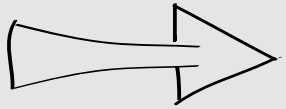
OpenNFP



Software Techniques for Programmable Data Plane Virtualization

David Hancock (dhancock@cs.utah.edu)

THE UNIVERSITY OF UTAH

Flux Research Group

Overview

Motivation

Approach

Controller (“Control Plane Hypervisor”)

HyPer4.p4 (“Data Plane Hypervisor”)

Demo

Wrap-up / Questions

Motivation

OpenNFP

Programmable data planes

- How dynamic are they? Can we insert functionality on demand without disrupting important flows?

Multi-tenant environments

Virtualization

- Isolation
- Abstraction
  - Composition
  - Introspection
  - Standard functionality

Approach

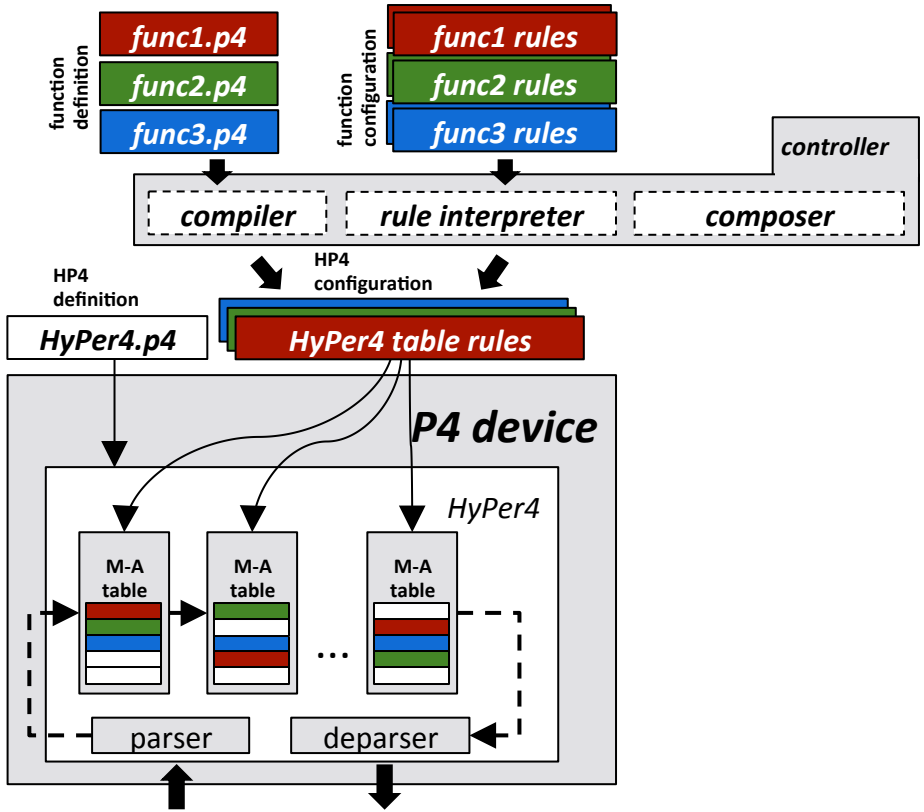
Software approach to expanding PDP device feature set

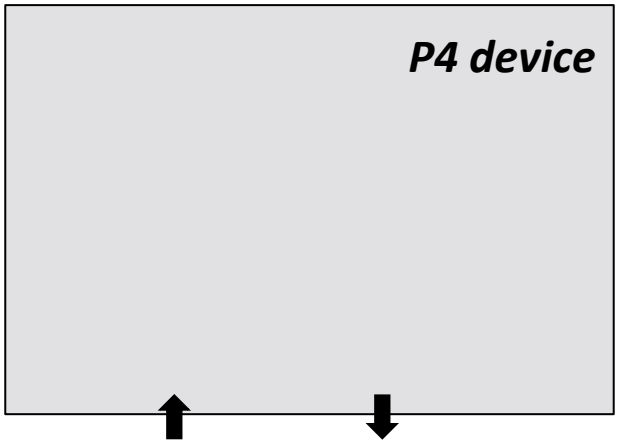
HyPer4.p4: “Data Plane Hypervisor”

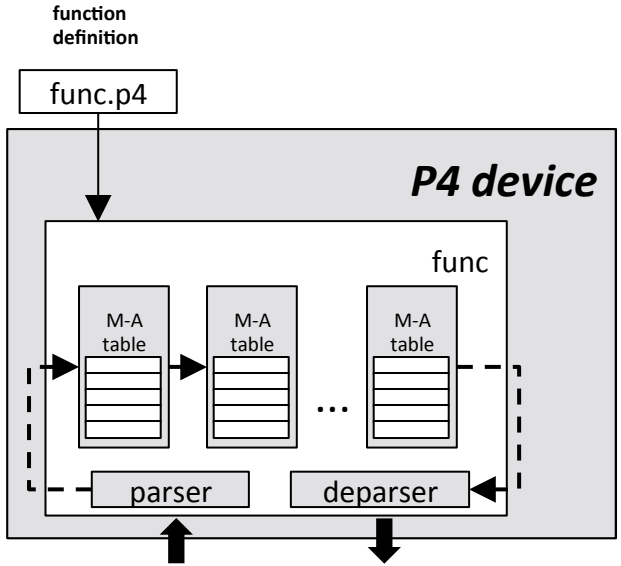
Controller: “Control Plane Hypervisor”

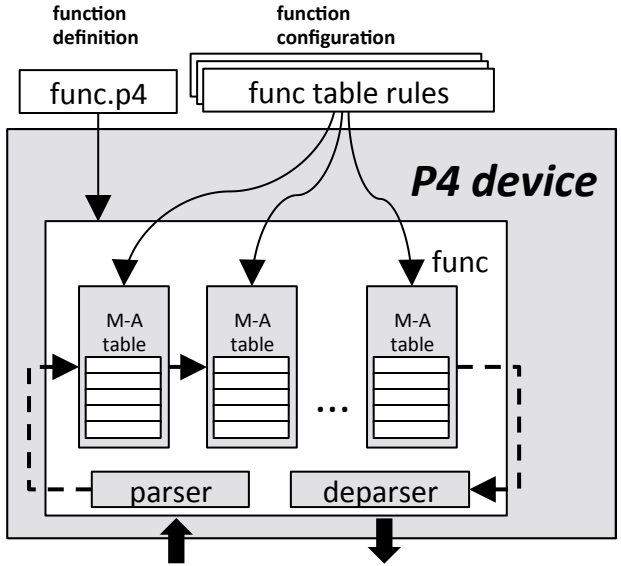
- Slice definition / provisioning
- Slice management
  - Compiler
  - Loader
  - Rule interpreter
  - Composer

