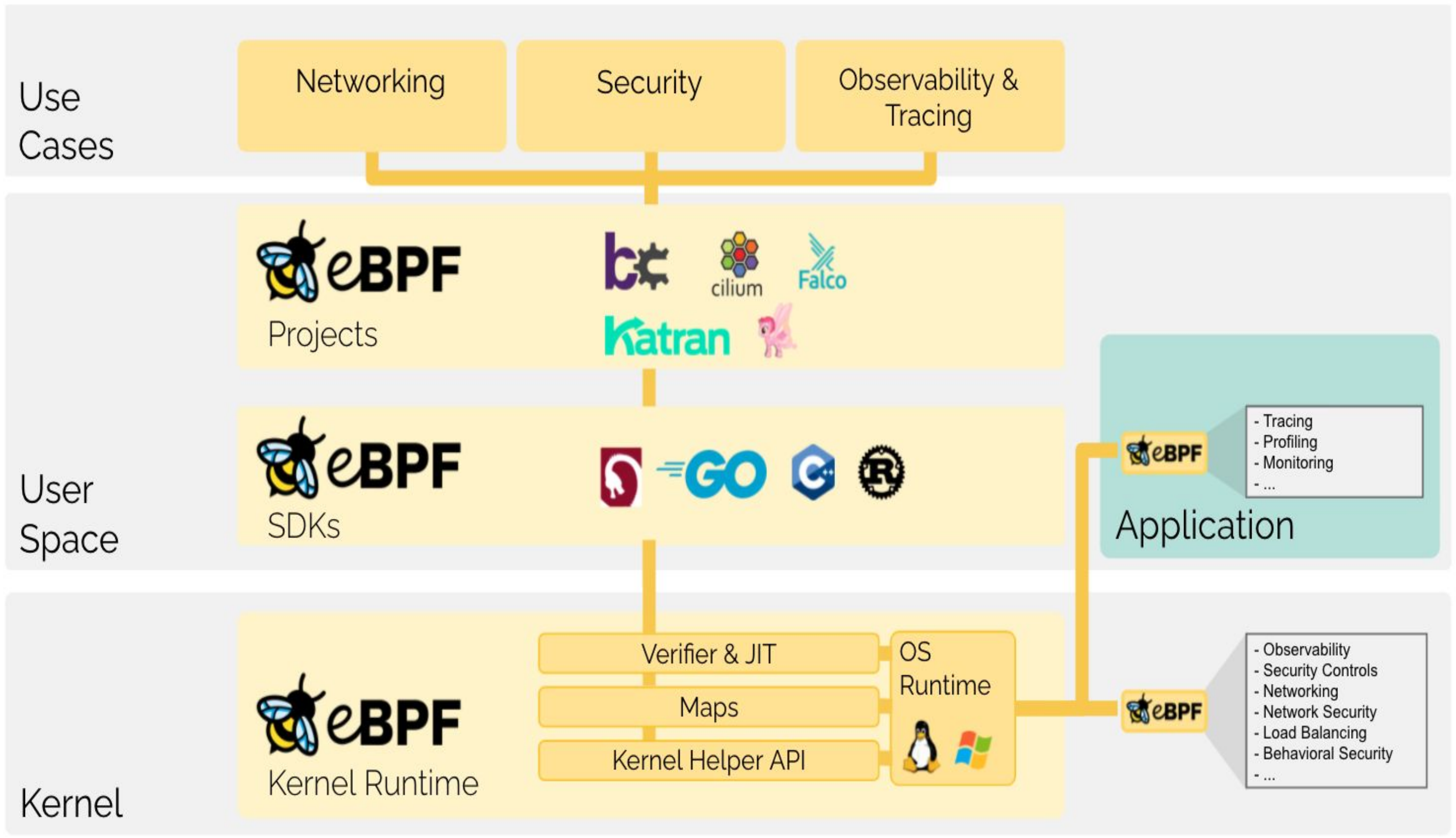
Hackathon - Berlin 2022

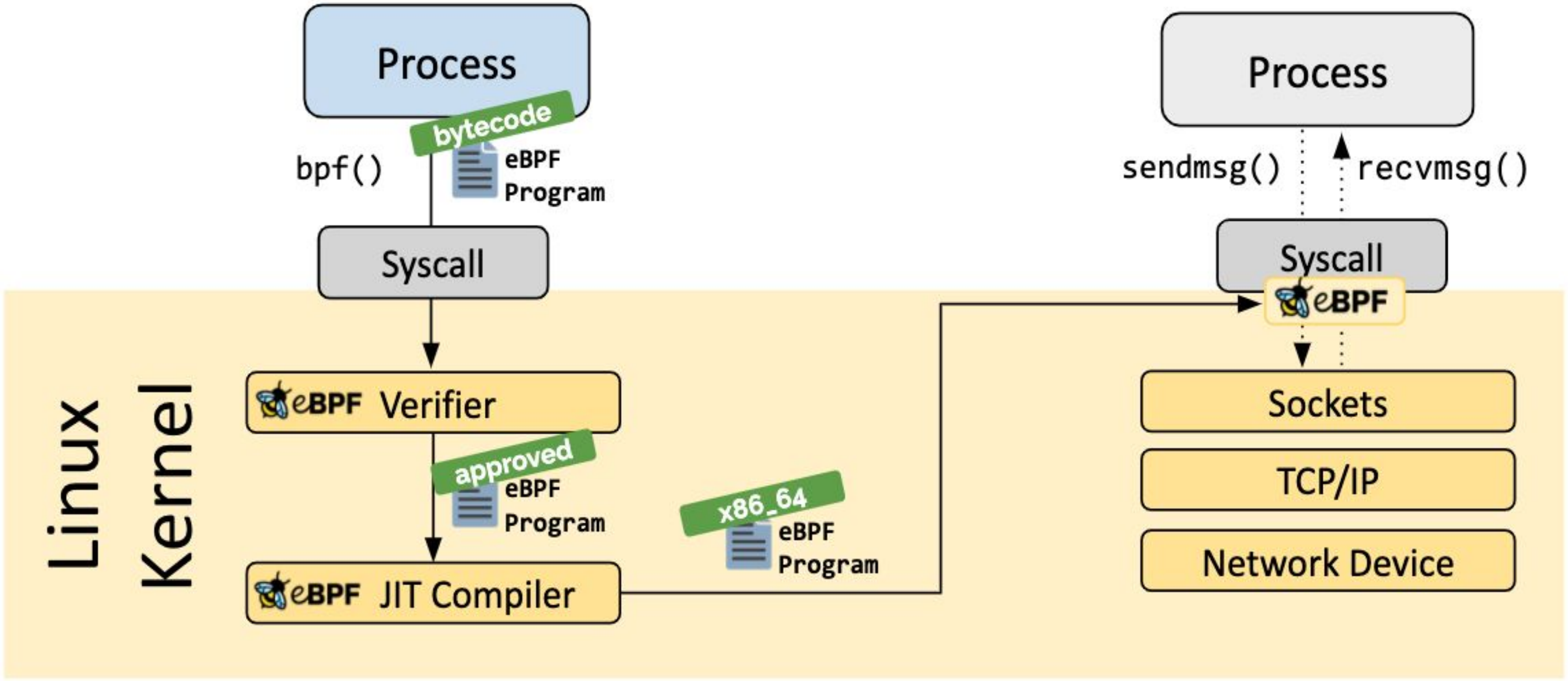
Learn some eBPF for greater good

KUBERMATIC

eBPF

theoretical introduction


Use Cases
Networking
Security
Observability & Tracing
User Space
eBPF
Projects
cilium
Falco
Katran
eBPF
SDKs
GO
Application
eBPF
- Tracing
- Profiling
- Monitoring
- ...
Kernel
eBPF
Kernel Runtime
Verifier & JIT
Maps
Kernel Helper API
OS Runtime
eBPF
- Observability
- Security Controls
- Networking
- Network Security
- Load Balancing
- Behavioral Security
- ...


