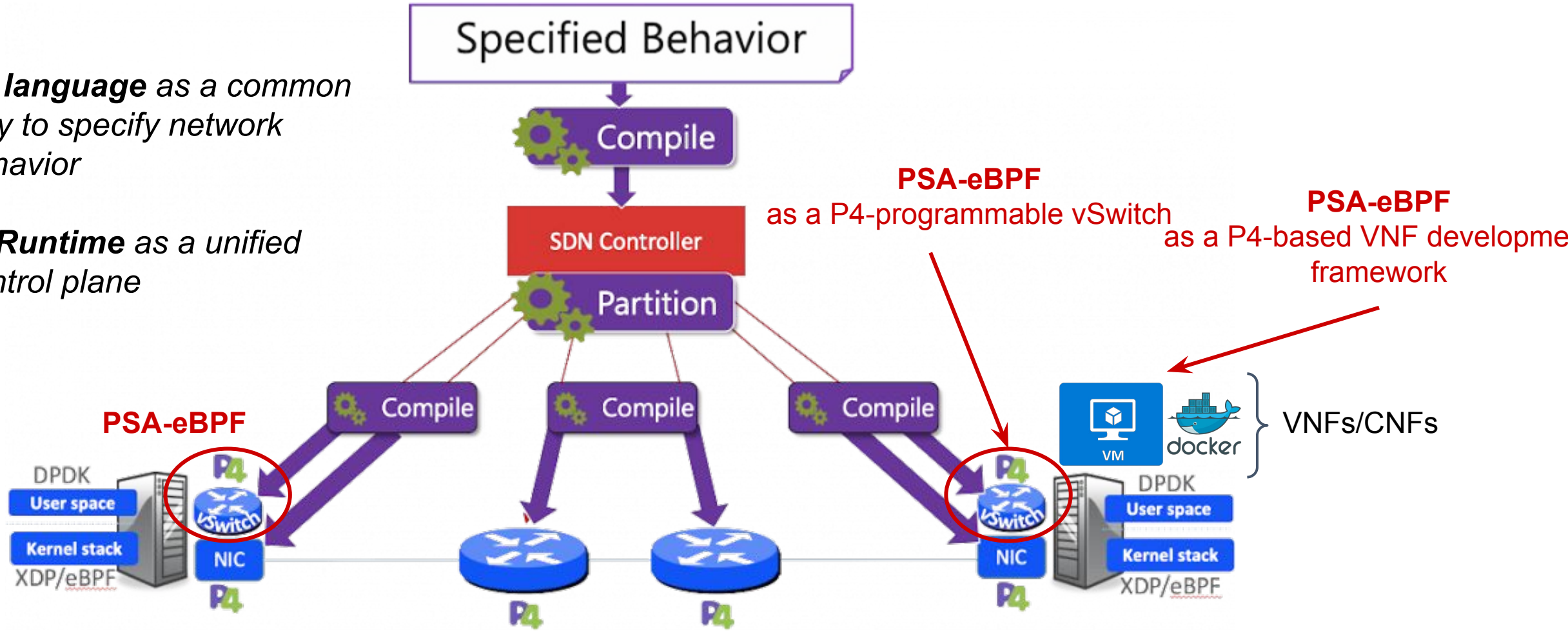
PSA-eBPF: Portable Switch Architecture for eBPF

Tomasz Osiński (Intel, ex-ONF),
Mateusz Kossakowski, Jan Palimąka (Orange)

PSA-eBPF in the E2E programmable platform

P4 language *as a common way to specify network behavior*

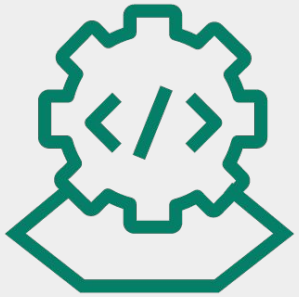
P4Runtime *as a unified control plane*



Announcing PSA-eBPF

- New extension to the eBPF backend of the open-source P4 compiler!
- Feature-rich Portable Switch Architecture (PSA) enables more use cases!
- psabpf API + psabpf-ctl CLI tool to load & manage PSA/eBPF programs