HyPer4

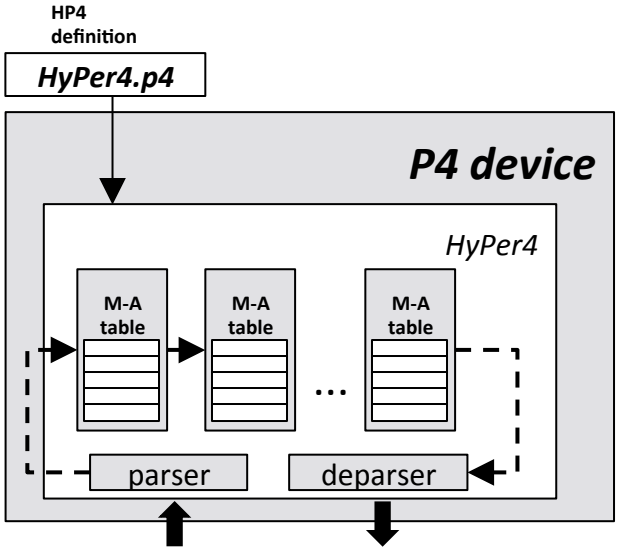




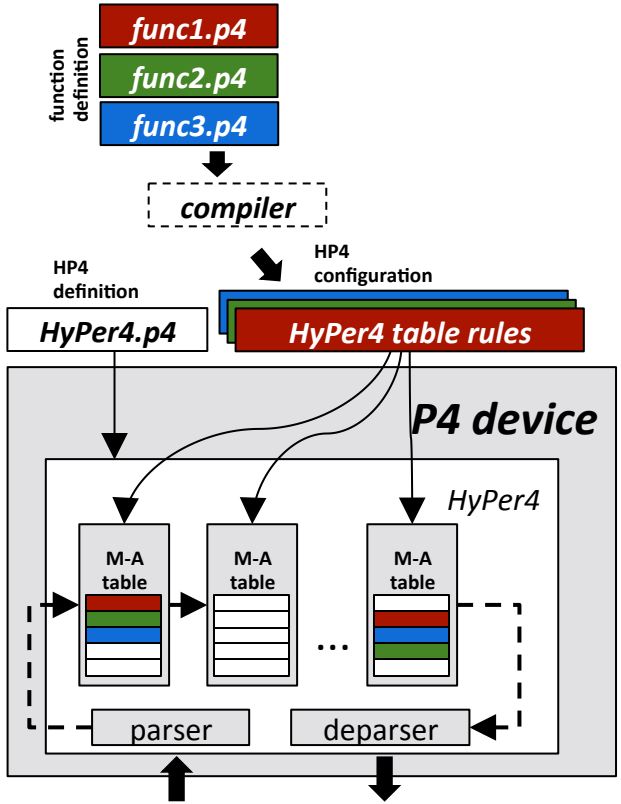




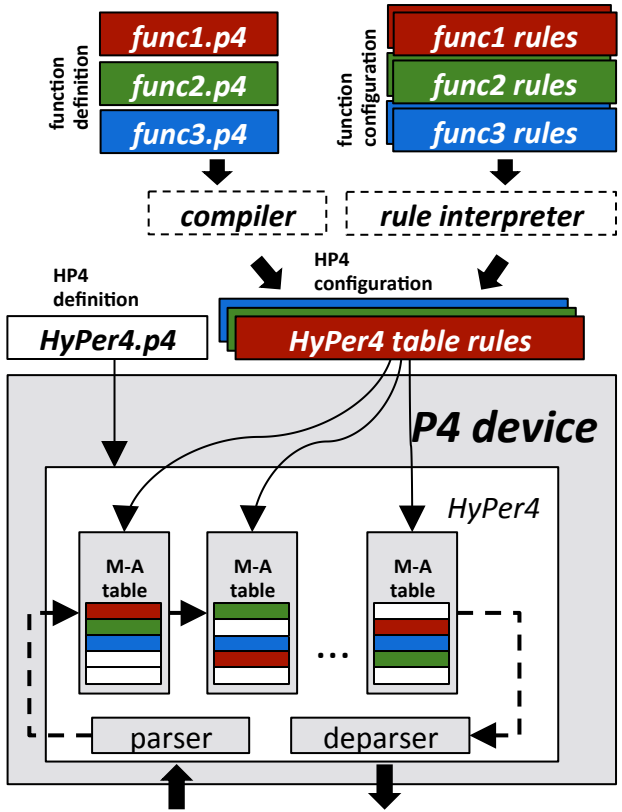
HyPer4



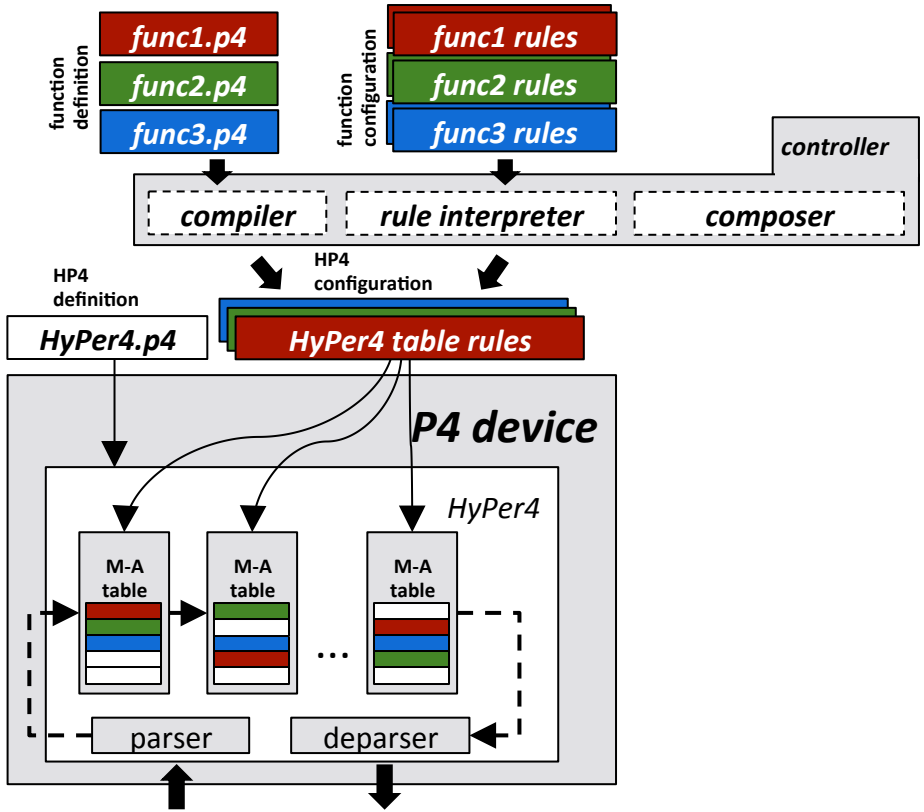
HyPer4


func1.p4
func2.p4
func3.p4
function definition
compiler
HP4 definition
HP4 configuration
HyPer4.p4
HyPer4 table rules
P4 device
HyPer4
M-A table
M-A table
M-A table
parser
deparser

HyPer4


function definition
func1.p4
func2.p4
func3.p4
function configuration
func1 rules
func2 rules
func3 rules
compiler
rule interpreter
HP4 definition
HyPer4.p4
HP4 configuration
HyPer4 table rules
P4 device
HyPer4
M-A table
M-A table
M-A table
parser
deparser