Process
bytecode
bpf()
eBPF
Program
Syscall
Linux
Kernel
eBPF Verifier
approved
eBPF
Program
eBPF JIT Compiler
x86_64
eBPF
Program
Process
sendmsg()
recvmsg()
Syscall
eBPF
Sockets
TCP/IP
Network Device

KUBERMATIC

uprobe

writing runtime code execution agent from scratch

```go
import (
 "fmt"
 "time"
)

func easyToFindFunctionName(arg uint32) {
  fmt.Println(arg)
}

func main() {
  t1, t2 := time.NewTicker(time.Second * 3), time.NewTicker(time.Second * 5)
  for {
    select {
    case <-t1.C:
      easyToFindFunctionName(1)
    case <-t2.C:
      easyToFindFunctionName(2)
    }
  }
}
```

```go
import (
 "fmt"
 "time"
)

func easyToFindFunctionName(arg uint32) {
  fmt.Println(arg)
}

func main() {
  t1, t2 := time.NewTicker(time.Second * 3), time.NewTicker(time.Second * 5)
  for {
    select {
    case <-t1.C:
      easyToFindFunctionName(1)
    case <-t2.C:
      easyToFindFunctionName(2)
    }
  }
}
```

KUBERMATIC

```go
import (
 "fmt"
 "time"
)

func easyToFindFunctionName(arg uint32) {
  fmt.Println(arg)
}

func main() {
  t1, t2 := time.NewTicker(time.Second * 3), time.NewTicker(time.Second * 5)
  for {
    select {
    case <-t1.C:
      easyToFindFunctionName(1)
    case <-t2.C:
      easyToFindFunctionName(2)
    }
  }
}
```

```go
import (
 "fmt"
 "time"
)

func easyToFindFunctionName(arg uint32) {
  fmt.Println(arg)
}

fun
  t1, t2 := time.NewTicker(time.Second * 3), time.NewTicker(time.Second * 5)
  for {
    select {
    case <-t1.C:
      easyToFindFunctionName(1)
    case <-t2.C:
      easyToFindFunctionName(2)
    }
  }
}
```

```
$ go build -gcflags '-N -l' -o testbin ./main.go
```

```go
import (
  "github.com/cilium/ebpf"
  "github.com/cilium/ebpf/link"
)

func main() {
  // Load ebpf byte code and extract maps and programs
  ebpf.LoadAndAssign(obj, opts)

  // open executable to be instrumented
  ex, err := link.OpenExecutable("/bin/testbin")

  // register uprobe on a particular symbol in a binary
  up, err := ex.Uprobe("easyToFindFunctionName", objs.UprobeTestbinTest)

  // listen on events from kernel through perf event map
  // BPF_MAP_TYPE_PERF_EVENT_ARRAY
  for{
    fmt.Println(ts, event.Pid, symbol, event.Arg)
  }
}
```

```
import (
  "github.com/cilium/ebpf"
  "github.com/cilium/ebpf/link"
)

func main() {
  // Load ebpf byte code and extract maps and programs
  ebpf.LoadAndAssign(obj, opts)

  // open executable to be instrumented
  ex, err := link.OpenExecutable("/bin/testbin")

  // register uprobe on a particular symbol in a binary
  up, err := ex.Uprobe("easyToFindFunctionName", objs.UprobeTestbinTest)

  // listen on events from kernel through perf event map
  // BPF_MAP_TYPE_PERF_EVENT_ARRAY
  for{
    fmt.Println(ts, event.Pid, symbol, event.Arg)
  }
}
```

```
import (
  "github.com/cilium/ebpf"
  "github.com/cilium/ebpf/link"
)

func main() {
  // Load ebpf byte code and extract maps and programs
  ebpf.LoadAndAssign(obj, opts)

  // open executable to be instrumented
  ex, err := link.OpenExecutable("/bin/testbin")

  // register uprobe on a particular symbol in a binary
  up, err := ex.Uprobe("easyToFindFunctionName", objs.UprobeTestbinTest)

  // listen on events from kernel through perf event map
  // BPF_MAP_TYPE_PERF_EVENT_ARRAY
  for{
    fmt.Println(ts, event.Pid, symbol, event.Arg)
  }
}
```

KUBERMATIC

```
import (
  "github.com/cilium/ebpf"
  "github.com/cilium/ebpf/link"
)

func main() {
  // Load ebpf byte code and extract maps and programs
  ebpf.LoadAndAssign(obj, opts)

  // open executable to be instrumented
  ex, err := link.OpenExecutable("/bin/testbin")

  // register uprobe on a particular symbol in a binary
  up, err := ex.Uprobe("easyToFindFunctionName", objs.UprobeTestbinTest)

  // listen on events from kernel through perf event map
  // BPF_MAP_TYPE_PERF_EVENT_ARRAY
  for{
    fmt.Println(ts, event.Pid, symbol, event.Arg)
  }
}
```

```
import (
  "github.com/cilium/ebpf"
  "github.com/cilium/ebpf/link"
)

func main() {
  // Load ebpf byte code and extract maps and programs
  ebpf.LoadAndAssign(obj, opts)

  // open executable to be instrumented
  ex, err := link.OpenExecutable("/bin/testbin")

  // register uprobe on a particular symbol in a binary
  up, err := ex.Uprobe("easyToFindFunctionName", objs.UprobeTestbinTest)

  // listen on events from kernel through perf event map
  // BPF_MAP_TYPE_PERF_EVENT_ARRAY
  for{
    fmt.Println(ts, event.Pid, symbol, event.Arg)
  }
}
```

```c
#include "bpf_tracing.h"

struct event {
  u32 pid;
  u32 arg;
};

struct {
  __uint(type, BPF_MAP_TYPE_PERF_EVENT_ARRAY);
} events SEC(".maps");

SEC("uprobe/testbin_test")
int uprobe_testbin_test(struct pt_regs *ctx) {
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

```c
#include "bpf_tracing.h"

struct event {
  u32 pid;
  u32 arg;
};

struct {
  __uint(type, BPF_MAP_TYPE_PERF_EVENT_ARRAY);
} events SEC(".maps");

SEC("uprobe/testbin_test")
int uprobe_testbin_test(struct pt_regs *ctx) {
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

```c
#include "bpf_tracing.h"

struct event {
  u32 pid;
  u32 arg;
};

struct {
  __uint(type, BPF_MAP_TYPE_PERF_EVENT_ARRAY);
} events SEC(".maps");

SEC("uprobe/testbin_test")
int uprobe_testbin_test(struct pt_regs *ctx) {
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

```
#include "bpf_tracing.h"

struct event {
  u32 pid;
  u32 arg;
};

struct {
  __uint(type, BPF_MAP_TYPE_PERF_EVENT_ARRAY);
} events SEC(".maps");

SEC("uprobe/testbin_test")
int uprobe_testbin_test(struct pt_regs *ctx) {
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

```c
#include "bpf_tracing.h"

struct event {
  u32 pid;
  u32 arg;
};

struct {
  __uint(type, BPF_MAP_TYPE_PERF_EVENT_ARRAY);
} events SEC(".maps");

SEC("uprobe/testbin_test")
int uprobe_testbin_test(struct pt_regs *ctx) {
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

```
#include "bpf_tracing.h"

struct event {
  u32 pid;
  u32 arg;
};

struct {
  __uint(type, BPF_MAP_TYPE_PERF_EVENT_ARRAY);
} events SEC(".maps");

SEC("uprobe/testbin_test")
int uprobe_testbin_test(struct pt_regs *ctx) {
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

Disclaimer

```
#include "bpf_tracing.h"

struct event {
  u32 pid;
  u32 arg;
};

struct {
  __uint(type, BPF_MAP_TYPE_PERF_EVENT_ARRAY);
} events SEC(".maps");

SEC("uprobe/testbin_test")
int uprobe_testbin_test(struct pt_regs *ctx) {
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

