Building a P4 target with BMv2

Radostin Stoyanov

PhD Candidate - Computing Infrastructure Group

Supervisor: Prof. Noa Zilberman and Prof. Wes Armour

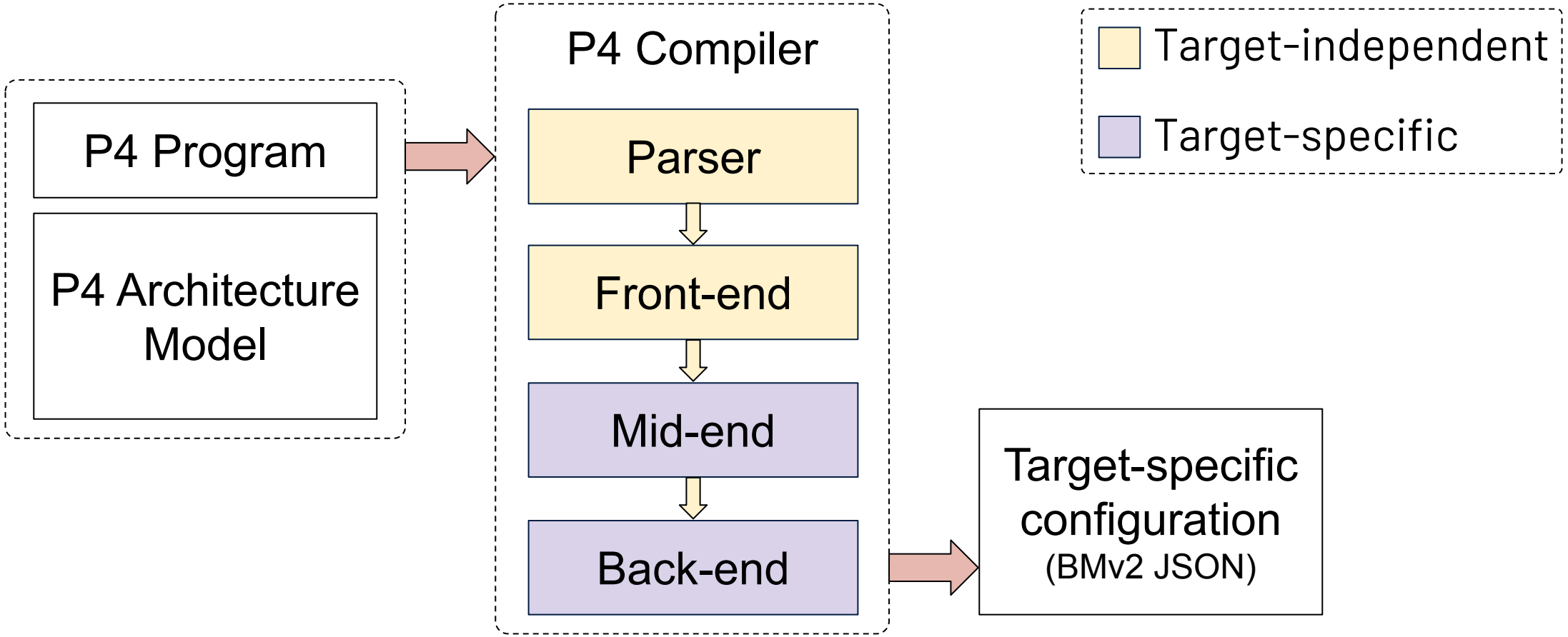
P4.org Open Source Developer Days - March 15, 2022

P4 Architecture

- **Programmable Blocks** - Parser, Pipeline, Deparser, etc.
- **Packet Paths** - Unicast I2E, Recirculate, Resubmit, etc.
- **Types** - `PortId_t`, `MulticastGroup_t`, etc.
- **Externs** - Counter, Meter, Checksum, etc.

"What is a P4 architecture?", Andy Fingerhut, PNA meetings, February 28, 2022

P4 Compiler

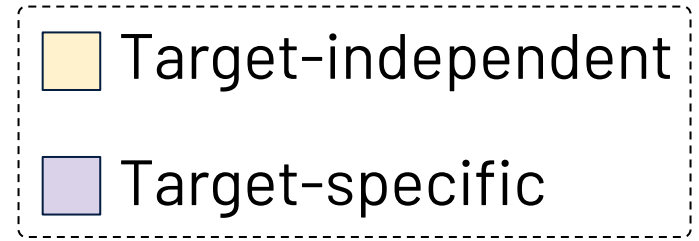


"Understanding the Open-Source P4_16 Compiler", Mihai Budiu, P4.org Open Source Developer Days, February 15, 2022