PSA-eBPF compiler

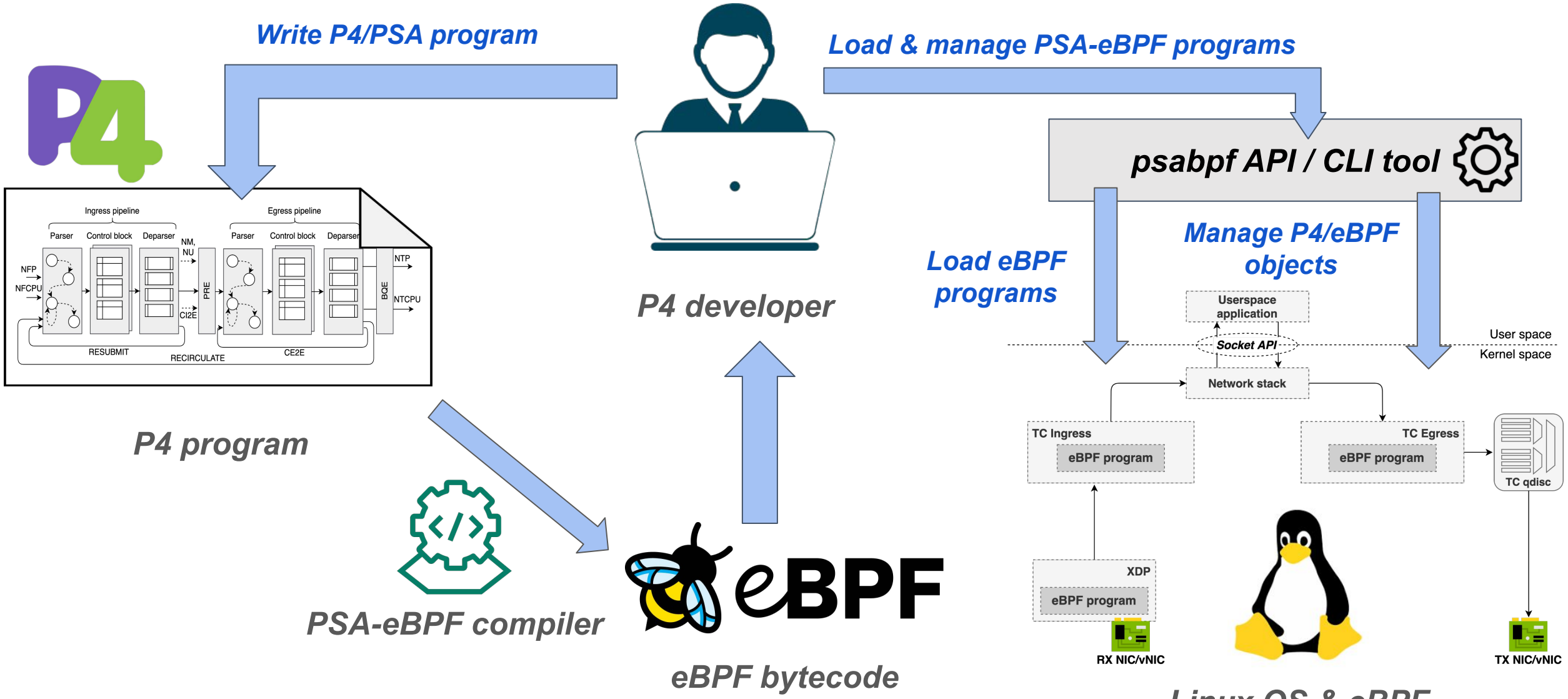


psabpf API / CLI tool

github.com/p4lang/p4c/tree/main/backends/ebpf/psa

github.com/P4-Research/psabpf

Overview of the PSA-eBPF workflow



PSA-eBPF in nutshell: near complete PSA impl.

- P4c -eBPF backend extended with support for PSA
  - uses a combination of eBPF data types, primitives & TC/XDP hooks

PSA-eBPF in nutshell: near complete PSA impl.