```
TEXT main.easyToFindFunctionName(SB) /ebpf-hackathon/uprobe_call_detect/test_bin.go
  test_bin.go:11  0x48e700    493b6610      CMPQ 0x10(R14), SP
  test_bin.go:11  0x48e70a    4883ec58      SUBQ $0x58, SP
  test_bin.go:11  0x48e70e    48896c2450    MOVQ BP, 0x50(SP)
  test_bin.go:12  0x48e76c    488b442420    MOVQ 0x20(SP), AX
...
  test_bin.go:12  0x48e794    e867aaffff    CALL fmt.Println(SB)
...
  test_bin.go:15  0x48e7f4    ebca          JMP main.easyToFindFunctionName(SB)

TEXT main.main(SB) /ebpf-hackathon/uprobe_call_detect/test_bin.go
...
  test_bin.go:27  0x48e8d2    b802000000    MOVL $0x2, 0x18(SP)
  test_bin.go:27  0x48e8d7    e8e4feffff    CALL main.easyToFindFunctionName(SB)
  test_bin.go:26  0x48e8dc    eb0c          JMP 0x48e8ea
  test_bin.go:25  0x48e8de    b801000000    MOVL $0x1, 0x18(SP)
  test_bin.go:25  0x48e8e3    e8d8feffff    CALL main.easyToFindFunctionName(SB)
...
```

```
#incl

struc
  u32
  u32
};

struc
  __u
} eve

SEC("
int up
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

```
TEXT main.easyToFindFunctionName(SB) /ebpf-hackathon/uprobe_call_detect/test_bin.go
  test_bin.go:11  0x48e700    493b6610      CMPQ 0x10(R14), SP
  test_bin.go:11  0x48e70a    4883ec58      SUBQ $0x58, SP
  test_bin.go:11  0x48e70e    48896c2450    MOVQ BP, 0x50(SP)
  test_bin.go:12  0x48e76c    488b442420    MOVQ 0x20(SP), AX
...
  test_bin.go:12  0x48e794    e867aaffff    CALL fmt.Println(SB)
...
  test_bin.go:15  0x48e7f4    ebca          JMP main.easyToFindFunctionName(SB)

TEXT main.main(SB) /ebpf-hackathon/uprobe_call_detect/test_bin.go
...
  test_bin.go:27  0x48e8d2    b802000000    MOVL $0x2, 0x18(SP)
  test_bin.go:27  0x48e8d7    e8e4feffff    CALL main.easyToFindFunctionName(SB)
  test_bin.go:26  0x48e8dc    eb0c          JMP 0x48e8ea
  test_bin.go:25  0x48e8de    b801000000    MOVL $0x1, 0x18(SP)
  test_bin.go:25  0x48e8e3    e8d8feffff    CALL main.easyToFindFunctionName(SB)
...
```

```
#incl

struc
  u32
  u32
};

struc
  __u
} eve

SEC("
int u
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

```
TEXT main.easyToFindFunctionName(SB) /ebpf-hackathon/uprobe_call_detect/test_bin.go
  test_bin.go:11  0x48e700    493b6610      CMPQ 0x10(R14), SP
  test_bin.go:11  0x48e70a    4883ec58      SUBQ $0x58, SP
  test_bin.go:11  0x48e70e    48896c2450    MOVQ BP, 0x50(SP)
  test_bin.go:12  0x48e76c    488b442420    MOVQ 0x20(SP), AX
...
  test_bin.go:12  0x48e794    e867aaffff    CALL fmt.Println(SB)
...
  test_bin.go:15  0x48e7f4    ebca          JMP main.easyToFindFunctionName(SB)

TEXT main.main(SB) /ebpf-hackathon/uprobe_call_detect/test_bin.go
...
  test_bin.go:27  0x48e8d2    b802000000    MOVL $0x2, 0x18(SP)
  test_bin.go:27  0x48e8d7    e8e4feffff    CALL main.easyToFindFunctionName(SB)
  test_bin.go:26  0x48e8dc    eb0c          JMP 0x48e8ea
  test_bin.go:25  0x48e8de    b801000000    MOVL $0x1, 0x18(SP)
  test_bin.go:25  0x48e8e3    e8d8feffff    CALL main.easyToFindFunctionName(SB)
...
```

```
#incl

struc
  u32
  u32
};

struc
  __u
} eve

SEC("
int up
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

```
TEXT main.easyToFindFunctionName(SB) /ebpf-hackathon/uprobe_call_detect/test_bin.go
  test_bin.go:11  0x48e700    493b6610      CMPQ 0x10(R14), SP
  test_bin.go:11  0x48e70a    4883ec58      SUBQ $0x58, SP
  test_bin.go:11  0x48e70e    48896c2450    MOVQ BP, 0x50(SP)
  test_bin.go:12  0x48e76c    488b442420    MOVQ 0x20(SP), AX
...
  test_bin.go:12  0x48e794    e867aaffff    CALL fmt.Println(SB)
...
  test_bin.go:15  0x48e7f4    ebca          JMP main.easyToFindFunctionName(SB)

TEXT main.main(SB) /ebpf-hackathon/uprobe_call_detect/test_bin.go
...
  test_bin.go:27  0x48e8d2    b802000000    MOVL $0x2, 0x18(SP)
  test_bin.go:27  0x48e8d7    e8e4feffff    CALL main.easyToFindFunctionName(SB)
  test_bin.go:26  0x48e8dc    eb0c          JMP 0x48e8ea
  test_bin.go:25  0x48e8de    b801000000    MOVL $0x1, 0x18(SP)
  test_bin.go:25  0x48e8e3    e8d8feffff    CALL main.easyToFindFunctionName(SB)
...
```

```
#inclu

struc
  u32
  u32
};

struc
  __u
} eve

SEC("
int up
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

KUBERMATIC

```
#incl

struc
  u32
  u32
};

struc
  _
} e

SEC("
int up
  struct event event;
  bpf_probe_read(&event.arg, sizeof(event.arg), (void*)PT_REGS_SP(ctx)+8);
  if (event.arg == 2) {
    event.pid = bpf_get_current_pid_tgid();
    bpf_perf_event_output(ctx, &events, BPF_F_CURRENT_CPU, &event, sizeof(event));
  }
  return 0;
}
```

```
TEXT main.easyToFindFunctionName(SB) /ebpf-hackathon/uprobe_call_detect/test_bin.go
  test_bin.go:11  0x48e700    493b6610      CMPQ 0x10(R14), SP
  test_bin.go:11  0x48e70a    4883ec58      SUBQ $0x58, SP
  test_bin.go:11  0x48e70e    48896c2450    MOVQ BP, 0x50(SP)
  test_bin.go:12  0x48e76c    488b442420    MOVQ 0x20(SP), AX
...
  test_bin.go:12  0x48e794    e867aaffff    CALL fmt.Println(SB)
...
  test_bin.go:15  0x48e7f4    ebca          JMP main.easyToFindFunctionName(SB)

TEXT main.main(SB) /ebpf-hackathon/uprobe_call_detect/test_bin.go
  test_bin.go:25  0x48e8de    b801000000    MOVL $0x1, 0x18(SP)
  test_bin.go:25  0x48e8e3    e8d8feffff    CALL main.easyToFindFunctionName(SB)
...
```

```
$ go build -gcflags '-N -l' -o testbin ./main.go
$ k get deployments
NAME       READY    UP-TO-DATE     AVAILABLE    AGE
testbin    2/2      2              2            41h

$ k get daemonsets
NAME       DESIRED    CURRENT    READY    UP-TO-DATE     AVAILABLE    NODE SELECTOR    AGE
tracer     9          9          0        9              0            <none>           41h

$ k get pods
NAME                           READY    STATUS     NODE
testbin-764b7bbdd9-h8kcx       1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
testbin-764b7bbdd9-nt5hp       1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
tracer-7jctq                   1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
...
```

```
$ k get pods
NAME                           READY   STATUS    NODE
testbin-764b7bbdd9-h8kcx       1/1     Running   gke-kubermatic-dev-cbd686cd-ck6t
testbin-764b7bbdd9-nt5hp       1/1     Running   gke-kubermatic-dev-cbd686cd-ck6t
tracer-7jctq                   1/1     Running   gke-kubermatic-dev-cbd686cd-ck6t
...