BMv2 Core Library

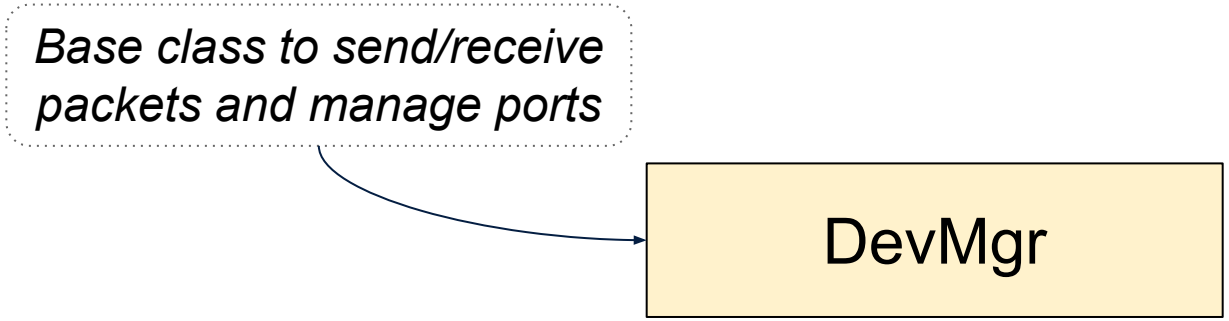
`behavioral-model/src/bm_sim`

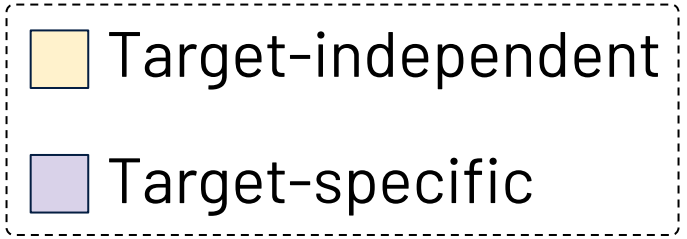
BMv2 Core Library



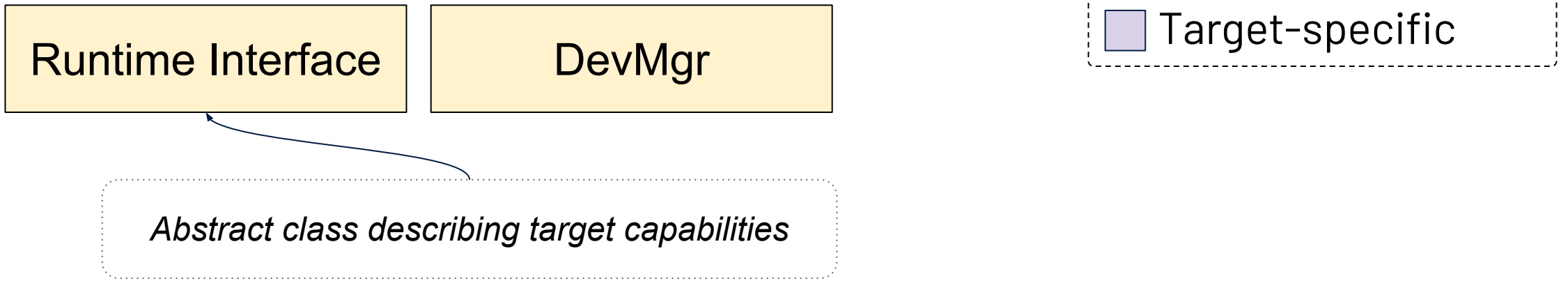
DevMgr

BMv2 Core Library

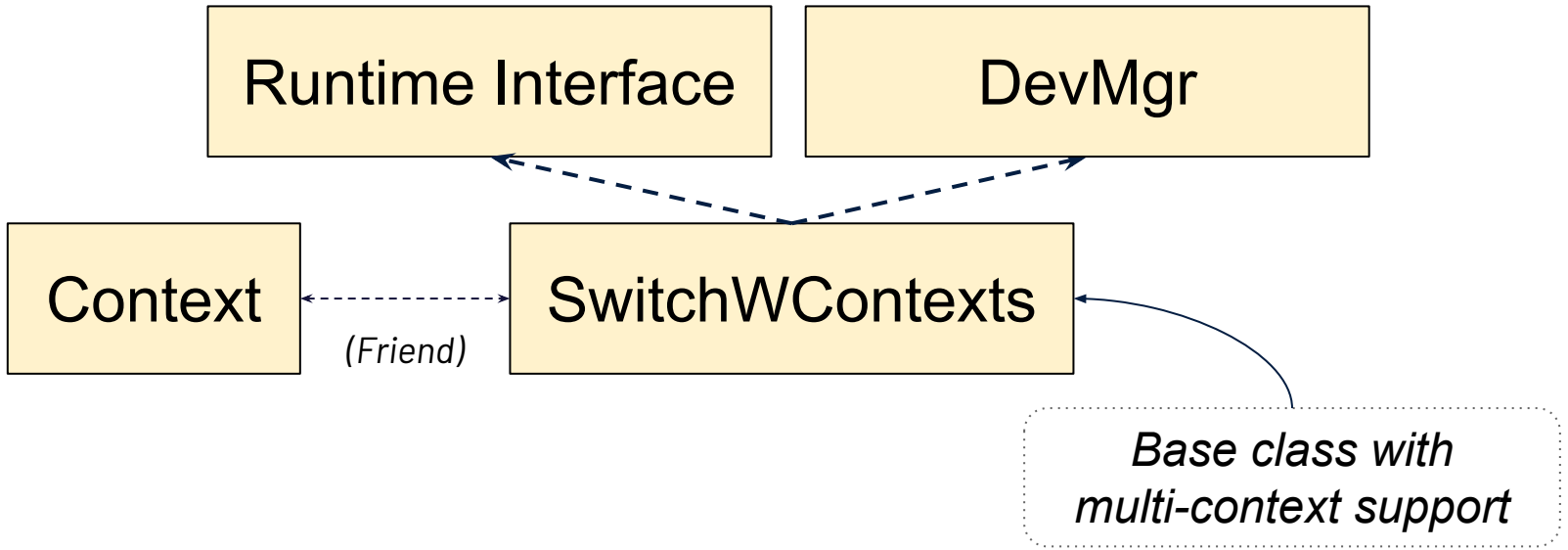


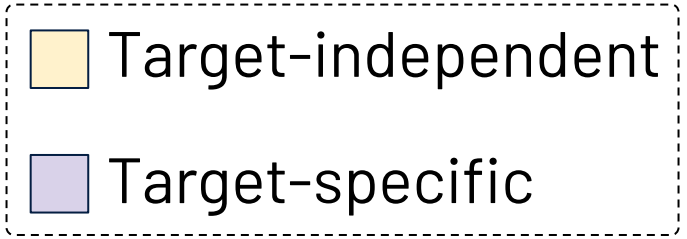