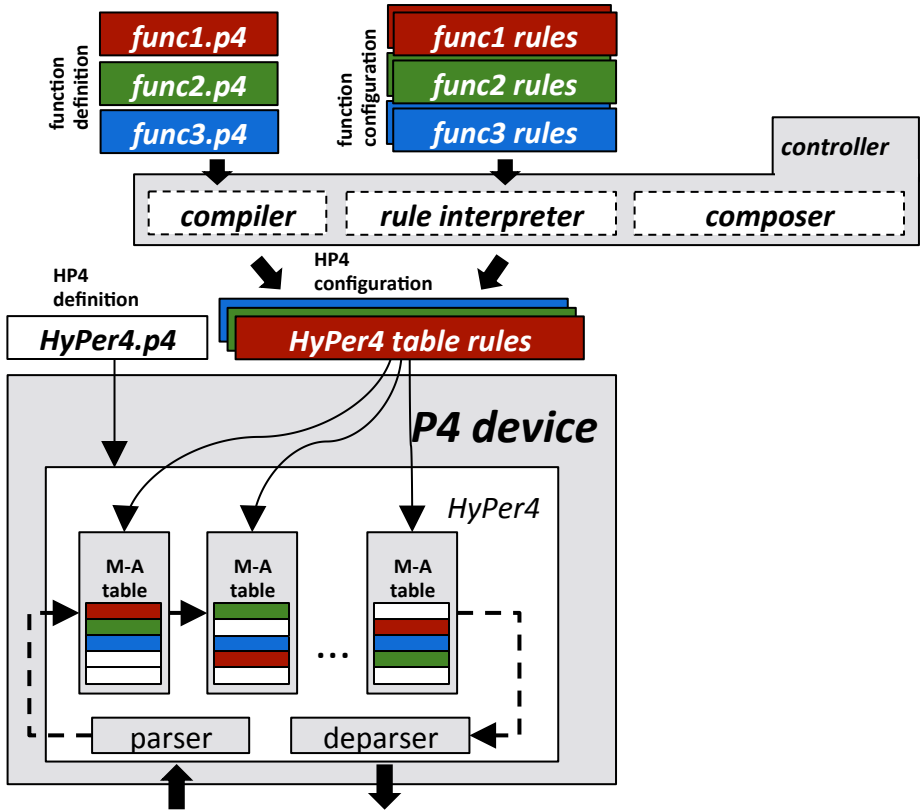
HyPer4



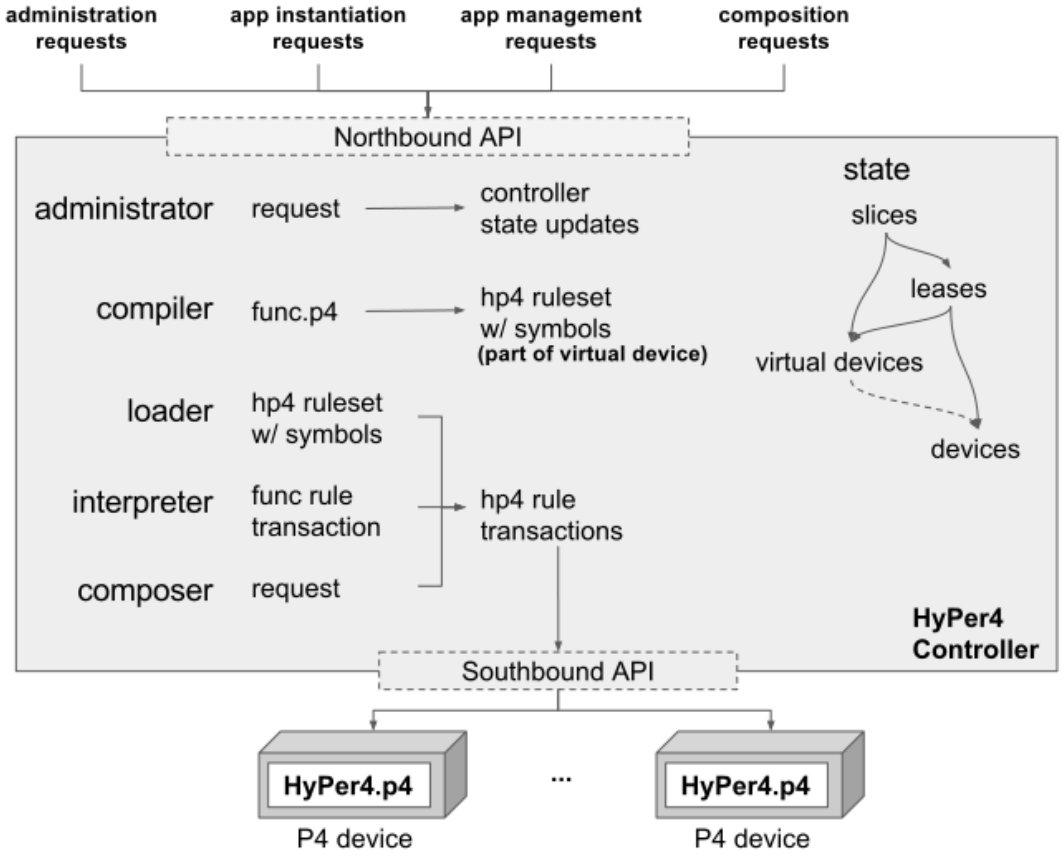
HyPer4.p4: Packet Tagging

```
action a_set_context(vdev_ID, vingress) {
  modify_field(meta.vdev_ID, vdev_ID);
  modify_field(meta.vingress, vingress);
}
table tset_context {
  reads { standard_metadata.ingress_port : exact; }
  actions { a_set_context; }
}
control setup {
  if (meta.vdev_ID == 0) {
    apply(tset_context);
  }
  // …
```

HyPer4



HyPer4 Controller



Northbound API

OpenNFP

Administrator

- create_device <dev name> <ip_addr> <port> \
  <dev_type> <# entries> <ports>
- create_slice <slice name>
- grant_lease <slice name> <dev name> <entry limit> <ports>
- list_devices
- list_slices
- reset_device <dev name>
- revoke_lease <slice name> <dev name>

Northbound API

OpenNFP

Slice Manager

- create_virtual_device <p4 path> <vdev name>
- destroy_virtual_device <vdev name>
- interpret <vdev name> <API style> <command>
- interpret_file <vdev name> <API style> <command> \
  <file>
- lease insert | append | remove | replace <vdev name> [args]
- list_devs
- list_vdev
- list_vdevs

Southbound API

OpenNFP

Device class defines interface

- send_command
  - do_table_add
  - do_table_modify
  - do_table_delete
- various multicast-related methods

Device subclasses implement

- Bmv2-SSwitch
- Agilio

Compiler/Interpreter

Task: P4 programs → “object-code-like” HyPer4 rulesets
→ rule interpreter guidance