$ k logs tracer-7jctq
2022/10/14 07:44:58 instrumenting
/host/io.containerd.snapshotter.v1.overlayfs/snapshots/265/fs/bin/testbin
2022/10/14 07:44:59 Listening for events..
2022/10/14 07:45:01 10536 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:02 10594 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:04 10536 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:05 10594 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:09 10594 main.easyToFindFunctionName argument: 2
```

```
$ k get pods
NAME                         READY   STATUS    NODE
testbin-764b7bbdd9-h8kcx     1/1     Running   gke-kubermatic-dev-cbd686cd-ck6t
testbin-764b7bbdd9-nt5hp     1/1     Running   gke-kubermatic-dev-cbd686cd-ck6t
tracer-7jctq                 1/1     Running   gke-kubermatic-dev-cbd686cd-ck6t
...


$ k logs tracer-7jctq
2022/10/14 07:44:58 instrumenting
/host/io.containerd.snapshotter.v1.overlayfs/snapshots/265/fs/bin/testbin
2022/10/14 07:44:59 Listening for events..
2022/10/14 07:45:01 10536 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:02 10594 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:04 10536 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:05 10594 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:09 10594 main.easyToFindFunctionName argument: 2
```

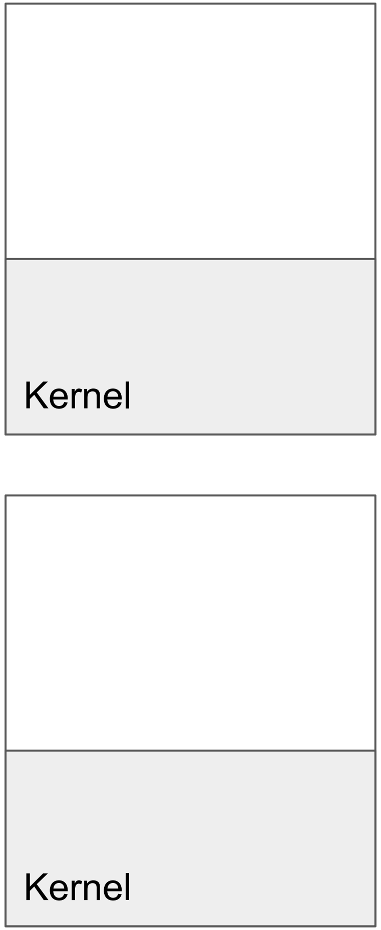
```
$ k get pods
NAME                           READY    STATUS     NODE
testbin-764b7bbdd9-h8kcx       1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
testbin-764b7bbdd9-nt5hp       1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
tracer-7jctq                   1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
...


$ k logs tracer-7jctq
2022/10/14 07:44:58 instrumenting
/host/io.containerd.snapshotter.v1.overlayfs/snapshots/265/fs/bin/testbin
2022/10/14 07:44:59 Listening for events..
2022/10/14 07:45:01 10536 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:02 10594 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:04 10536 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:05 10594 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:09 10594 main.easyToFindFunctionName argument: 2
```

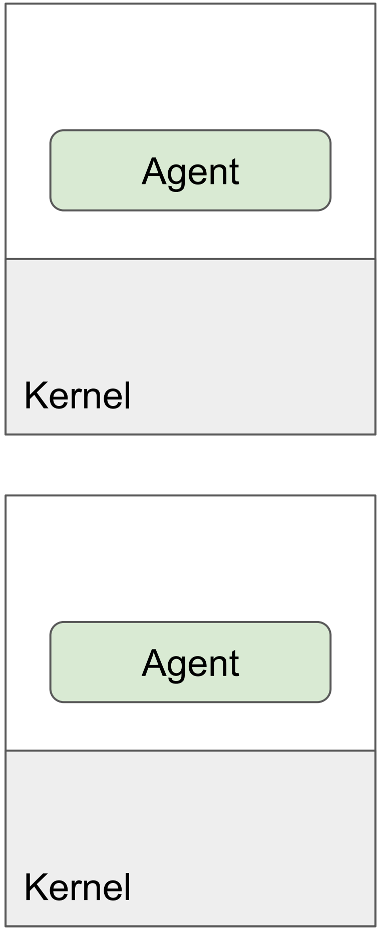
```
$ k get pods
NAME                         READY    STATUS     NODE
testbin-764b7bbdd9-h8kcx     1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
testbin-764b7bbdd9-nt5hp     1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
tracer-7jctq                 1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
...


$ k logs tracer-7jctq
2022/10/14 07:44:58 instrumenting
/host/io.containerd.snapshotter.v1.overlayfs/snapshots/265/fs/bin/testbin
2022/10/14 07:44:59 Listening for events..
2022/10/14 07:45:01 10536 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:02 10594 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:04 10536 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:05 10594 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:09 10594 main.easyToFindFunctionName argument: 2
```

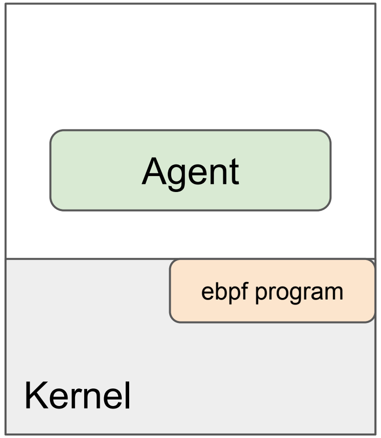
```
$ k get pods
NAME                         READY    STATUS     NODE
testbin-764b7bbdd9-h8kcx     1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
testbin-764b7bbdd9-nt5hp     1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
tracer-7jctq                 1/1      Running    gke-kubermatic-dev-cbd686cd-ck6t
...