BMv2 Core Library

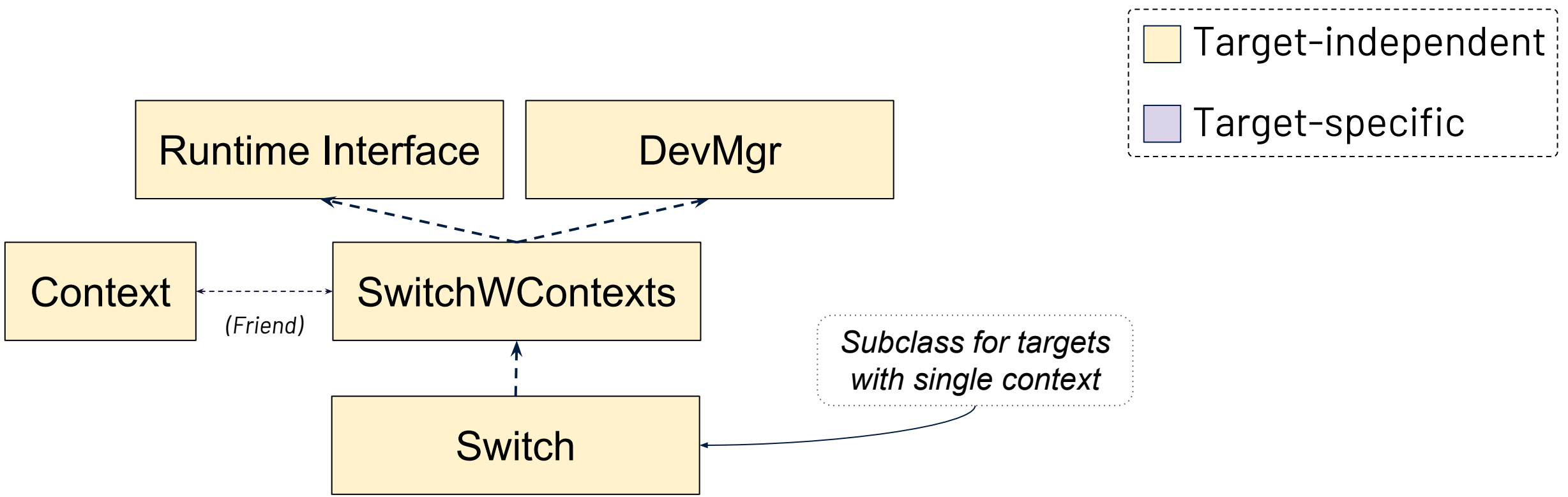


BMv2 Core Library

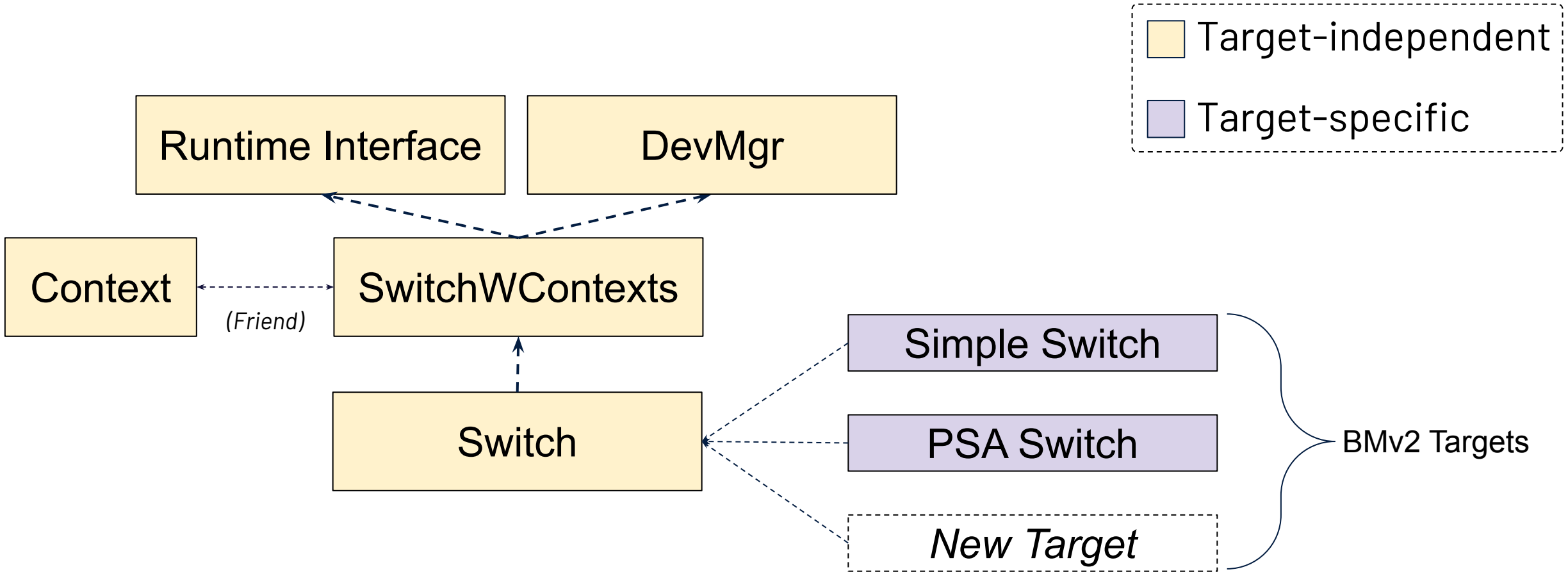




BMv2 Core Library



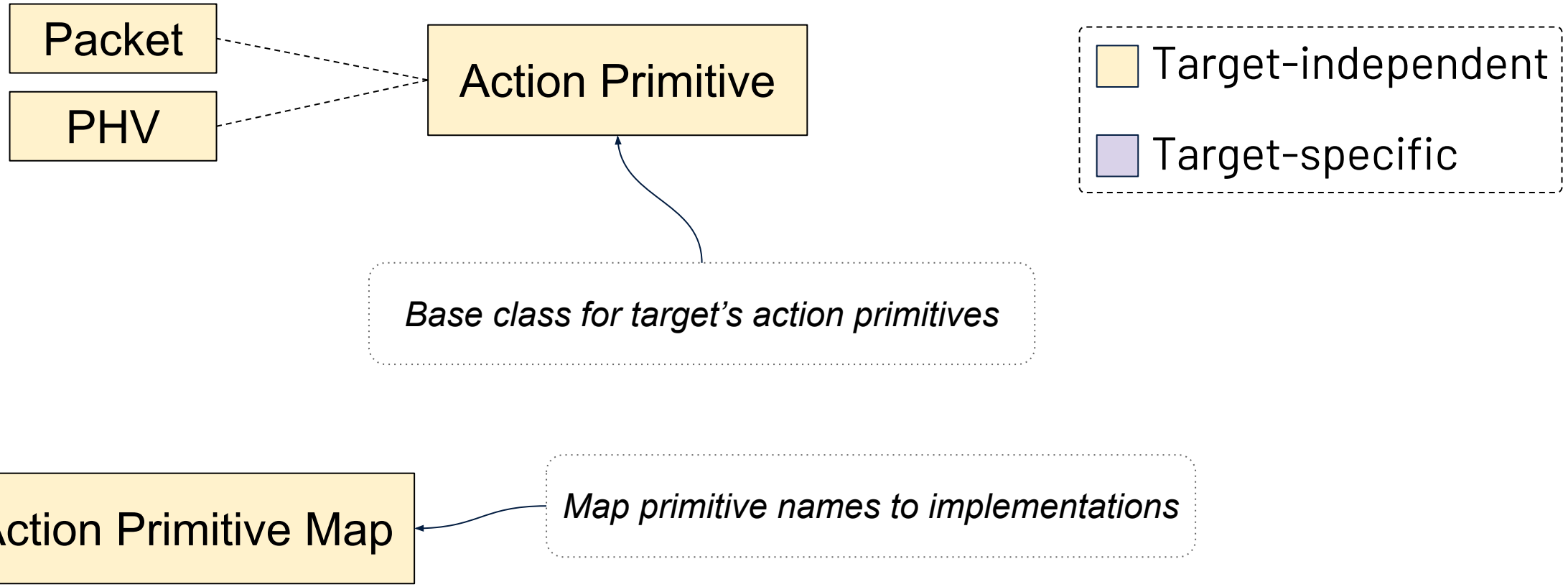
BMv2 Core Library



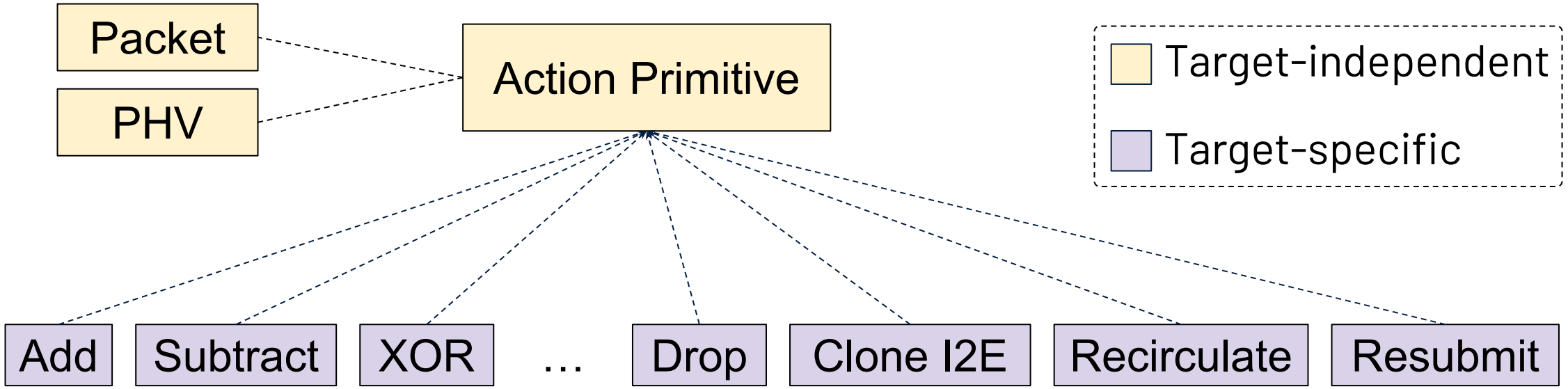
Action Primitives

`behavioral-model/targets/*/primitives.cpp`

