Lyra: A Cross-Platform Language and Compiler for Data Plane Programming on Heterogeneous ASICs

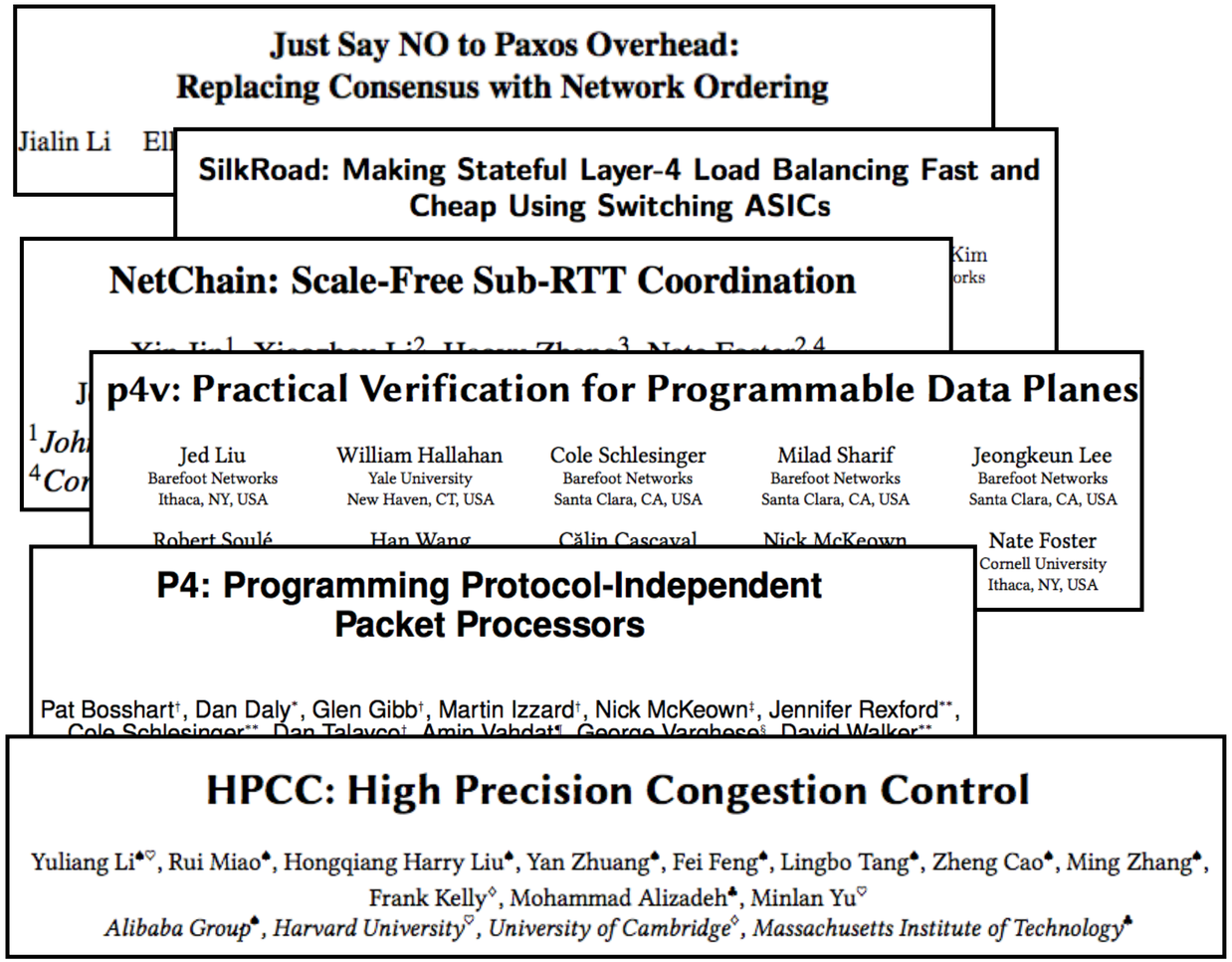
Jiaqi Gao, Ennan Zhai, Hongqiang Harry Liu, Rui Miao, Yu Zhou
Bingchuan Tian, Chen Sun, Dennis Cai, Ming Zhang, Minlan Yu

Alibaba Group 阿里巴巴集团
HARVARD UNIVERSITY
清華大學 Tsinghua University
南京大學 NANJING UNIVERSITY