$ k logs tracer-7jctq
2022/10/14 07:44:58 instrumenting
/host/io.containerd.snapshotter.v1.overlayfs/snapshots/265/fs/bin/testbin
2022/10/14 07:44:59 Listening for events..
2022/10/14 07:45:01 10536 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:02 10594 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:04 10536 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:05 10594 main.easyToFindFunctionName argument: 2
2022/10/14 07:45:09 10594 main.easyToFindFunctionName argument: 2
```

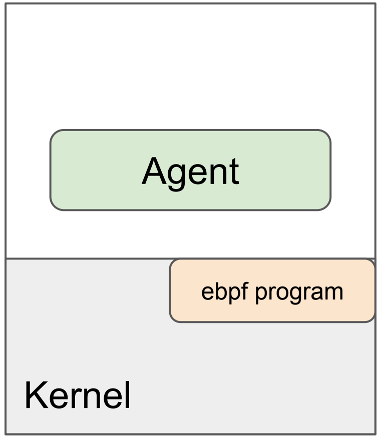
KUBERMATIC

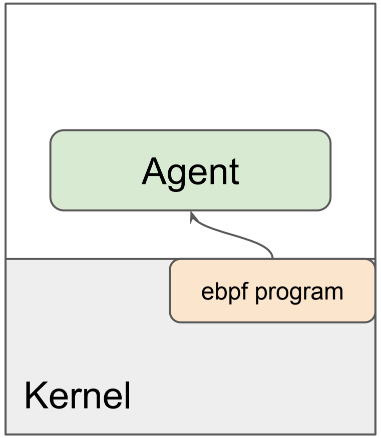
ebpf & Kubernetes

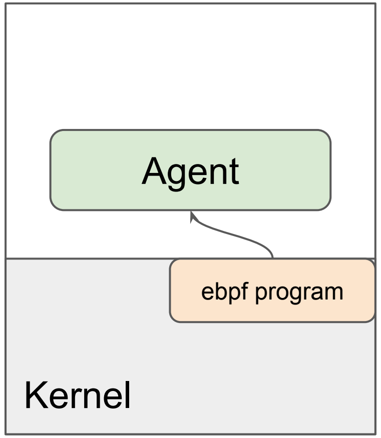

Kernel
Kernel

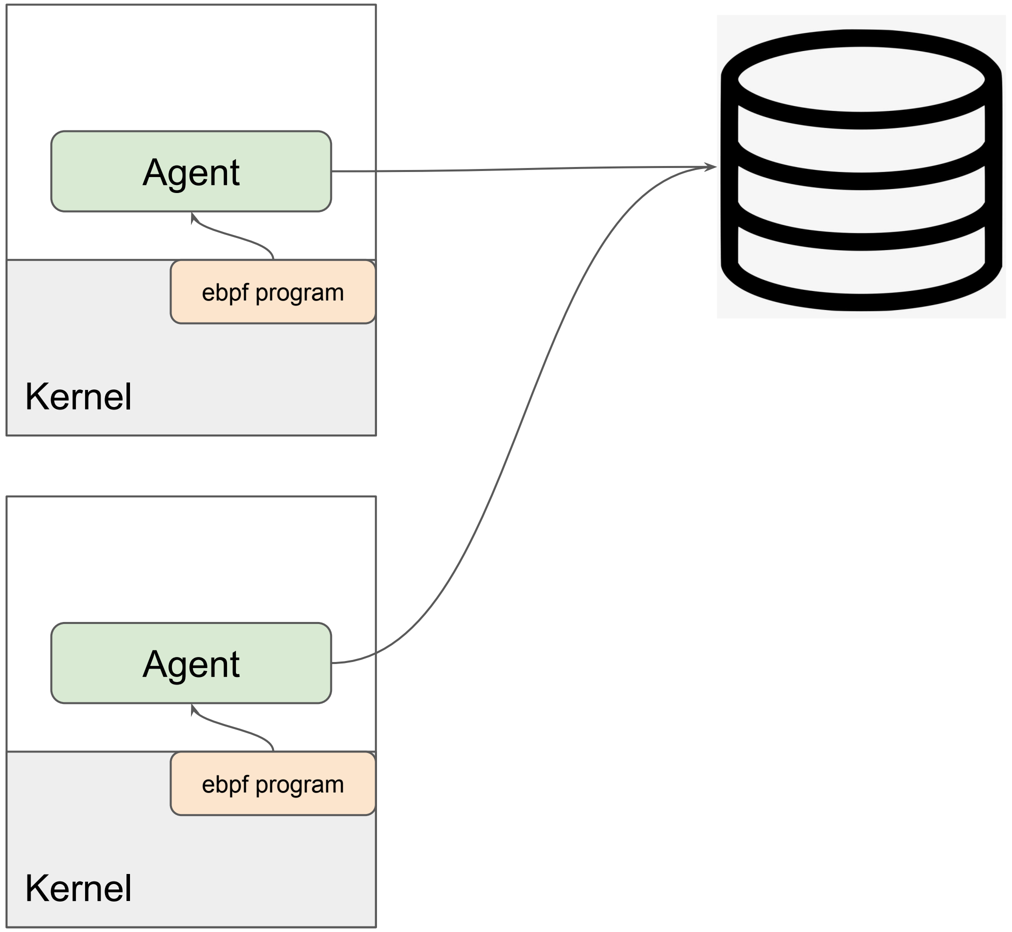

Agent
Kernel
Agent
Kernel

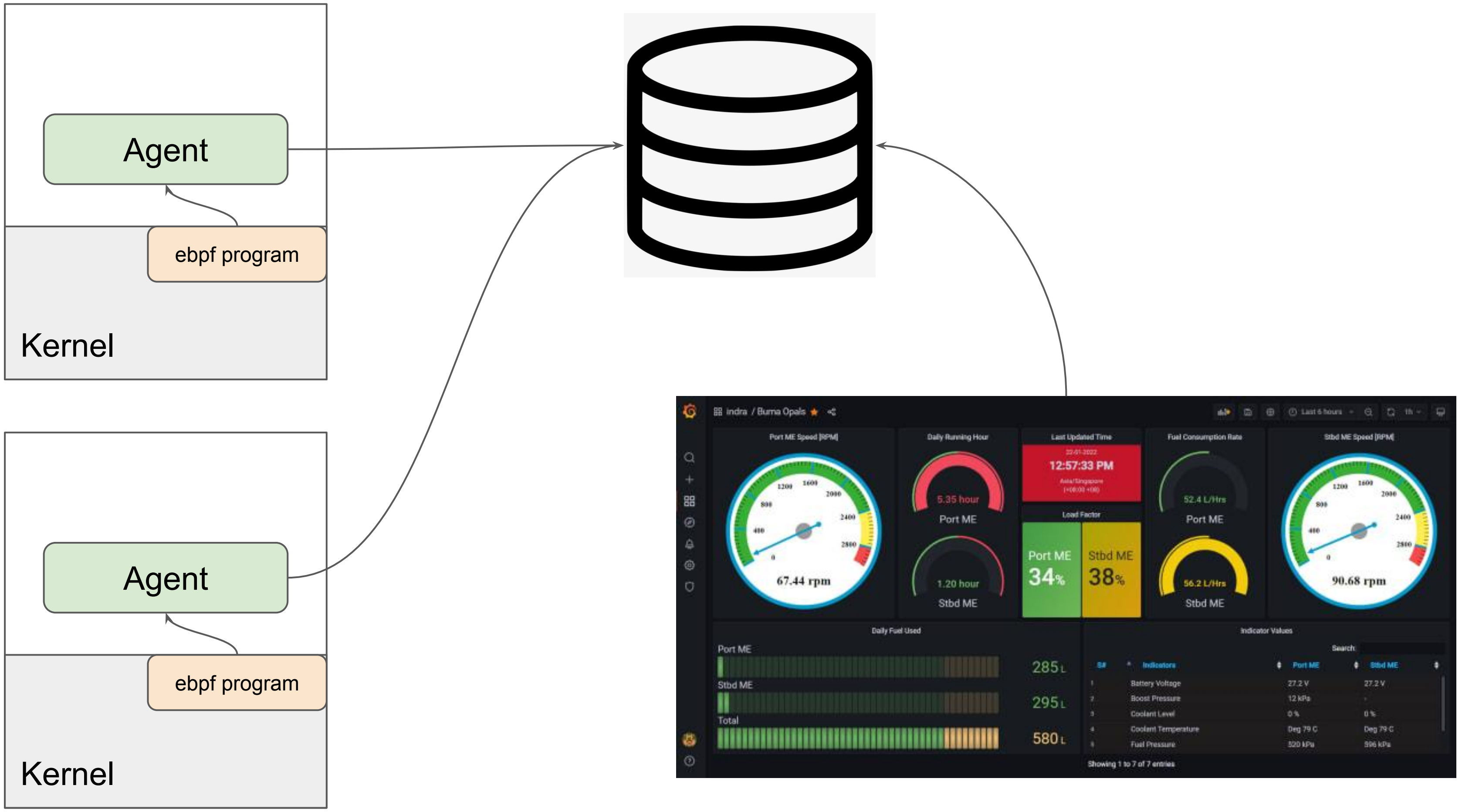

Agent
ebpf program
Kernel


Agent
ebpf program
Kernel


Agent
ebpf program
Kernel


Agent
ebpf program
Kernel


Agent
ebpf program
Kernel
Agent
ebpf program
Kernel

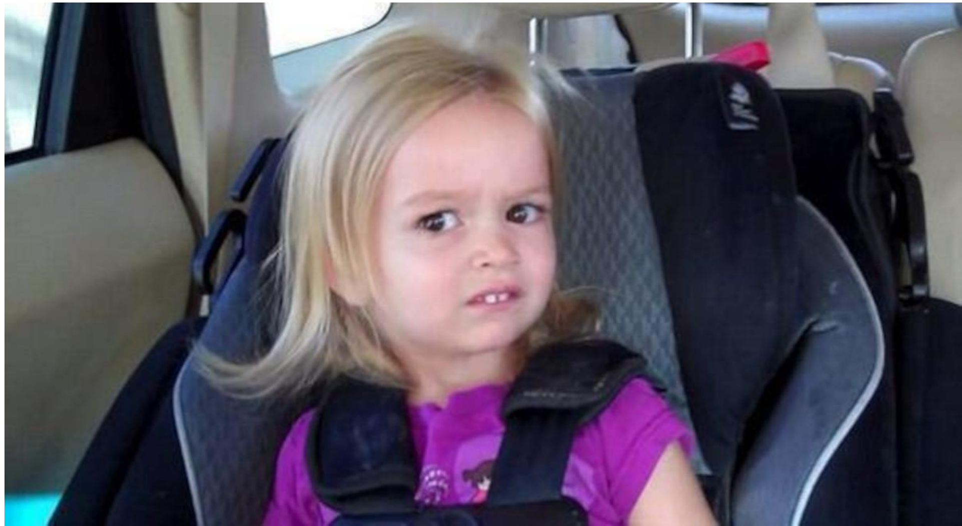


S#	Indicators	Port ME	Stbd ME
1	Battery Voltage	27.2 V	27.2 V
2	Boost Pressure	12 kPa	-
3	Coolant Level	0 %	0 %
4	Coolant Temperature	Deg 79 C	Deg 79 C
5	Fuel Pressure	520 kPa	596 kPa

Showing 1 to 7 of 7 entries

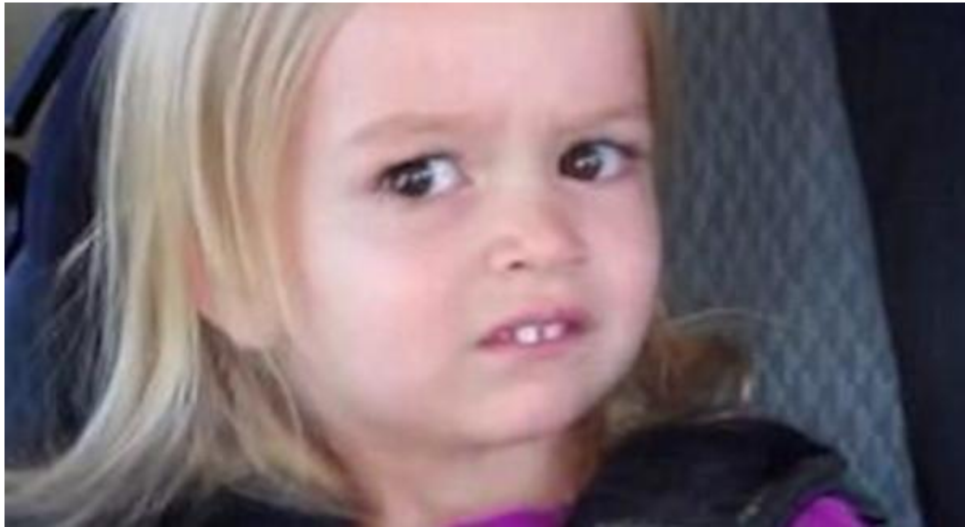
Security?