- P4c -eBPF backend extended with support for PSA
  - uses a combination of eBPF data types, primitives & TC/XDP hooks
- All PSA packet paths verified, designed & implemented
  - e.g., normal unicast, normal multicast, clone sessions (CI2E, CE2E), recirculation, resubmission

PSA-eBPF in nutshell: near complete PSA impl.

- P4c -eBPF backend extended with support for PSA
  - uses a combination of eBPF data types, primitives & TC/XDP hooks
- All PSA packet paths verified, designed & implemented
  - e.g., normal unicast, normal multicast, clone sessions (CI2E, CE2E), recirculation, resubmission
- Developed (almost) all match kinds for P4 tables
  - **exact & lpm** implemented by basic eBPF primitives (BPF hash/LPM_TRIE)
  - **ternary** matching implemented by adopting Tuple Space Search (TSS)
  - **range** not supported yet

PSA-eBPF in nutshell: near complete PSA impl.

- P4c -eBPF backend extended with support for PSA
  - uses a combination of eBPF data types, primitives & TC/XDP hooks
- All PSA packet paths verified, designed & implemented
  - e.g., normal unicast, normal multicast, clone sessions (CI2E, CE2E), recirculation, resubmission
- Developed (almost) all match kinds for P4 tables
  - **exact & lpm** implemented by basic eBPF primitives (BPF hash/LPM_TRIE)
  - **ternary** matching implemented by adopting Tuple Space Search (TSS)
  - **range** not supported yet
- All PSA externs designed & implemented
  - Counter, DirectCounter, Meter, DirectMeter, Register, Hash, Checksum, Internet Checksum, Digest, Random, Action Profile, Action Selector

PSA-eBPF in nutshell: near complete PSA impl.

- P4c -eBPF backend extended with support for PSA
  - uses a combination of eBPF data types, primitives & TC/XDP hooks
- All PSA packet paths verified, designed & implemented
  - e.g., normal unicast, normal multicast, clone sessions (CI2E, CE2E), recirculation, resubmission
- Developed (almost) all match kinds for P4 tables
  - **exact & lpm** implemented by basic eBPF primitives (BPF hash/LPM_TRIE)
  - **ternary** matching implemented by adopting Tuple Space Search (TSS)
  - **range** not supported yet
- All PSA externs designed & implemented
  - Counter, DirectCounter, Meter, DirectMeter, Register, Hash, Checksum, Internet Checksum, Digest, Random, Action Profile, Action Selector
- PTF test coverage
  - each feature covered by PTF test; currently ~70 PTF tests running as pre-merge job

PSA-eBPF in nutshell: near complete PSA impl.

- P4c -eBPF backend extended with support for PSA
  - uses a combination of eBPF data types, primitives & TC/XDP hooks
- All PSA packet paths verified, designed & implemented
  - e.g., normal unicast, normal multicast, clone sessions (CI2E, CE2E), recirculation, resubmission
- Developed (almost) all match kinds for P4 tables
  - **exact & lpm** implemented by basic eBPF primitives (BPF hash/LPM_TRIE)
  - **ternary** matching implemented by adopting Tuple Space Search (TSS)
  - **range** not supported yet
- All PSA externs designed & implemented
  - Counter, DirectCounter, Meter, DirectMeter, Register, Hash, Checksum, Internet Checksum, Digest, Random, Action Profile, Action Selector
- PTF test coverage
  - each feature covered by PTF test; currently ~70 PTF tests running as pre-merge job
- psabpf C API + CLI tool
  - low-level C API to be used by control plane stack (e.g., P4Runtime)

Next steps

- Learn more:
  - Watch the P4 workshop tutorial:
    “Deep dive & Getting started with PSA implementation for eBPF”
  - Visit the PSA-eBPF documentation site
- Start playing with PSA-eBPF:
  - Re-produce the PSA-eBPF demo implementing a basic load balancer, rate limiter & QoS
  - Run PSA-eBPF with Mininet
- Feel free to open Github issues to report a bug or ask questions!
- Reach out to us on #p4-ebpf channel in the P4 Lang Slack

Thank You

tomasz.osinski@intel.com / osinstom@gmail.com
mateusz.kossakowski@orange.com
jan.palimaka@orange.com