Programmable switches gain significant traction

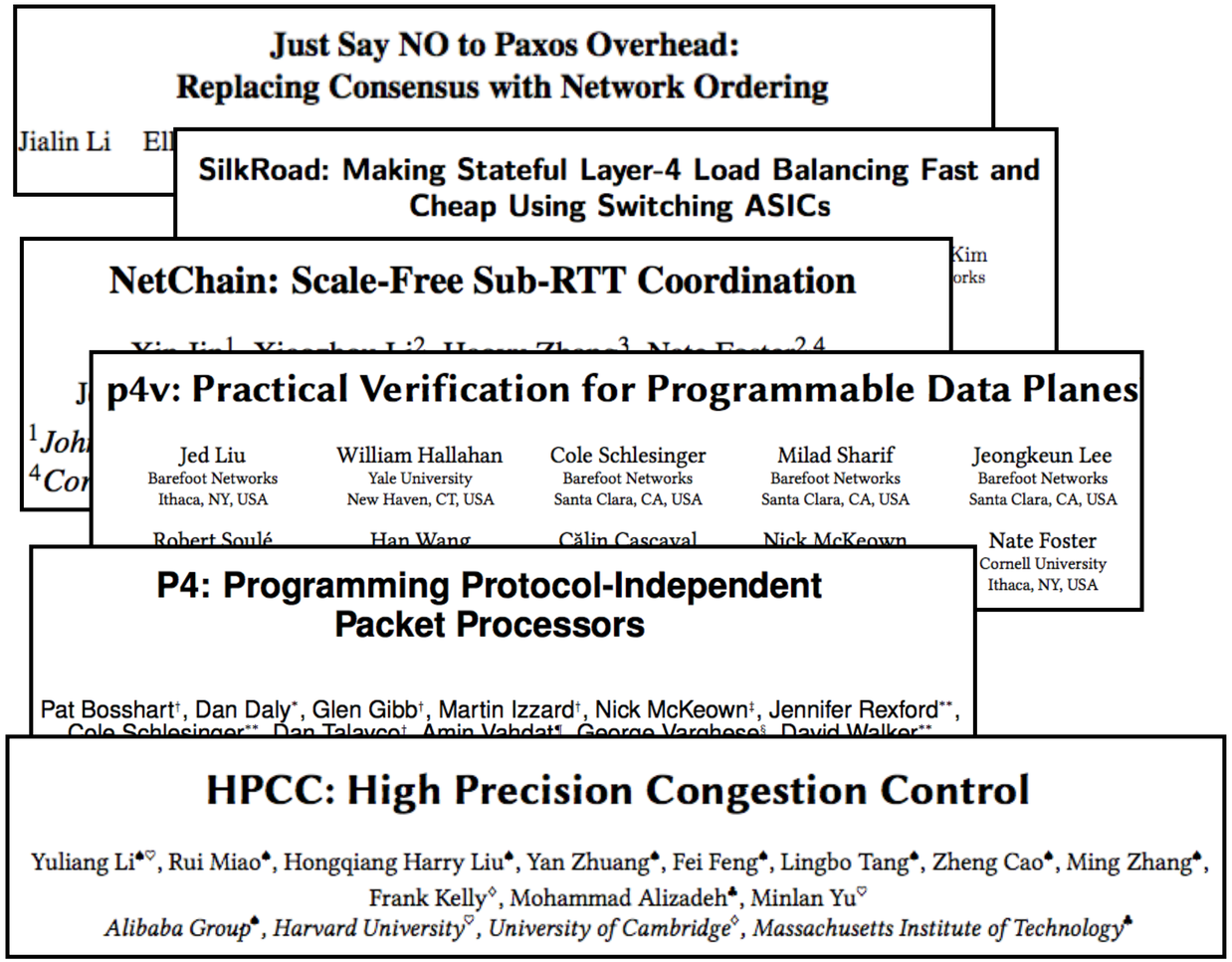
Programmable switches gain significant traction

Alibaba Cloud
Worldwide Cloud Services Partner



Programmable switches gain significant traction

Alibaba Cloud
Worldwide Cloud Services Partner



Just Say NO to Paxos Overhead:
Replacing Consensus with Network Ordering

Jialin Li El

SilkRoad: Making Stateful Layer-4 Load Balancing Fast and Cheap Using Switching ASICs