```
hostPID: true
```



Security?

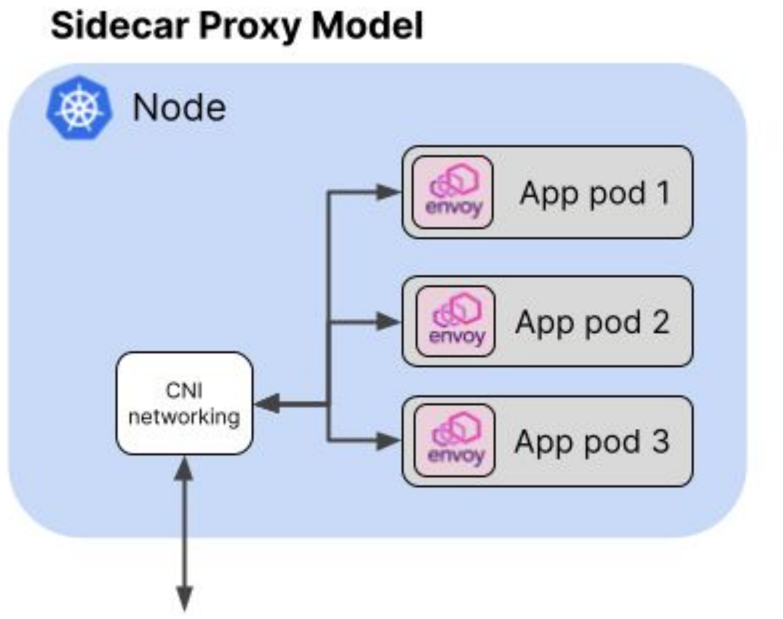
```
securityContext:
    privileged: true
```

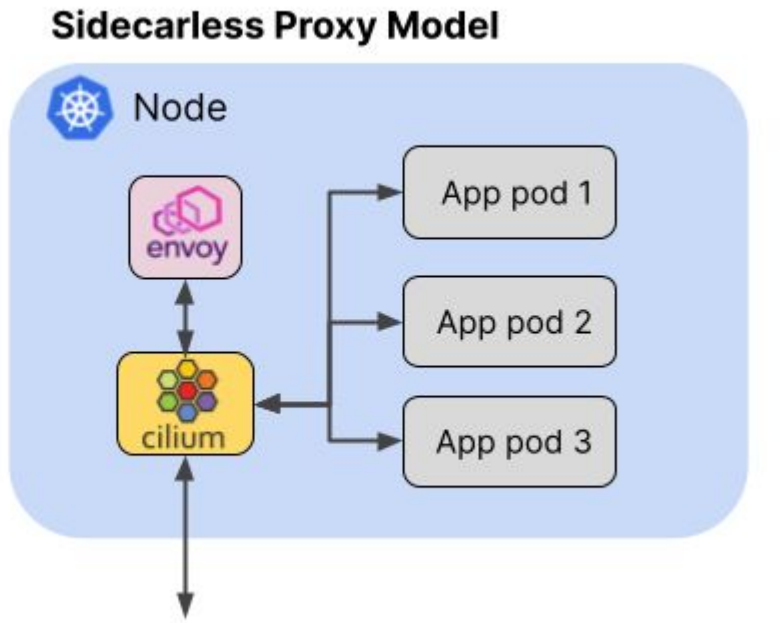


Security?