Action Primitives



Action Primitives



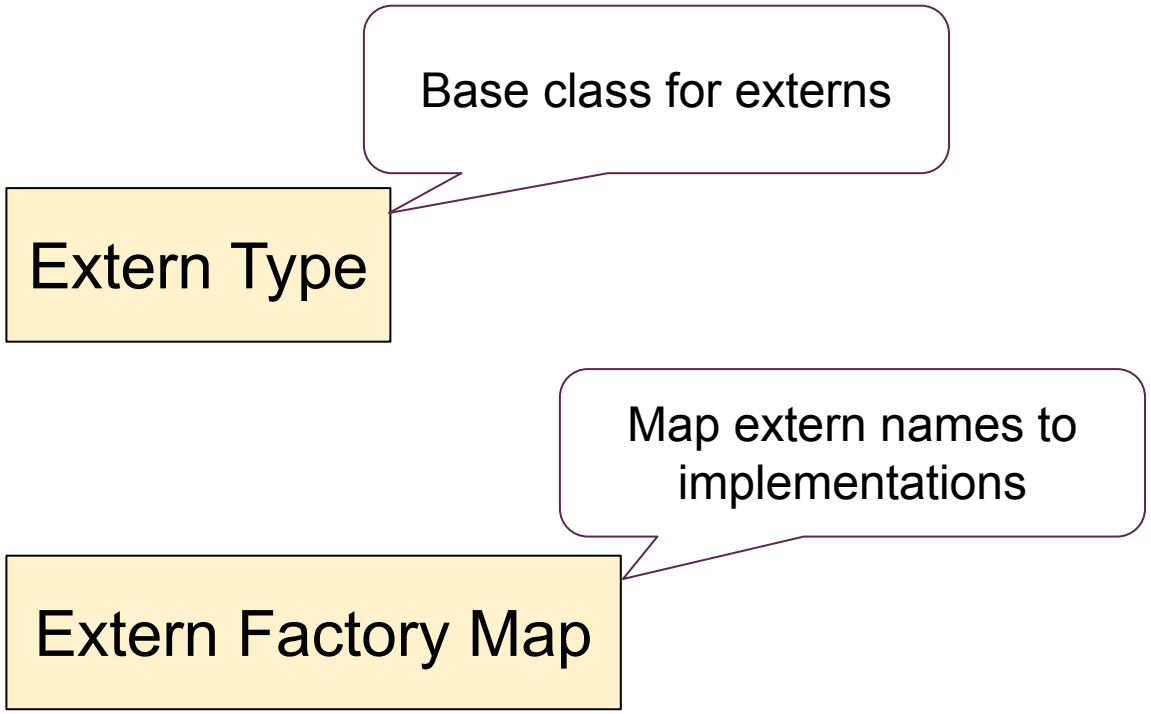
Action Primitive Map

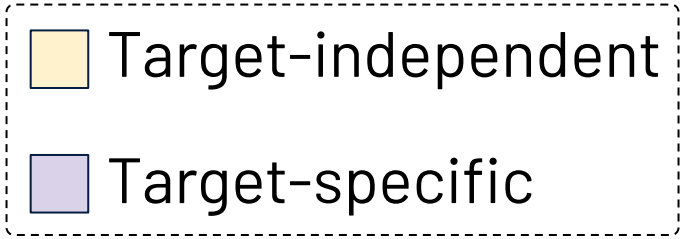
```
REGISTER_PRIMITIVE(primitive_name)
```

Externs

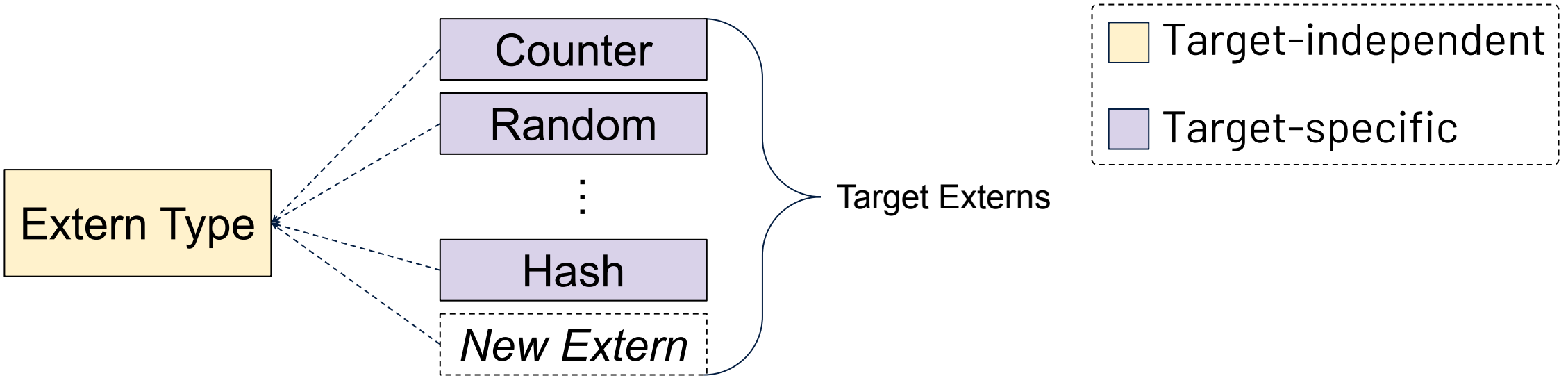
`behavioral-model/targets/*/externs`

Externs


Base class for externs