Kim
orks

NetChain: Scale-Free Sub-RTT Coordination

p4v: Practical Verification for Programmable Data Planes

Jed Liu	William Hallahan	Cole Schlesinger	Milad Sharif	Jeongkeun Lee
Barefoot Networks	Yale University	Barefoot Networks	Barefoot Networks	Barefoot Networks
Ithaca, NY, USA	New Haven, CT, USA	Santa Clara, CA, USA	Santa Clara, CA, USA	Santa Clara, CA, USA

Robert Soulé Han Wang Călin Cascaval Nick McKeown Nate Foster
Cornell University
Ithaca, NY, USA

P4: Programming Protocol-Independent Packet Processors

Pat Bosshart†, Dan Daly*, Glen Gibb†, Martin Izzard†, Nick McKeown‡, Jennifer Rexford**,

HPCC: High Precision Congestion Control

Yuliang Li♠♡, Rui Miao♠, Hongqiang Harry Liu♠, Yan Zhuang♠, Fei Feng♠, Lingbo Tang♠, Zheng Cao♠, Ming Zhang♠,
Frank Kelly♢, Mohammad Alizadeh♠, Minlan Yu♡
Alibaba Group♠, Harvard University♡, University of Cambridge♢, Massachusetts Institute of Technology♠

BROADCOM®

CISCO

Mellanox®
TECHNOLOGIES

BAREFOOT
NETWORKS

Data plane programming is still at an early stage …

Data plane programming is still at an early stage …

Chip specific languages ~ Assembly languages

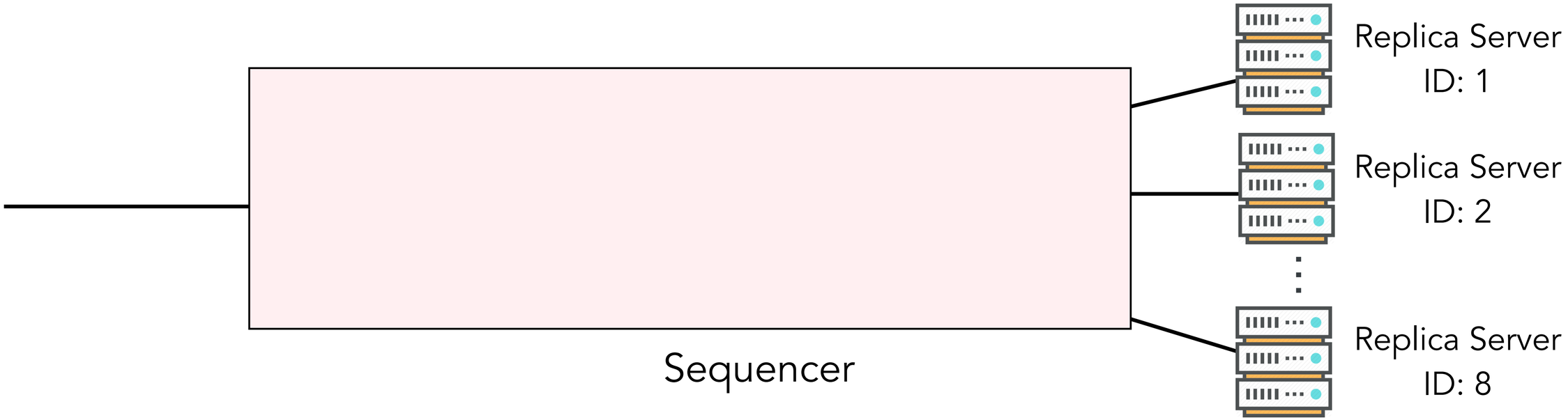
Running example: A network sequencer

Running example: A network sequencer

Linearizing the transactions for consensus protocols such as Paxos

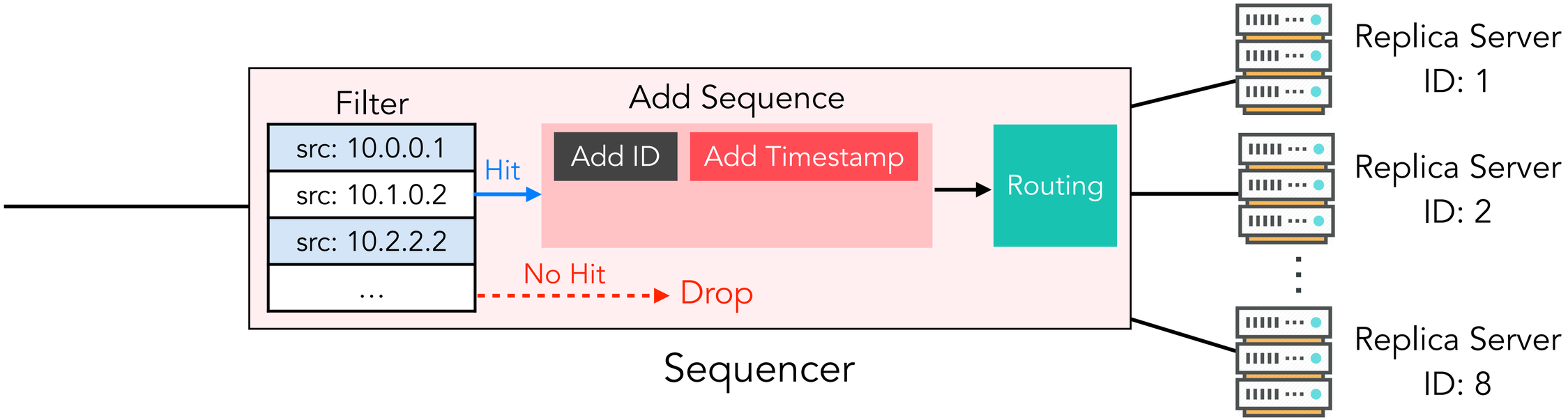
Running example: A network sequencer

Linearizing the transactions for consensus protocols such as Paxos



Running example: A network sequencer

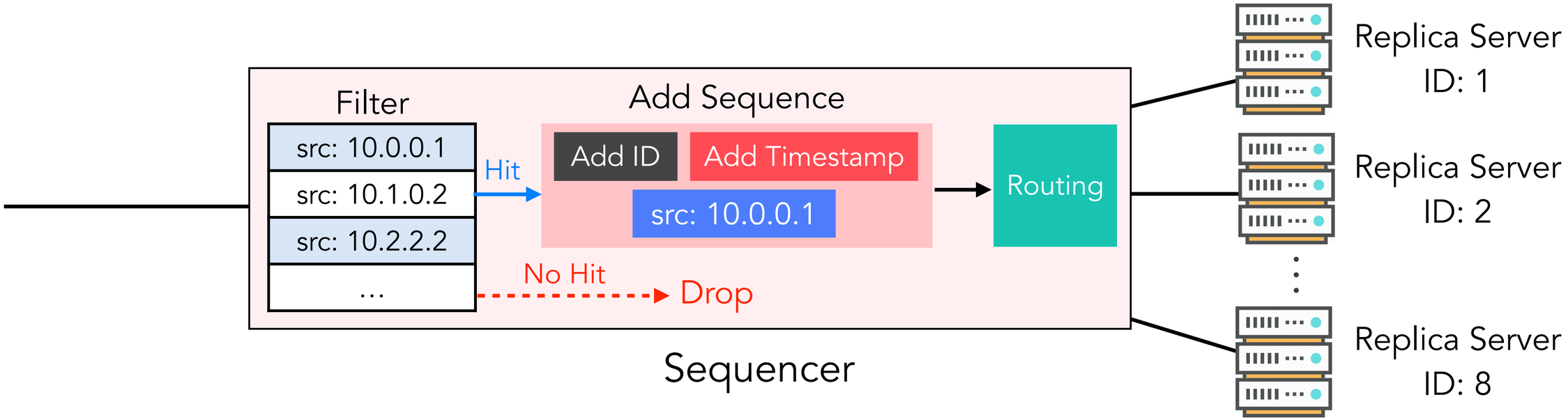
Linearizing the transactions for consensus protocols such as Paxos