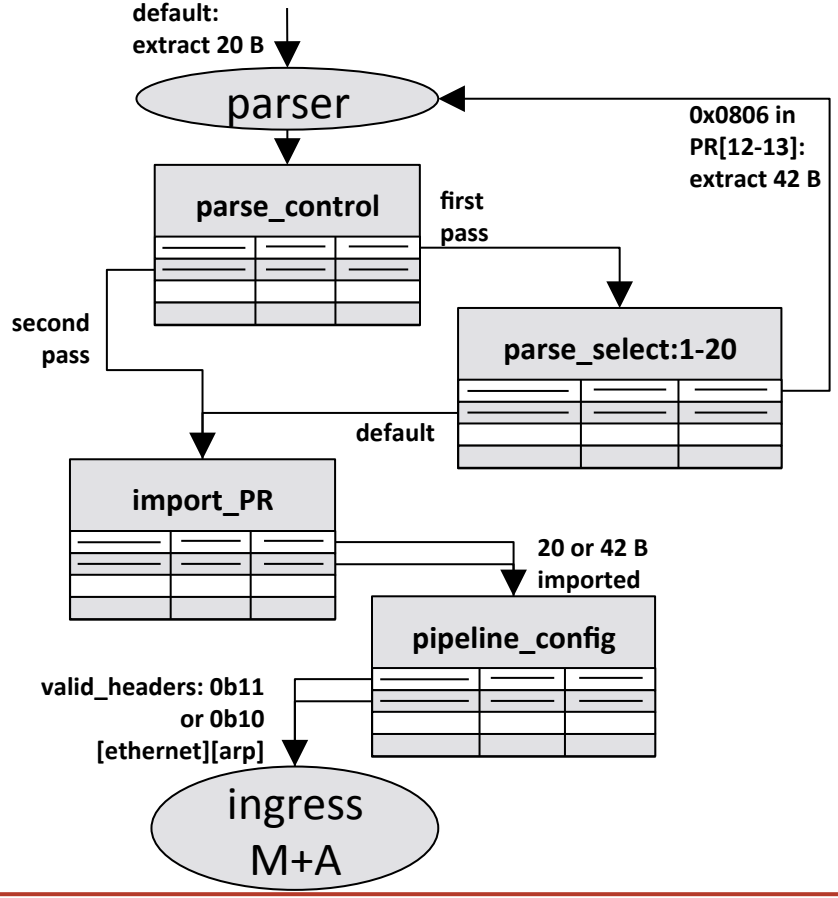
Subtasks:

- metadata field addressing
  - All metadata fields consolidated into HyPer4.p4’s meta.data field
  - Reads/writes done with bitmasks and bitshifts applied to meta.data
- parsing
- matching
- actions
- control flow

Compiler/Interpreter

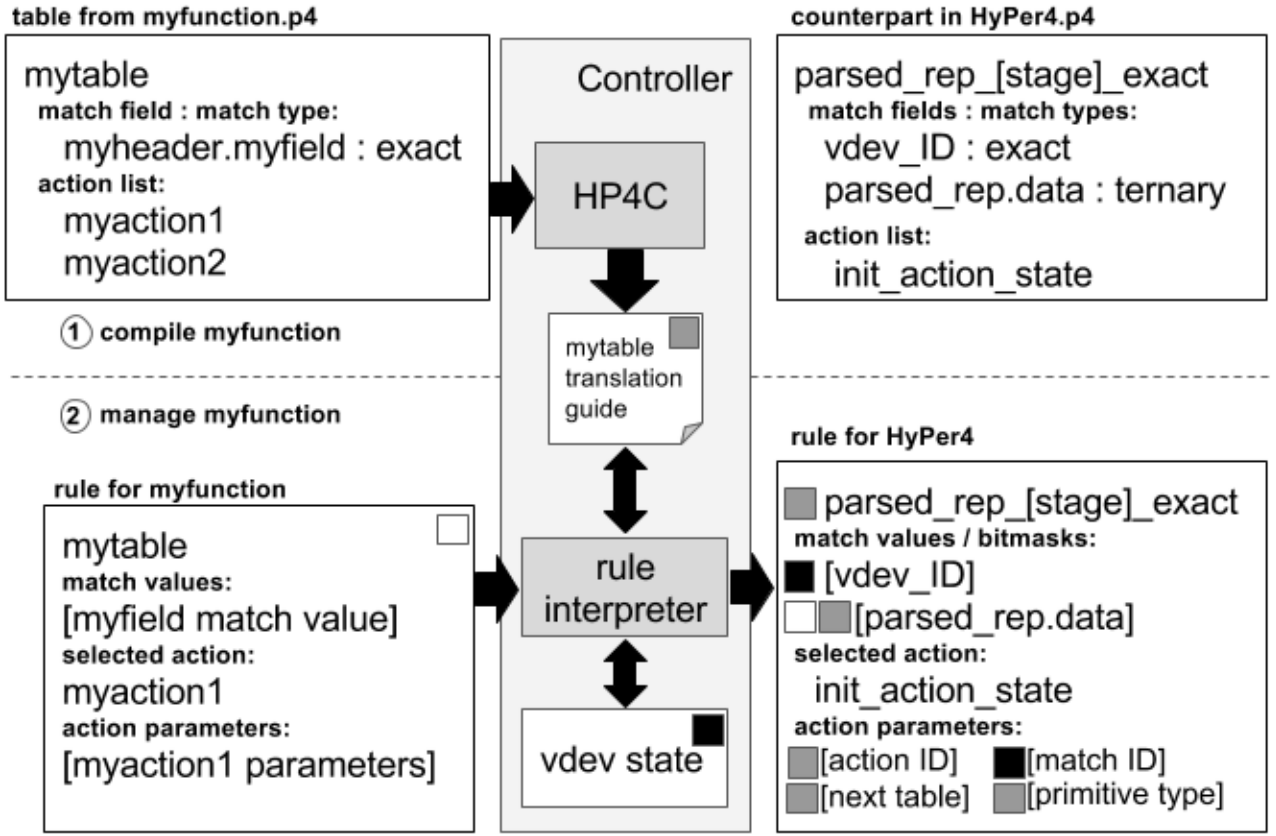
Subtasks: Parsing

```
parser start {
extract(ethernet);
    return select(ethernet.EtherType) {
        0x0806: parse_arp;
        default : ingress;
    }
}
parser parse_arp {
        extract(arp);
        return ingress;
}
```