Extern Type
Map extern names to implementations
Extern Factory Map


Target-independent
Target-specific

Externs



Extern Factory Map

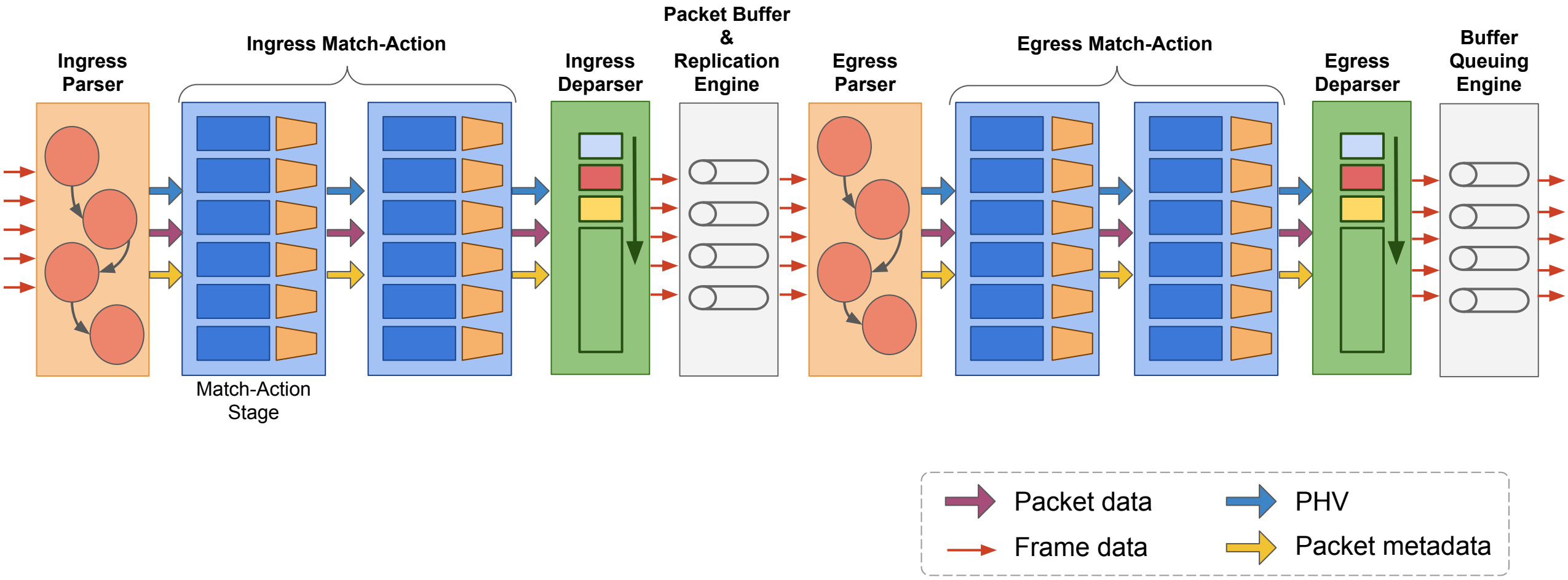
```
BM_REGISTER_EXTERN(extern_name)
BM_REGISTER_EXTERN_METHOD(extern_name, extern_method_name, ...)
BM_EXTERN_ATTRIBUTES
```