Add sequence header to selected flows and drop others

Running example: A network sequencer

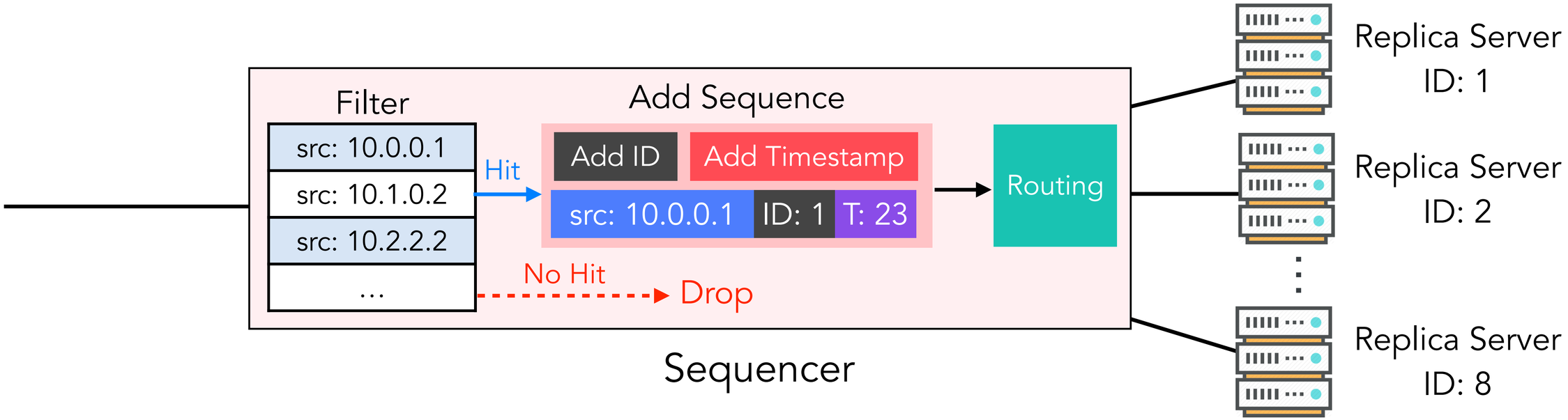
Linearizing the transactions for consensus protocols such as Paxos



Add sequence header to selected flows and drop others

Running example: A network sequencer

Linearizing the transactions for consensus protocols such as Paxos



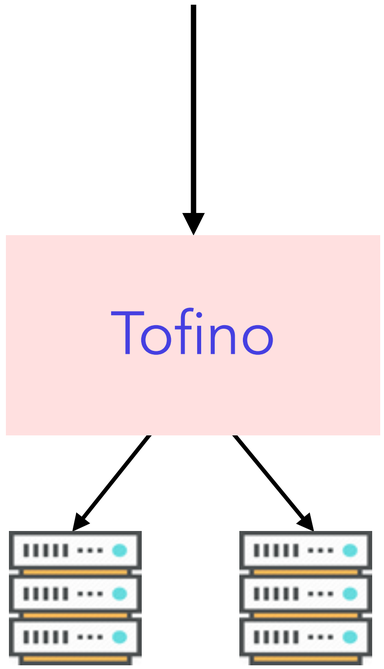
Add sequence header to selected flows and drop others