Compiler/Interpreter

Subtasks: Matching

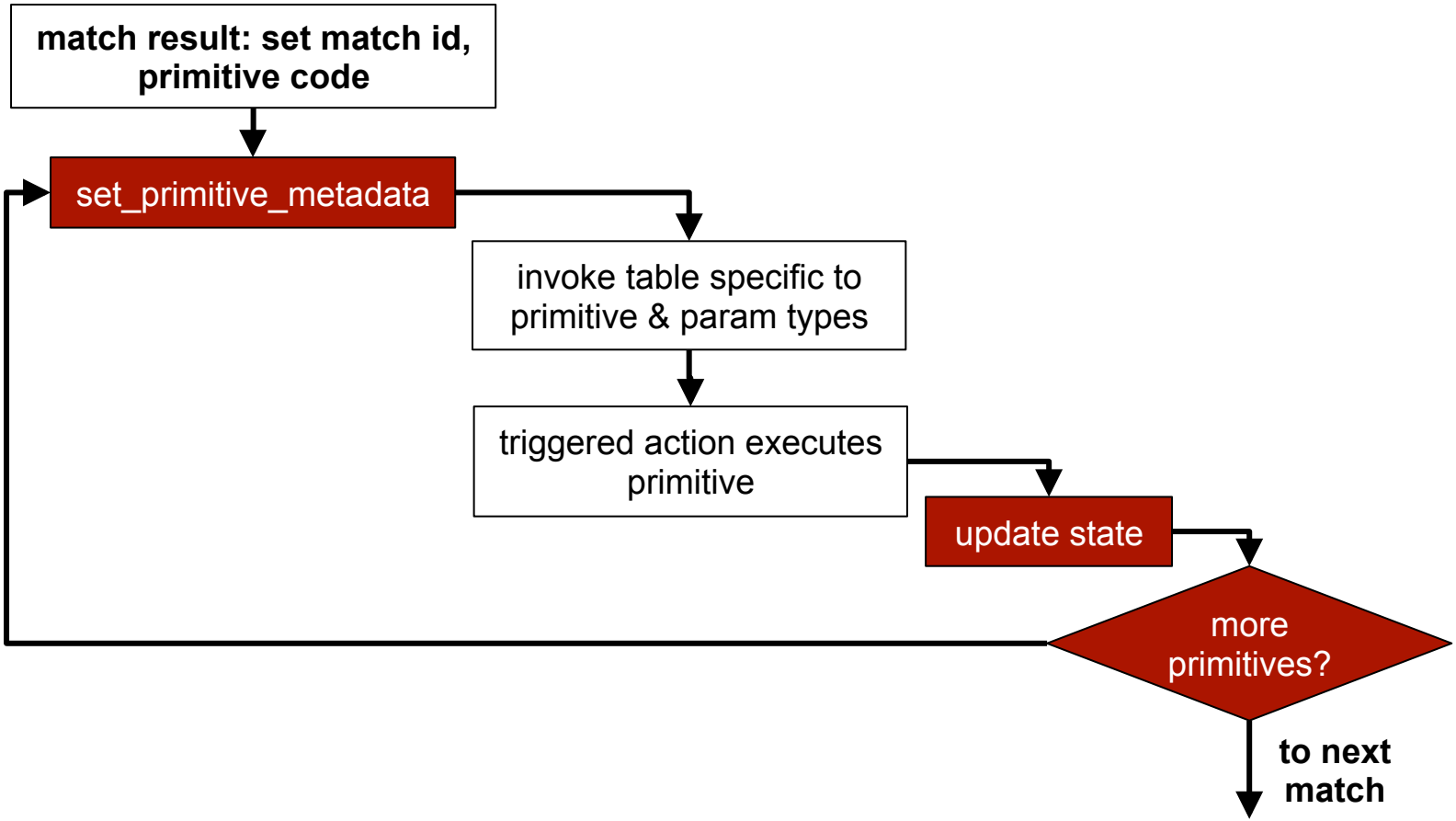


HyPer4.p4: Matching

Exact matching vs arbitrary fields:

- Use ternary matching
  - rules include mask
- Strategy: apply mask to pr.data to isolate relevant bits

HyPer4.p4: Actions



Loader

OpenNFP

"Object code":

```
tset_pipeline_config a_set_pipeline :[vdev ID] 1:
  [STDMETA_INGRESS_PORT_EXACT] 0x80[9x00s] 0



t_mod_41 mod_stdmeta_egressspec_stdmeta_ingressport :
  [vdev ID] 11 4 0&&&0:[MAX PRIORITY]
```

Loader

“Object code”:

table (end of “setup”) action virtualized function ID parse_state

```
tset_pipeline_config a_set_pipeline :[vdev ID] 1:
  [STDMETA_INGRESS_PORT_EXACT] 0x80[9x00s] 0
```

egress handling mode

ID of 1st match table

valid headers bitmap (0b1000…: ethernet only)

```
t_mod_41 mod_stdmeta_egressspec_stdmeta_ingressport :
  [vdev ID] 11 4 0&&&0:[MAX PRIORITY]
```

Loader

OpenNFP

"Object code":

```
tset_pipeline_config a_set_pipeline :[vdev ID] 1:
  [STDMETA_INGRESS_PORT_EXACT] 0x80[9x00s] 0
```

table (modify_field) action

```
t_mod_41 mod_stdmeta_egressspec_stdmeta_ingressport :
[vdev ID]   11      4      0&&&0:[LOWEST PRIORITY]
```

virtualized function ID

prim subtype

action ID

match ID

ternary match priority

Loader

OpenNFP

"Object code":

```
tset_pipeline_config a_set_pipeline :[vdev ID] 1:
  [STDMETA_INGRESS_PORT_EXACT] 0x80[9x00s] 0



t_mod_41 mod_stdmeta_egressspec_stdmeta_ingressport :
  [vdev ID] 11 4 0&&&0:[MAX PRIORITY]
```

Loader

Bmv2-SS API Commands:

```
table_add tset_pipeline_config a_set_pipeline 1 1 =>
  5 0x80000000000000000000 0
```

```
table_add t_mod_41 mod_stdmeta_egressspec_stdmeta_ingressport
  1 11 4 0&&&0 => 2147483646
```

HyPer4.p4: Composition Support

End of ingress

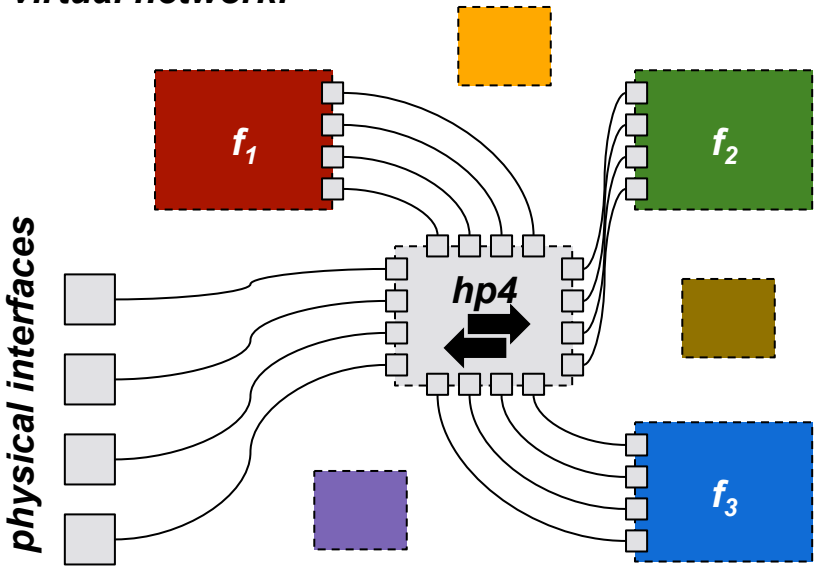
- Match on vegress
- Virtual forwarding? Set egress to ingress port, set virt forwarding flag

Egress

- Virt forwarding flag set?
  - Match on vegress
  - Set next vdev_ID, vingress
  - Recirculate with field list that includes key fields