PSA Switch Target

`p4c/backends/bmv2/psa_switch/`

`behavioral-model/targets/psa_switch/`

Portable Switch Architecture (PSA)

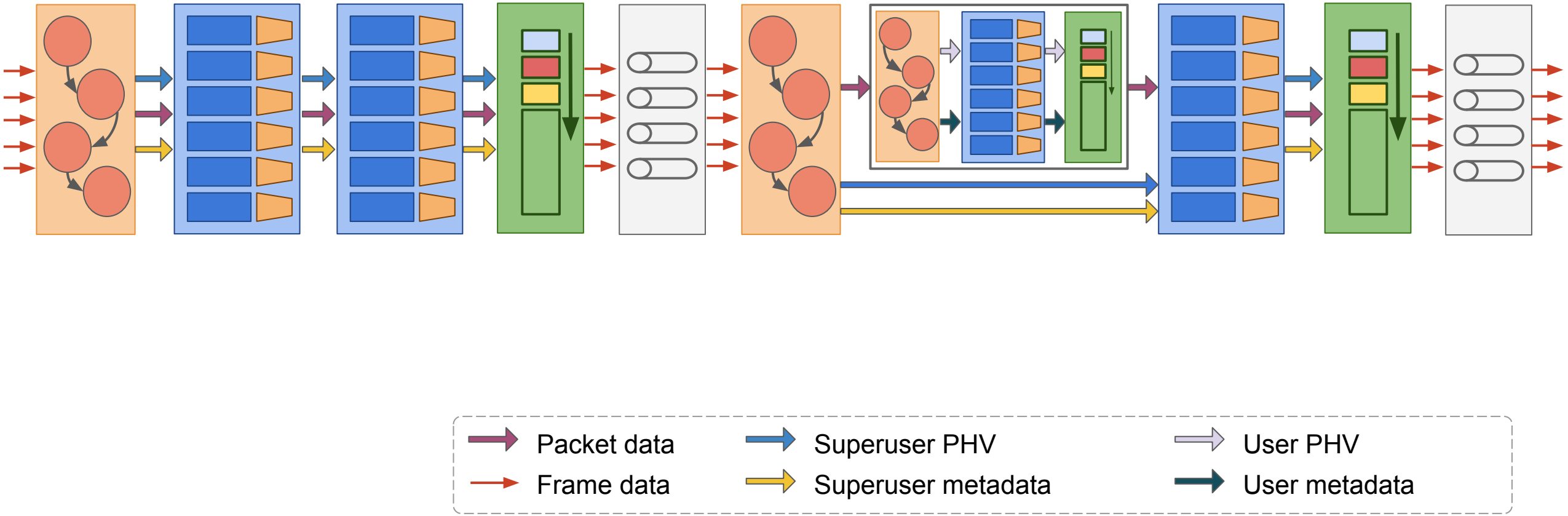


MTPSA Switch Target

`p4c/backends/bmv2/mtpsa_switch/`

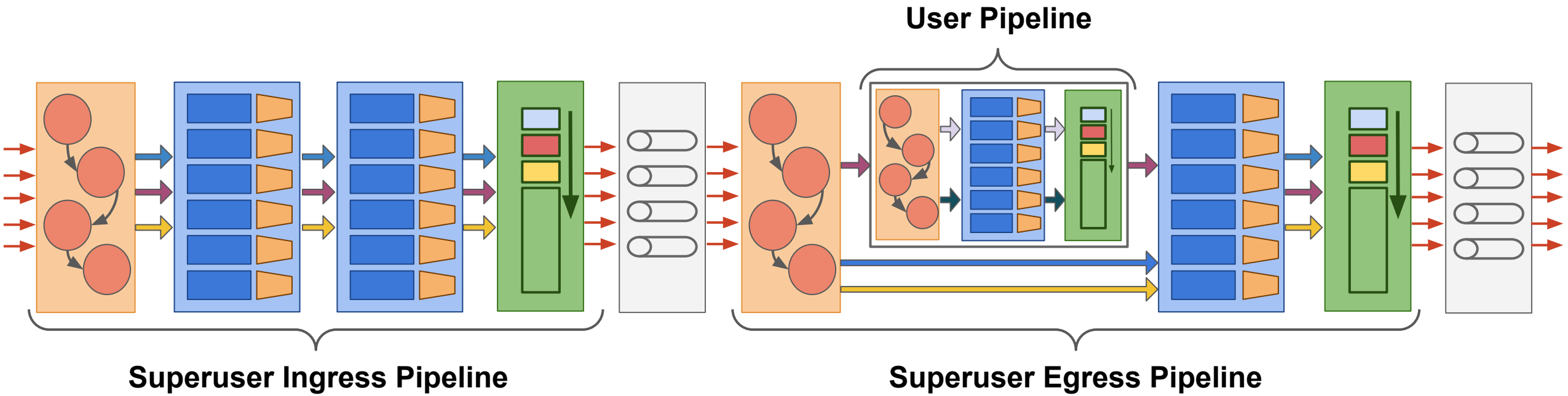
`behavioral-model/targets/mtpsa_switch/`

Multi-Tenant Programmable Switch



"MTPSA: Multi-Tenant Portable Switch Architecture", Radostin Stoyanov, and Noa Zilberman, EuroP4. 2020.

Multi-Tenant Programmable Switch



"MTPSA: Multi-Tenant Portable Switch Architecture", Radostin Stoyanov, and Noa Zilberman, EuroP4. 2020.