Problem 1: Portability

Low level, chip-specific languages

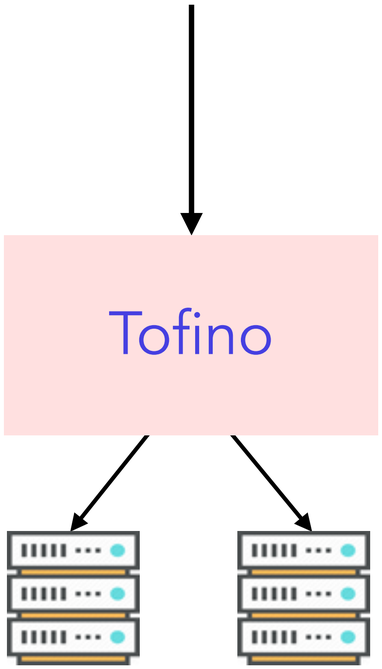
Problem 1: Portability

Low level, chip-specific languages



Problem 1: Portability

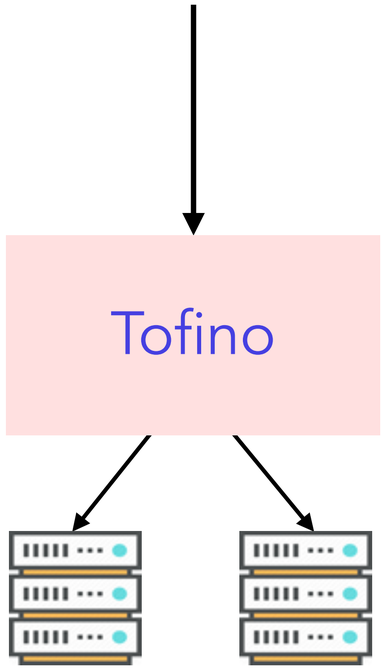
Low level, chip-specific languages




```
// P4_14
action a_shift_switch_id(server_id) {
    shift_left(sequence.seq, server_id, 28);
}
table shift_switch_id {
    reads {
        ipv4.src_ip: exact;
    }
    actions {
        a_shift_switch_id;
    }
}
action a_and_timestamp() {
    bit_and(ig_ts, ig_ts, 0x0FFFFFFF);
}
table and_timestamp {
    actions {
        a_and_timestamp;
    }
    default_action: a_and_timestamp();
}
action a_sequence_header() {
    add_header(sequence);
    bit_and(sequence.seq, sequence.seq, ig_ts);
}
table add_sequence_header {
    actions {
        a_sequence_header;
    }
    default_action: a_sequence_header();
}
```

Problem 1: Portability

Low level, chip-specific languages



Add ID

Add Timestamp

```
add_header(sequence);
sequence.seq = (server_id << 28) & (ig_ts & 0x0FFFFFFF);
```

```
// P4_14
action a_shift_switch_id(server_id) {
    shift_left(sequence.seq, server_id, 28);
}
table shift_switch_id {
    reads {
        ipv4.src_ip: exact;
    }
    actions {
        a_shift_switch_id;
    }
}
action a_and_timestamp() {
    bit_and(ig_ts, ig_ts, 0x0FFFFFFF);
}
table and_timestamp {
    actions {
        a_and_timestamp;
    }
    default_action: a_and_timestamp();
}
action a_sequence_header() {
    add_header(sequence);
    bit_and(sequence.seq, sequence.seq, ig_ts);
}
table add_sequence_header {
    actions {
        a_sequence_header;
    }
    default_action: a_sequence_header();
}
```

Problem 1: Portability

Low level, chip-specific languages

Chip Vendors

BROADCOM

CISCO

BAREFOOT NETWORKS

Mellanox TECHNOLOGIES

Add ID

Add Timestamp

```
add_header(sequence);
sequence.seq = (server_id << 28) & (ig_ts & 0x0FFFFFFF);
```

```
// P4_14
action a_shift_switch_id(server_id) {
    shift_left(sequence.seq, server_id, 28);
}
table shift_switch_id {
    reads {
        ipv4.src_ip: exact;
    }
    actions {
        a_shift_switch_id;
    }
}
action a_and_timestamp() {
    bit_and(ig_ts, ig_ts, 0x0FFFFFFF);
}
table and_timestamp {
    actions {
        a_and_timestamp;
    }
    default_action: a_and_timestamp();
}
action a_sequence_header() {
    add_header(sequence);
    bit_and(sequence.seq, sequence.seq, ig_ts);
}
table add_sequence_header {
    actions {
        a_sequence_header;
    }
    default_action: a_sequence_header();
}
```

Problem 1: Portability

Low level, chip-specific languages

Alibaba Cloud
Worldwide Cloud Services Partner

Chip Vendors

BROADCOM

CISCO

BAREFOOT NETWORKS

Mellanox TECHNOLOGIES

Languages

NPL

P4

Add ID

Add Timestamp

```
add_header(sequence);
sequence.seq = (server_id << 28) & (ig_ts & 0x0FFFFFFF);
```

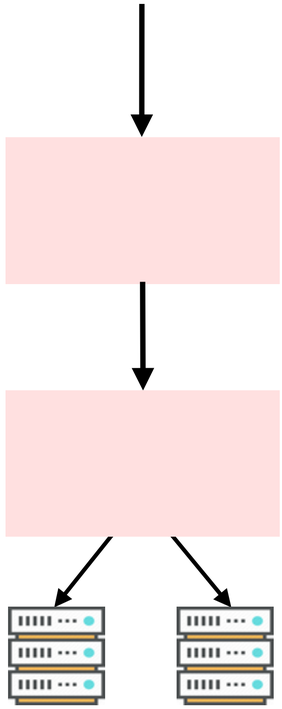
```
// P4_14
action a_shift_switch_id(server_id) {
    shift_left(sequence.seq, server_id, 28);
}
table shift_switch_id {
    reads {
        ipv4.src_ip: exact;
    }
    actions {
        a_shift_switch_id;
    }
}
action a_and_timestamp() {
    bit_and(ig_ts, ig_ts, 0x0FFFFFFF);
}
table and_timestamp {
    actions {
        a_and_timestamp;
    }
    default_action: a_and_timestamp();
}
action a_sequence_header() {
    add_header(sequence);
    bit_and(sequence.seq, sequence.seq, ig_ts);
}
table add_sequence_header {
    actions {
        a_sequence_header;
    }
    default_action: a_sequence_header();
}
```