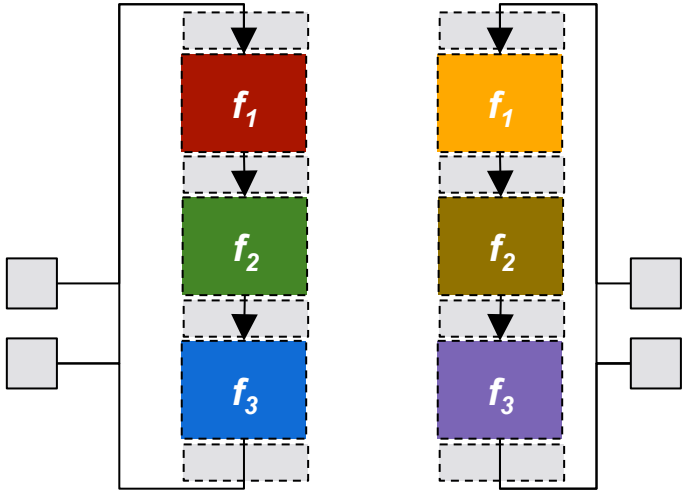
Composition Abstractions

virtual network:



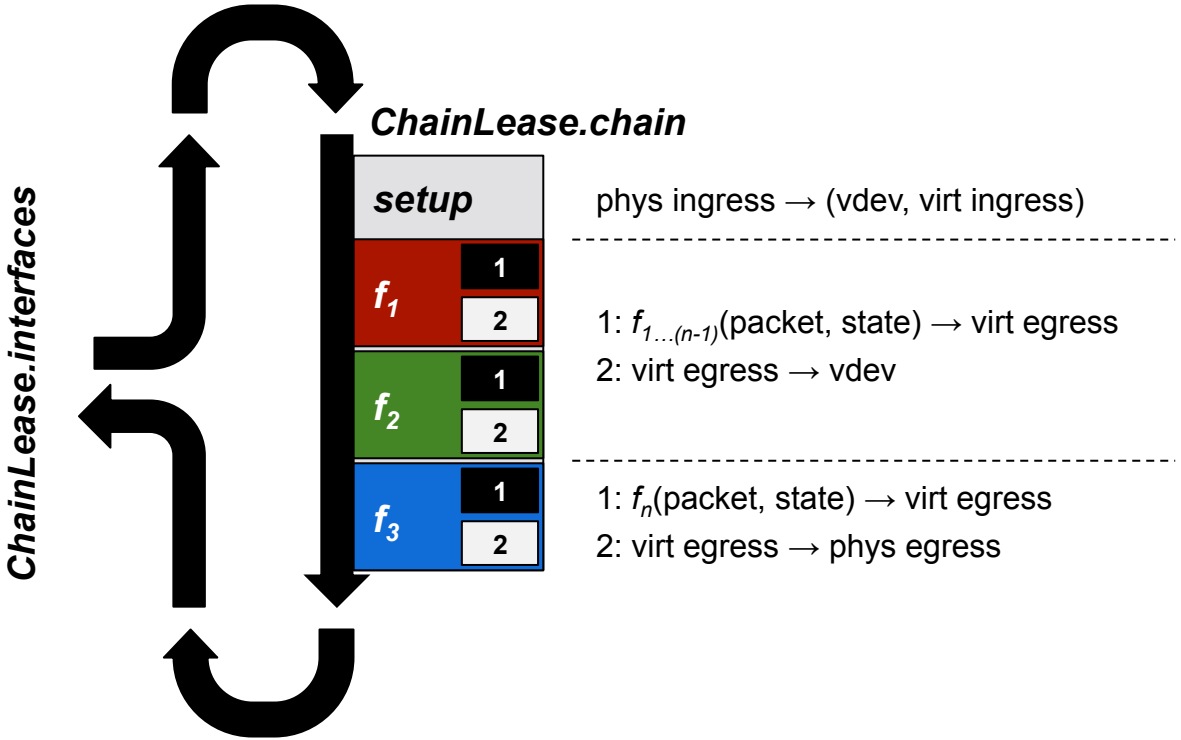
- Very flexible
- Complicated to implement and manage

vdev chains:



- Flexible
- Simpler to implement and manage

Chains


ChainLease.interfaces
ChainLease.chain
setup
f_1
1
2
f_2
1
2
f_3
1
2
phys ingress → (vdev, virt ingress)
1: $f_{1...(n-1)}$(packet, state) → virt egress
2: virt egress → vdev
1: f_n(packet, state) → virt egress
2: virt egress → phys egress

Egress Conflict Resolution

Types of Forwarding Abstractions

- return-to-sender: single port
- bump-in-the-wire [+ return-to-sender]: two ports
- switch: (packet, state) → egress: two+ ports

VDev sequence results in (sometimes unintentional) egress overrides

Resolution:

- ~~Enforce switch at the end~~
- Slice manager indicates preferences with respect to egress modifications for each vdev via controller command option:
  - always write to egress_spec
  - conditionally write to egress_spec (if void)
  - never write to egress_spec

Possibilities

Migration

Cloning

Maintain rulesets separately, attach to compatible vdevs on-demand

Automatically evolving functions

Linked functions (change to one begets [a]symmetrical change to another across the network)

HyPer4.p4: Clean up

Handle virtual networking

Egress filtering (avoid broadcast storms)

Handle checksum (e.g., IPv4)

Resize parsed representation (in case of add_header / remove_header)

Write pr.data back to parsed representation

HyPer4.p4: Code Generation

P4 compiler / target restriction: ONE reference per table

- Want: arbitrary functionality at arbitrary points in the pipeline
- Strategy: many functionally redundant copies, differing in name only
- Want: configurable support for specified # stages, specified # primitives / stage
- Strategy: define .p4 for single stage, single primitive, add tokens directing processing tool to produce desired configuration

HyPer4.p4: Code Generation

```
[+ sloop +]
control stage[+X+] {
  match_[+X+](); // match.p4
  [+ nif +]
  if(meta_ctrl.stage_state != COMPLETE) {
    switch_primitivetype_[+X+][+Y+]();
    apply(tstg[+X+][+Y+]_update_state);
  [+ endnif +]
}
[+ endsloop +]
```

```
control stage1 {
  match_1();
  if(meta_ctrl.stage_state != COMPLETE) {
    switch_primitivetype_11();
    apply(tstg11_update_state);
    if(meta_ctrl.stage_state != COMPLETE) {
      switch_primitivetype_12();
      apply(tstg12_update_state);
      // …
}
control stage2 {
  match_2();
  if(meta_ctrl.stage_state != COMPLETE) {
```

HyPer4.p4: Objections, Rebuttals

Performance Concerns

- May be addressed with custom targets
- May be addressed with more sensible default parsing behavior and streamlined match-action pipeline
- May be addressed by using programmable parser + fixed pipeline, or fixed parser + programmable pipeline
- Should not get in the way of exploring high level abstractions made possible by vPDPs

Demo

Wrap-up

Motivation

Approach

Controller (“Control Plane Hypervisor”)

HyPer4.p4 (“Data Plane Hypervisor”)

- git@gitlab.flux.utah.edu:dhankook/hp4-src.git
- README.md provides commit used for CoNEXT 2016 paper

Demo

Questions

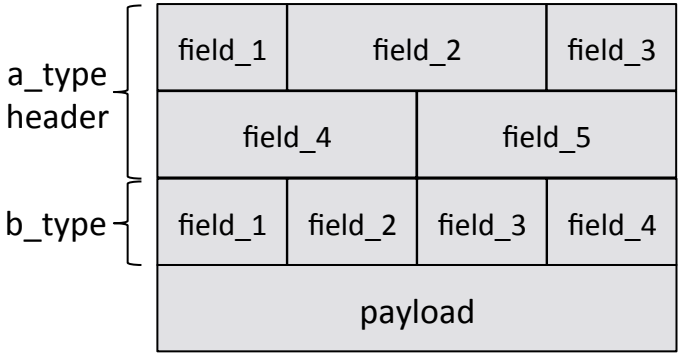
Questions?