```
volumes:
- hostPath:
    path: /run
  name: run
- hostPath:
    path: /sys/fs/cgroup
  name: cgroup
- hostPath:
    path: /lib/modules
  name: modules
- hostPath:
    path: /sys/fs/bpf
  name: bpffs
- hostPath:
    path: /sys/kernel/debug
  name: debugfs
```

ServiceMesh?





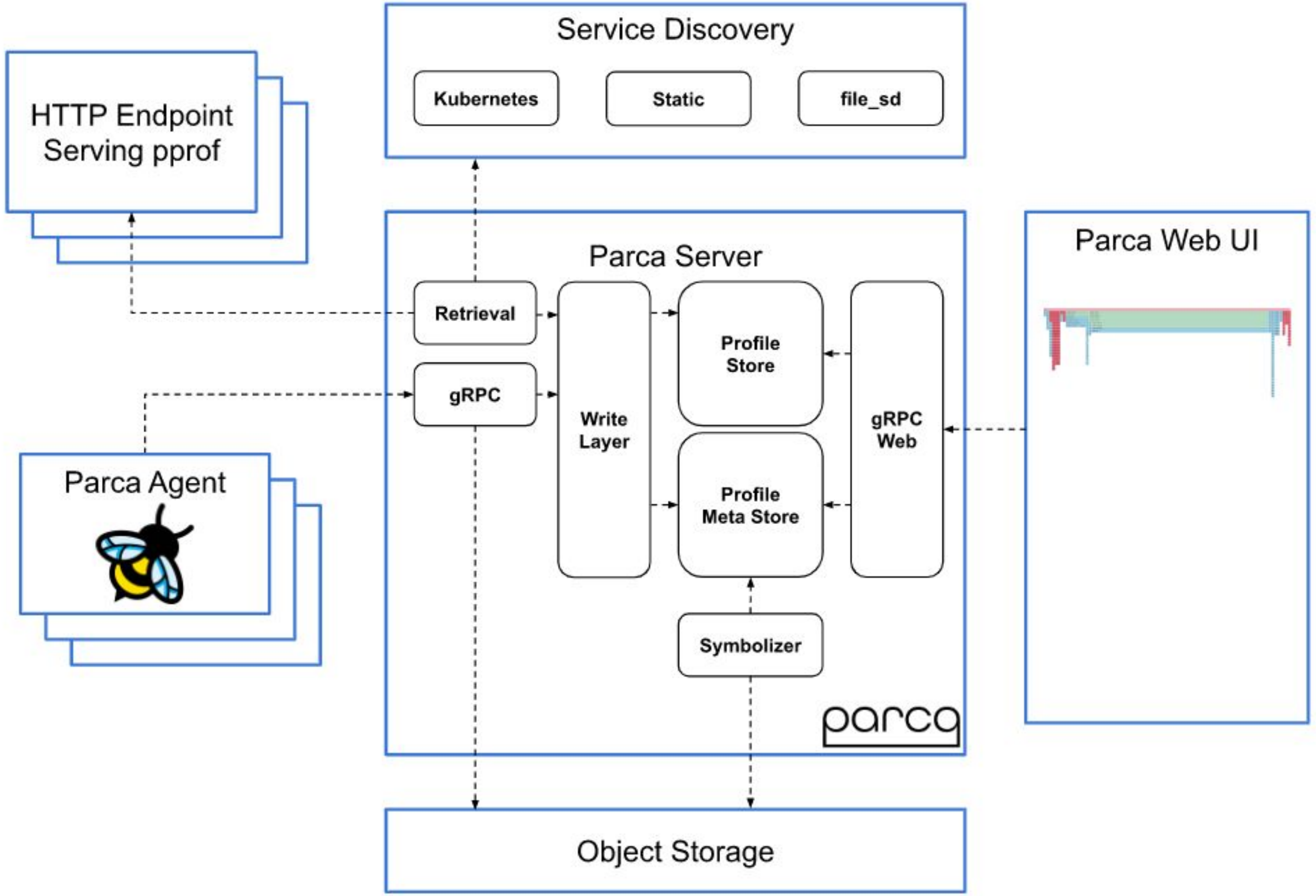
Parca

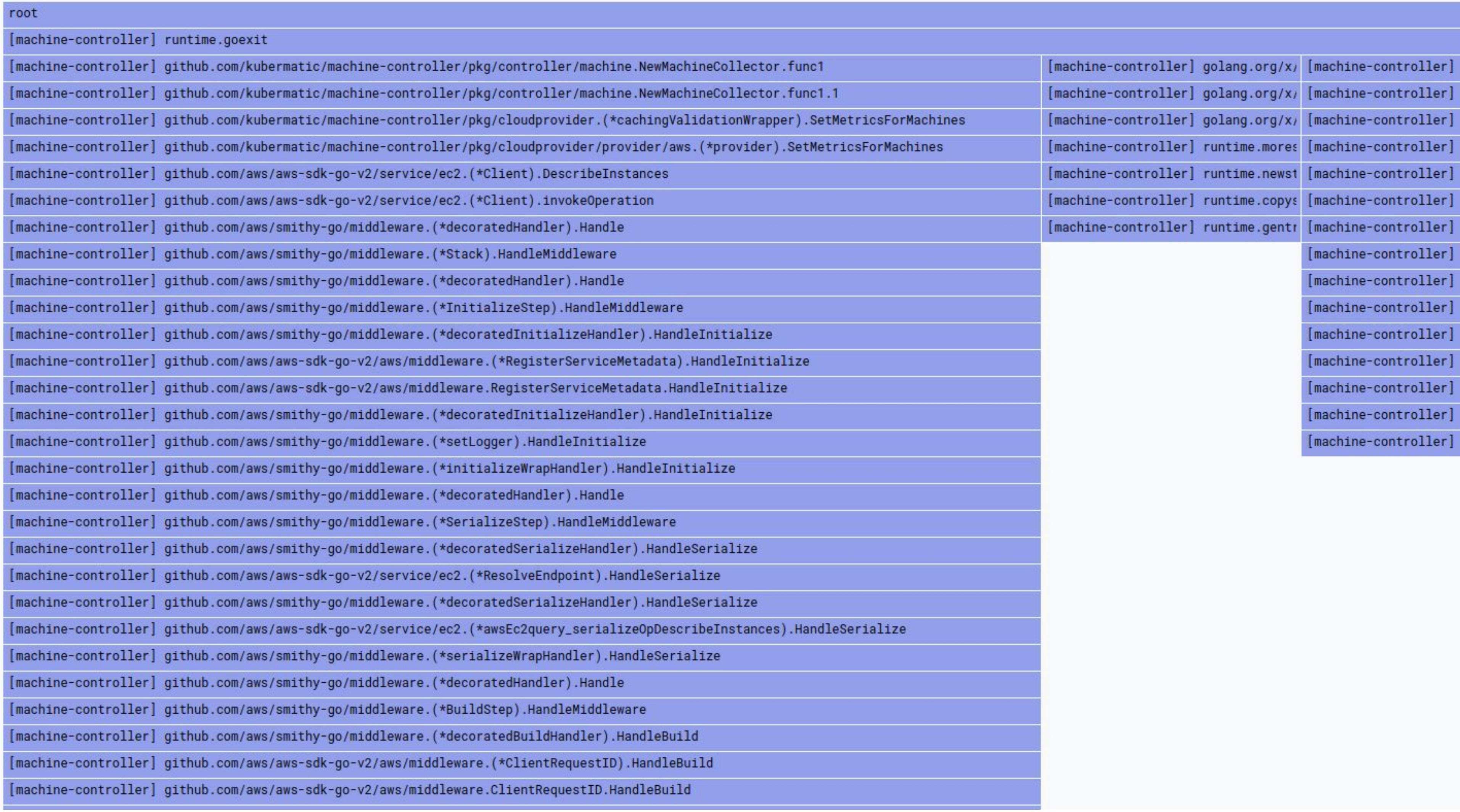
What is Parca?

Continuous profiling for analysis of CPU, memory usage over time, and down to the line number, via ebpf.

Why?

- Save Money
- Improve Performance
- Understand Incidents


Service Discovery
Kubernetes
Static
file_sd
HTTP Endpoint Serving pprof
Parca Server
Retrieval
gRPC
Write Layer
Profile Store
Profile Meta Store
gRPC Web
Symbolizer
parca
Parca Agent
Parca Web UI
Object Storage