MTPSA Compiler

`p4include/mtpsa.p4` → *Superuser Program*

`p4include/mtpsa_user.p4` → *User Program*

```
$ p4c-bm2-mtpsa --help 2>&1 | grep user
--user   [MtPsaSwitch back-end] Compile user program
```

MTPSA Superuser Program

p4include/mtpsa.p4

p4include/mtpsa_user.p4

```
struct mtpsa_ingress_output_metadata_t {
    ...
    bit<16>   user_id;
    bit<16>   user_permissions;
}
```

MTPSA Superuser Program

`p4include/mtpsa.p4`

`p4include/mtpsa_user.p4`

```
package MTPSA_Switch<> (
    IngressPipeline<> ingress,
    PacketReplicationEngine pre,
    EgressParser<> ep,
    Egress<> eg,
    EgressDeparser<> ed,
    BufferingQueueingEngine bqe
);
```

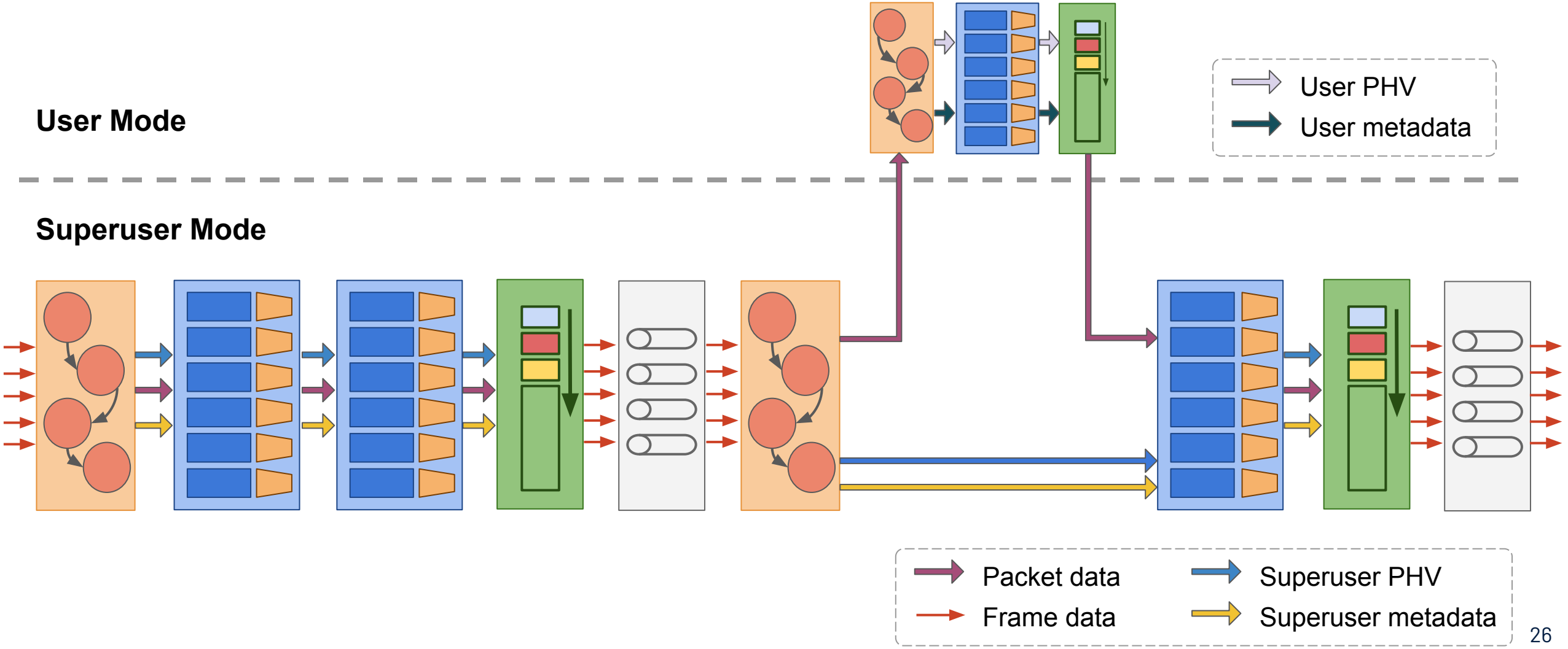
MTPSA User Programs

p4include/mtpsa.p4

p4include/mtpsa_user.p4

```
package MTPSA_User_Switch<> (
    Parser<> pr,
    Pipeline<> pi,
    Deparser<> dp
);
```