HyPer4.p4: Parsing

```
// not shown: header_type definitions
header a_t a;
header b_t b;

parser start {
    extract(a);
    return select(a.field_3) {
        0x01 : parse_b;
        default : ingress;
    }
}
parser parse_b {
    extract(b);
    return ingress;
}
```

parsed representation: a | a, b

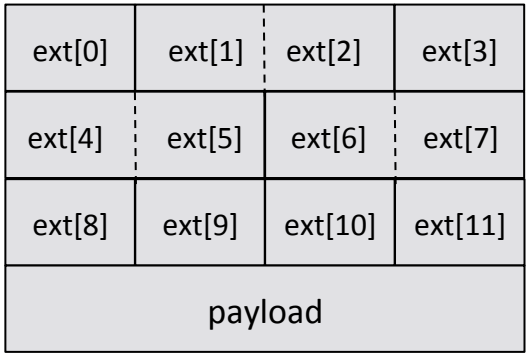


P4 source includes
references to named fields

HyPer4.p4: Parsing

Strategy:

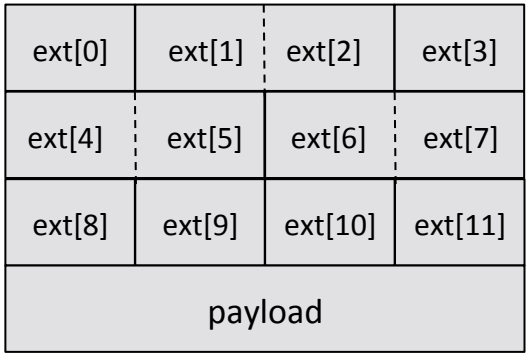
- generic 8-bit header type
- ***ext[100]***: array of such headers



HyPer4.p4: Parsing

Strategy:

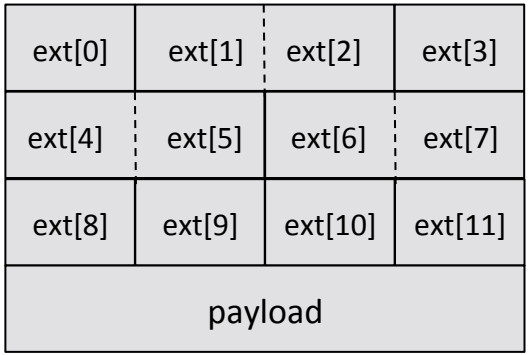
- generic 8-bit header type
- ***ext[100]***: array of such headers
- metadata fields: ***#bytes, parse_state***



HyPer4.p4: Parsing

Strategy:

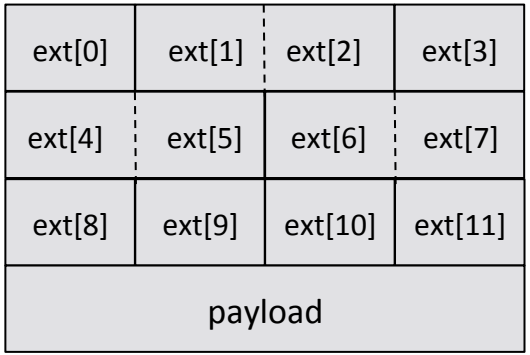
- generic 8-bit header type
- ***ext[100]***: array of such headers
- metadata fields: ***#bytes, parse_state***
- Parser: extract default -> 100 bytes
  - **#bytes** guides parser



HyPer4.p4: Parsing

Strategy:

- generic 8-bit header type
- ***ext[100]***: array of such headers
- metadata fields: ***#bytes, parse_state***
- Parser: extract default -> 100 bytes
  - **#bytes** guides parser
- Complex parsing logic moved to Ingress: **parse_state** may trigger update of **#bytes** + `resubmit`



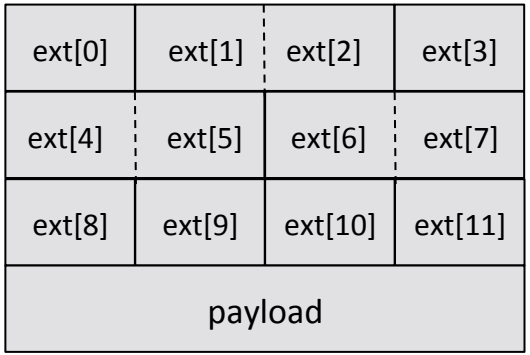
HyPer4.p4: Parsing

Strategy:

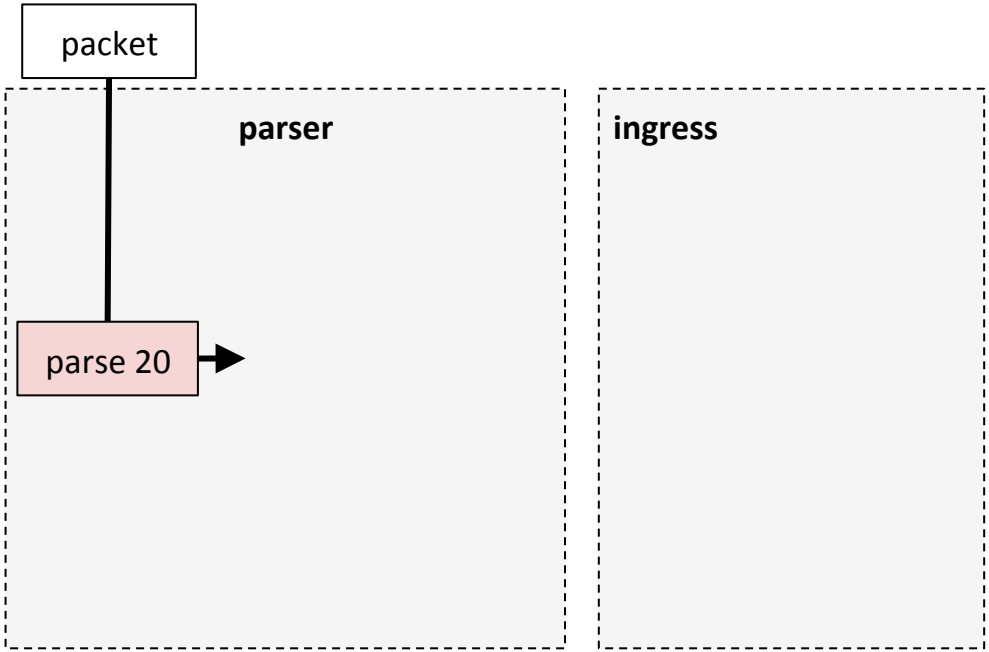
- generic 8-bit header type
- ***ext[100]***: array of such headers
- metadata fields: ***#bytes, parse_state***
- Parser: extract default -> 100 bytes
  - **#bytes** guides parser
- Complex parsing logic moved to Ingress: **parse_state** may trigger update of **#bytes** + `resubmit`