```
root
[machine-controller] runtime.goexit
[machine-controller] github.com/kubermatic/machine-controller/pkg/controller/machine.NewMachineCollector.func1          [machine-controller] golang.org/x/  [machine-controller]
[machine-controller] github.com/kubermatic/machine-controller/pkg/controller/machine.NewMachineCollector.func1.1        [machine-controller] golang.org/x/  [machine-controller]
[machine-controller] github.com/kubermatic/machine-controller/pkg/cloudprovider.(*cachingValidationWrapper).SetMetricsForMachines  [machine-controller] golang.org/x/  [machine-controller]
[machine-controller] github.com/kubermatic/machine-controller/pkg/cloudprovider/provider/aws.(*provider).SetMetricsForMachines  [machine-controller] runtime.mores  [machine-controller]
[machine-controller] github.com/aws/aws-sdk-go-v2/service/ec2.(*Client).DescribeInstances                             [machine-controller] runtime.newst  [machine-controller]
[machine-controller] github.com/aws/aws-sdk-go-v2/service/ec2.(*Client).invokeOperation                               [machine-controller] runtime.copys  [machine-controller]
[machine-controller] github.com/aws/smithy-go/middleware.(*decoratedHandler).Handle                                   [machine-controller] runtime.gentr  [machine-controller]
[machine-controller] github.com/aws/smithy-go/middleware.(*Stack).HandleMiddleware                                                                          [machine-controller]
[machine-controller] github.com/aws/smithy-go/middleware.(*decoratedHandler).Handle                                                                         [machine-controller]
[machine-controller] github.com/aws/smithy-go/middleware.(*InitializeStep).HandleMiddleware                                                                 [machine-controller]
[machine-controller] github.com/aws/smithy-go/middleware.(*decoratedInitializeHandler).HandleInitialize                                                     [machine-controller]
[machine-controller] github.com/aws/aws-sdk-go-v2/aws/middleware.(*RegisterServiceMetadata).HandleInitialize                                                [machine-controller]
[machine-controller] github.com/aws/aws-sdk-go-v2/aws/middleware.RegisterServiceMetadata.HandleInitialize                                                   [machine-controller]
[machine-controller] github.com/aws/smithy-go/middleware.(*decoratedInitializeHandler).HandleInitialize                                                     [machine-controller]
[machine-controller] github.com/aws/smithy-go/middleware.(*setLogger).HandleInitialize                                                                      [machine-controller]
[machine-controller] github.com/aws/smithy-go/middleware.(*initializeWrapHandler).HandleInitialize
[machine-controller] github.com/aws/smithy-go/middleware.(*decoratedHandler).Handle
[machine-controller] github.com/aws/smithy-go/middleware.(*SerializeStep).HandleMiddleware
[machine-controller] github.com/aws/smithy-go/middleware.(*decoratedSerializeHandler).HandleSerialize
[machine-controller] github.com/aws/aws-sdk-go-v2/service/ec2.(*ResolveEndpoint).HandleSerialize
[machine-controller] github.com/aws/smithy-go/middleware.(*decoratedSerializeHandler).HandleSerialize
[machine-controller] github.com/aws/aws-sdk-go-v2/service/ec2.(*awsEc2query_serializeOpDescribeInstances).HandleSerialize
[machine-controller] github.com/aws/smithy-go/middleware.(*serializeWrapHandler).HandleSerialize
[machine-controller] github.com/aws/smithy-go/middleware.(*decoratedHandler).Handle
[machine-controller] github.com/aws/smithy-go/middleware.(*BuildStep).HandleMiddleware
[machine-controller] github.com/aws/smithy-go/middleware.(*decoratedBuildHandler).HandleBuild
[machine-controller] github.com/aws/aws-sdk-go-v2/aws/middleware.(*ClientRequestID).HandleBuild
[machine-controller] github.com/aws/aws-sdk-go-v2/aws/middleware.ClientRequestID.HandleBuild
```

Our conclusions

Positive Points

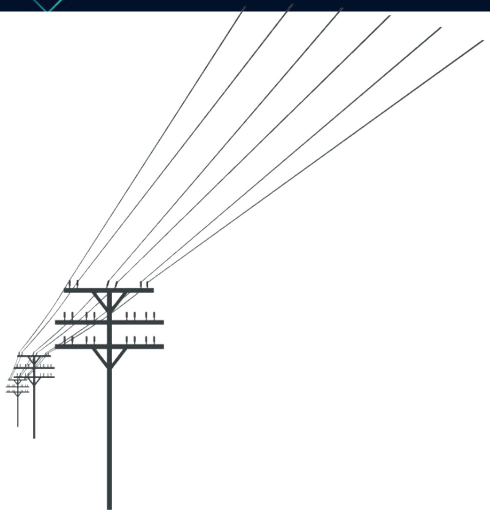
- Does not require code instrumentation
- Easy to install
- Overhead is little - we did not crash DEV env
- Nice UI

Negative Points

- UI/server buggy???
- Interpretation of Icicle Graphs is not intuitive
- Currently only CPU metrics supported
- Installation via DaemonSet probably not doable on some projects because of security

KUBERMATIC

Kepler

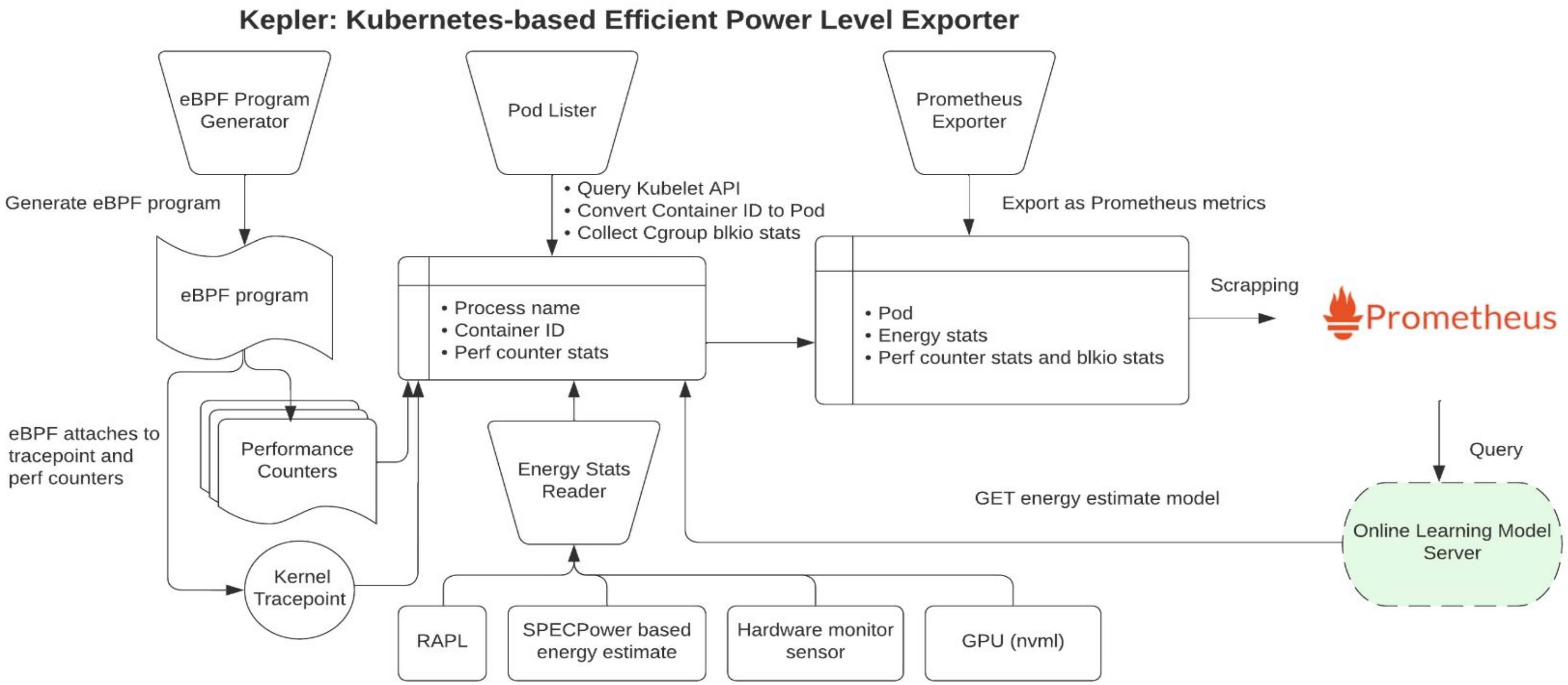


What is Kepler?

Kepler uses eBPF to probe energy related system stats and exports as Prometheus metrics

Why?

- Energy will get more expensive
- Because we can ;)


Kepler: Kubernetes-based Efficient Power Level Exporter
eBPF Program Generator
Generate eBPF program
eBPF program
eBPF attaches to tracepoint and perf counters
Performance Counters
Kernel Tracepoint
Pod Lister
• Query Kubelet API
• Convert Container ID to Pod
• Collect Cgroup blkio stats
• Process name
• Container ID
• Perf counter stats
Energy Stats Reader
RAPL
SPECPower based energy estimate
Hardware monitor sensor
GPU (nvml)
Prometheus Exporter
Export as Prometheus metrics
• Pod
• Energy stats
• Perf counter stats and blkio stats
Scrapping
Prometheus
Query
GET energy estimate model
Online Learning Model Server

Our conclusions

Positive Points

- Prometheus Metric Scheme looks promising
- Autoscaling based on energy costs?

Negative Points

- It does not work on
  - Ubuntu
  - Rocky Linux
  - CentOS
  - FlatCar
  - RHEL
  - Amazon Linux
- Some kernel module was missing, after installation it still did not work (github issue)