MTPSA Context Switching

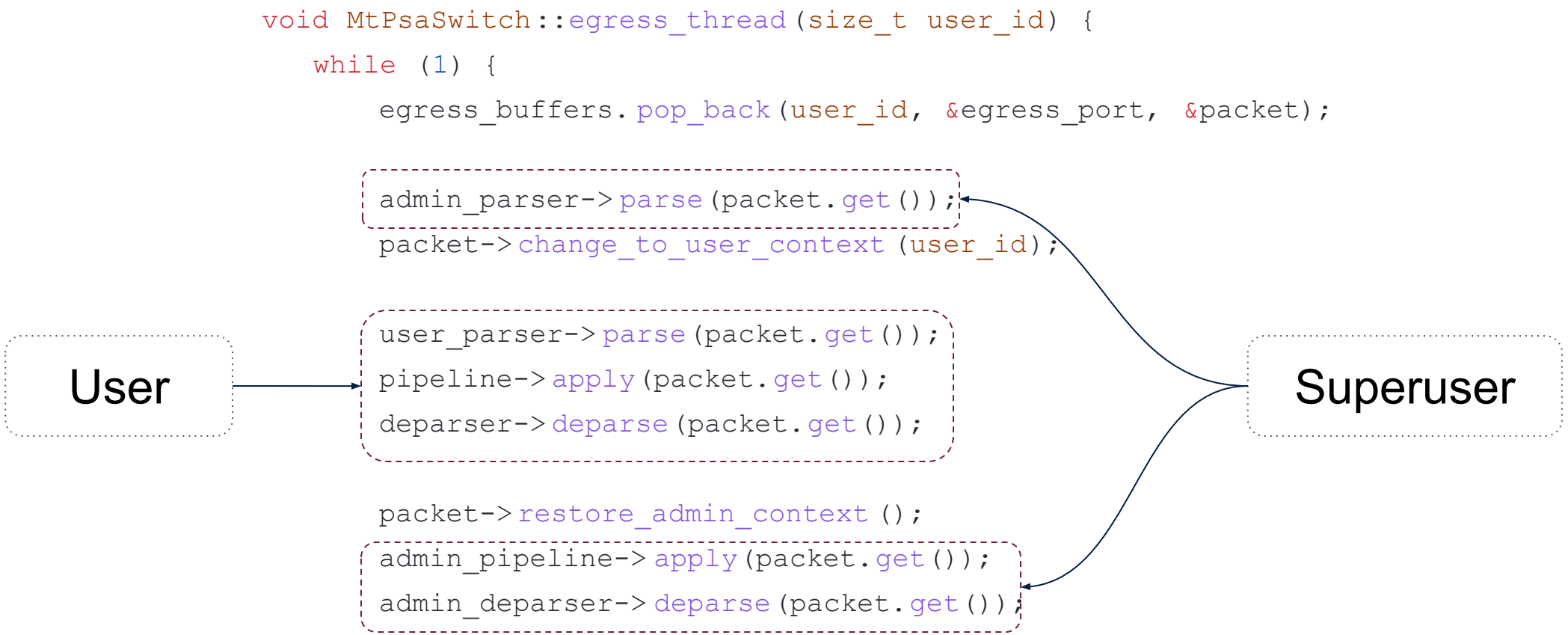


MTPSA Context Switching

User programs run in separate threads

```
void MtPsaSwitch::start_and_return_() {
    threads_.push_back(std::thread(&MtPsaSwitch::ingress_thread, this));
    for (size_t i = 0; i <= nb_user_threads; i++) {
        threads_.push_back(std::thread(&MtPsaSwitch::egress_thread, this, i));
    }
    threads_.push_back(std::thread(&MtPsaSwitch::transmit_thread, this));
}
```

MTPSA Context Switching



MTPSA User Permissions

Apply permissions to user pipeline

```
void MtPsaSwitch::egress_thread(size_t user_id) {
    while (1) {
        …
        packet->change_to_user_context(user_id);

        PHV *user_phv = packet->get_phv();
        user_phv->set_packet_permissions(user_permissions);

        user_parser->parse(packet.get());
        pipeline->apply(packet.get());
        deparser->deparse(packet.get());
        …
    }
}
```

MTPSA User Permissions

User permissions check

```
void MTPSA_Counter::count(const Data &index) {
    const Packet &packet = get_packet();
    const PHV *phv = packet.get_phv();
    const unsigned permissions = phv->get_packet_permissions();
    if (!(permissions & MTPSA_PERM_COUNTER))
        _counter->get_counter(index.get<size_t>()).increment_counter(packet);
}
```

MTPSA Workflow

1. Compile Superuser program

```
$ p4c-bm2-mtpsa …
```

MTPSA Workflow

1. Compile Superuser program

```
$ p4c-bm2-mtpsa …
```

2. Compile User programs

```
$ p4c-bm2-mtpsa --user …
```

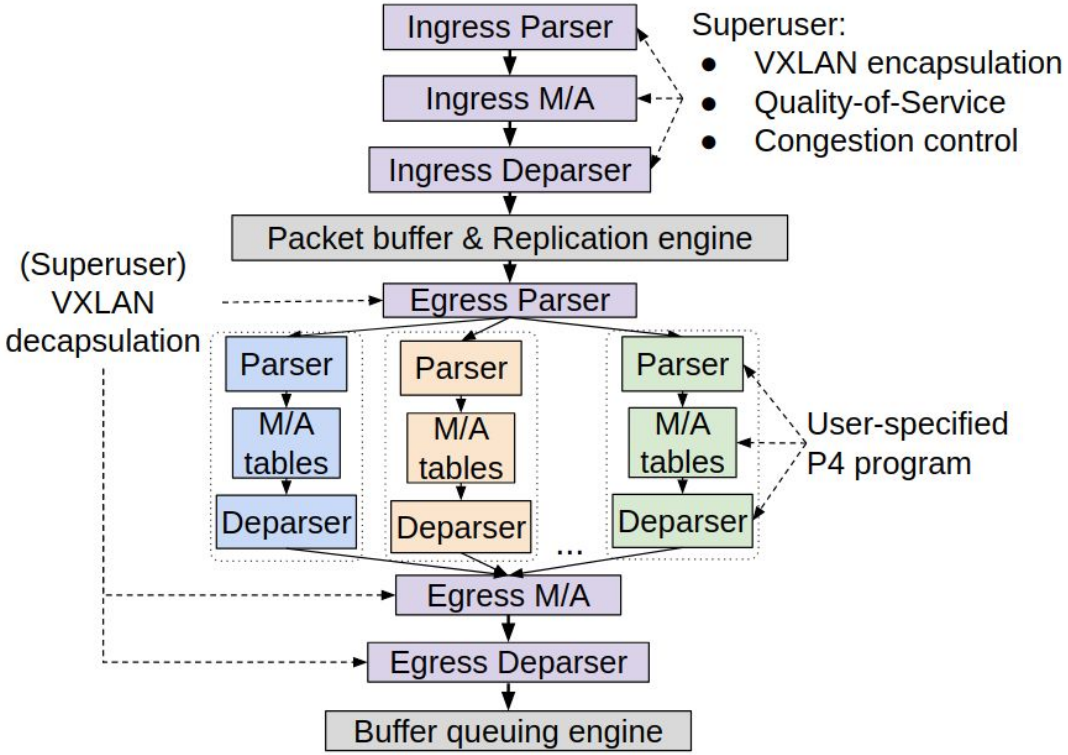
MTPSA Workflow

1. Compile Superuser program
2. Compile User programs
3. Start switch target
4. Configure Superuser Pipeline
5. Configure User Pipelines

```
$ p4c-bm2-mtpsa …

$ p4c-bm2-mtpsa --user …

$ mtpsa_switch … \
    --device-id 0 \
    --user 1@user01.json \
    --user 2@user02.json \
    main.json
```



Summary & Questions?

Building a P4 target with BMv2

- **P4 Architecture**
- **P4 Compiler**
- **Target-independents vs Target-specific components**
- **MTPSA Switch Target**

https://github.com/p4lang/behavioral-model

https://github.com/p4lang/p4c

https://github.com/mtpsa