parsed representation: all valid elements of ext

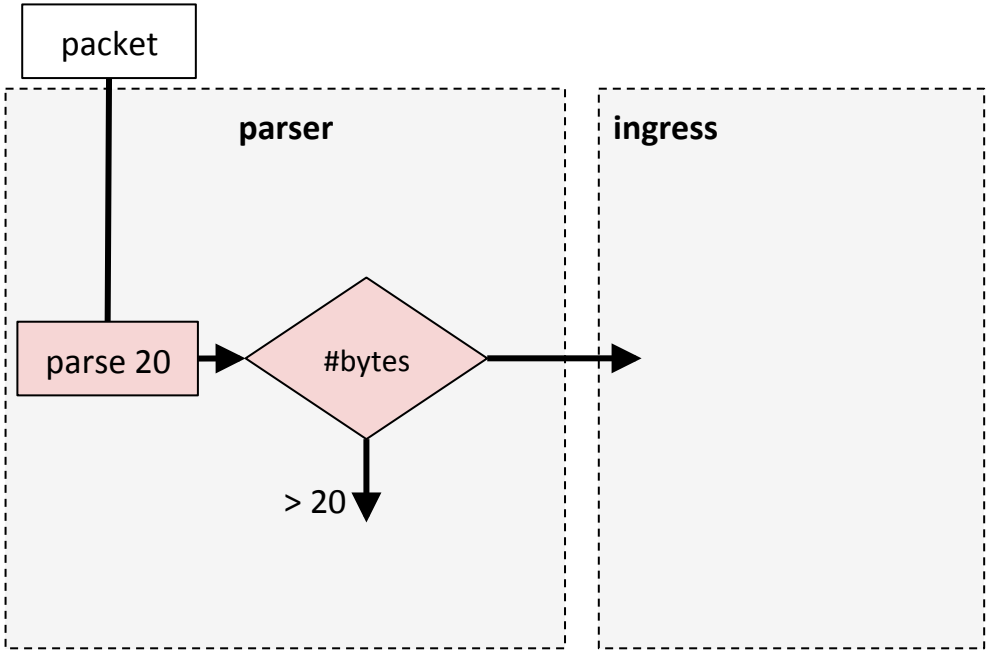
- Inconvenient to work with; copy to single very wide metadata field



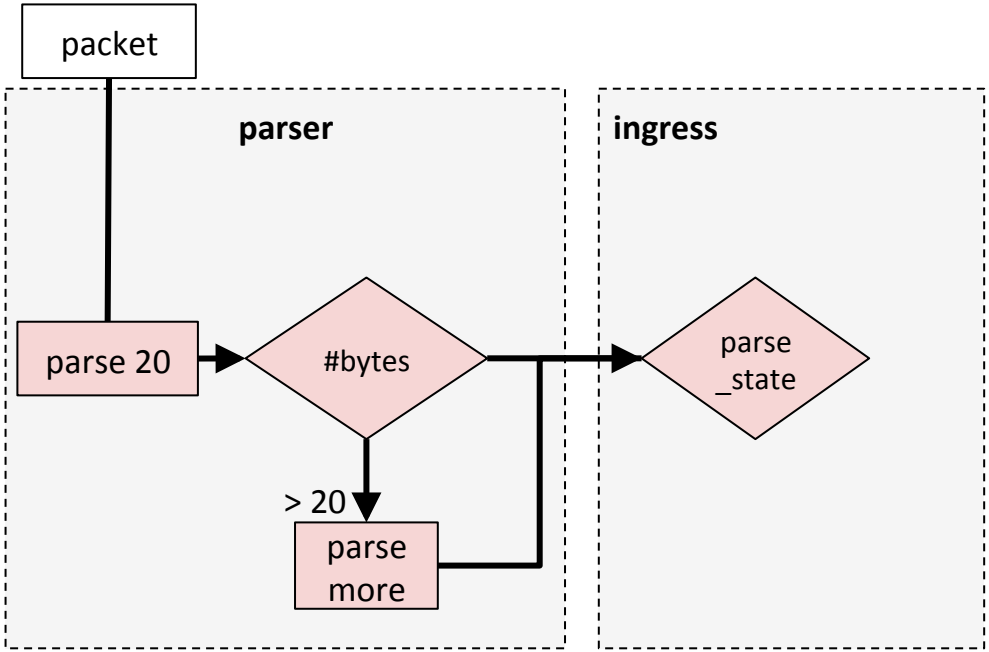
HyPer4.p4: Parsing



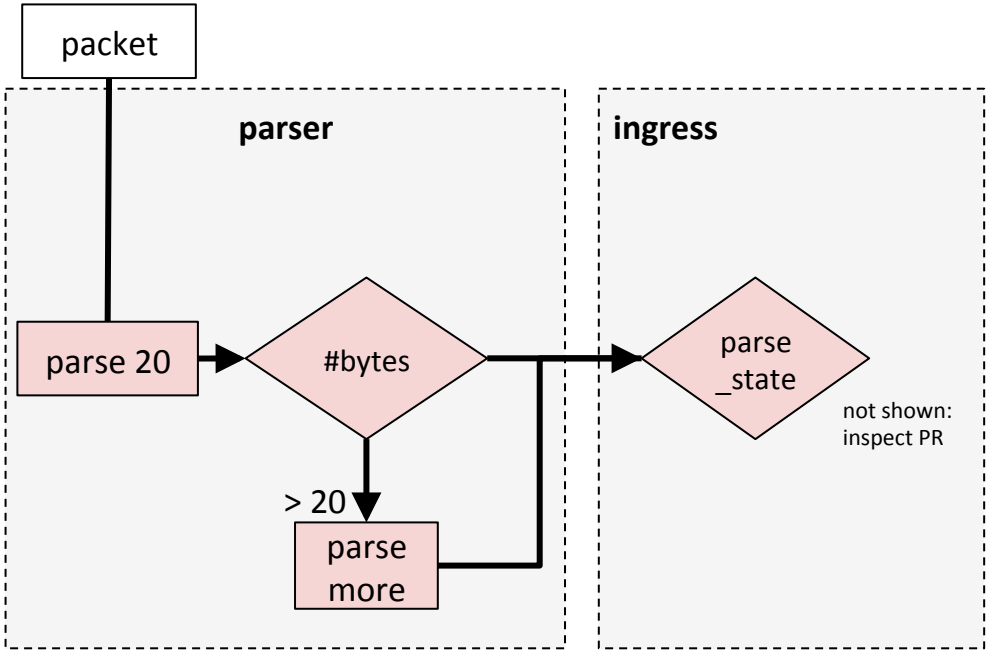
HyPer4.p4: Parsing



HyPer4.p4: Parsing



HyPer4.p4: Parsing



HyPer4.p4: Parsing