Problem 2: Extensibility

Cannot program across switches

Problem 2: Extensibility

Cannot program across switches

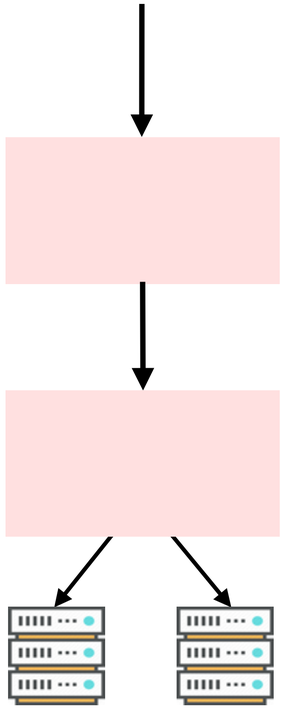


Problem 2: Extensibility

Cannot program across switches

Aggregation
1000 entries

Top of Rack
1000 entries



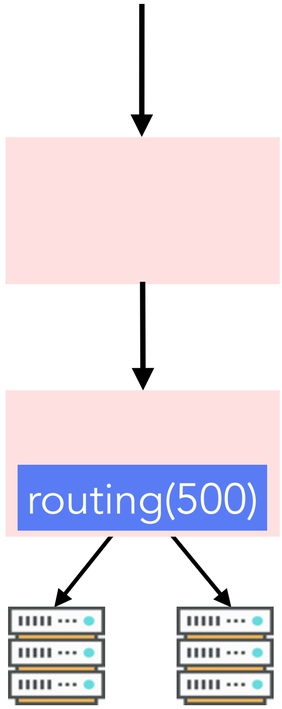
Filter table

Problem 2: Extensibility

Cannot program across switches

Aggregation
1000 entries

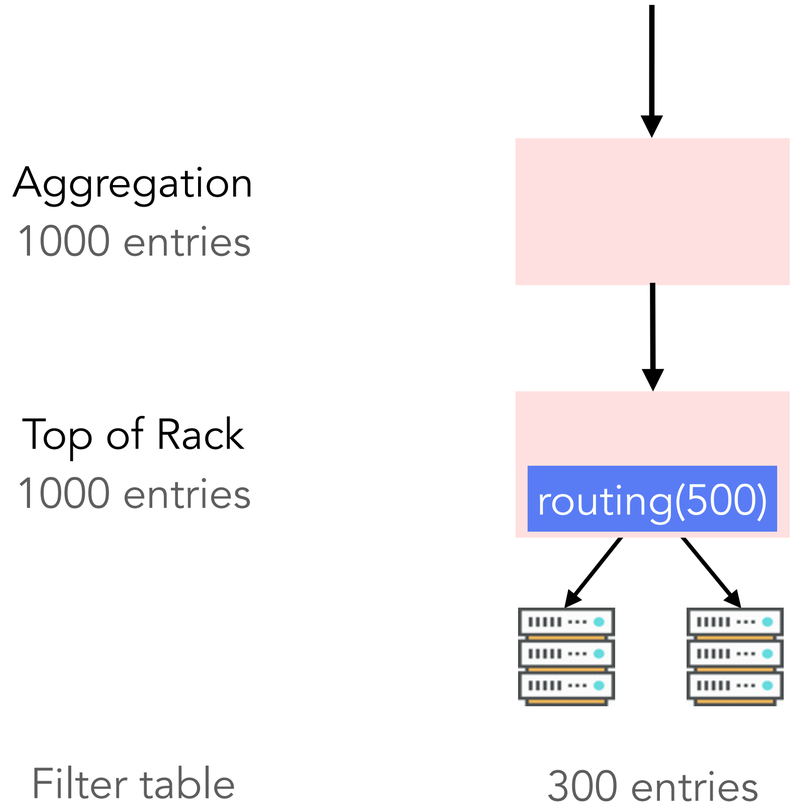
Top of Rack
1000 entries



Filter table

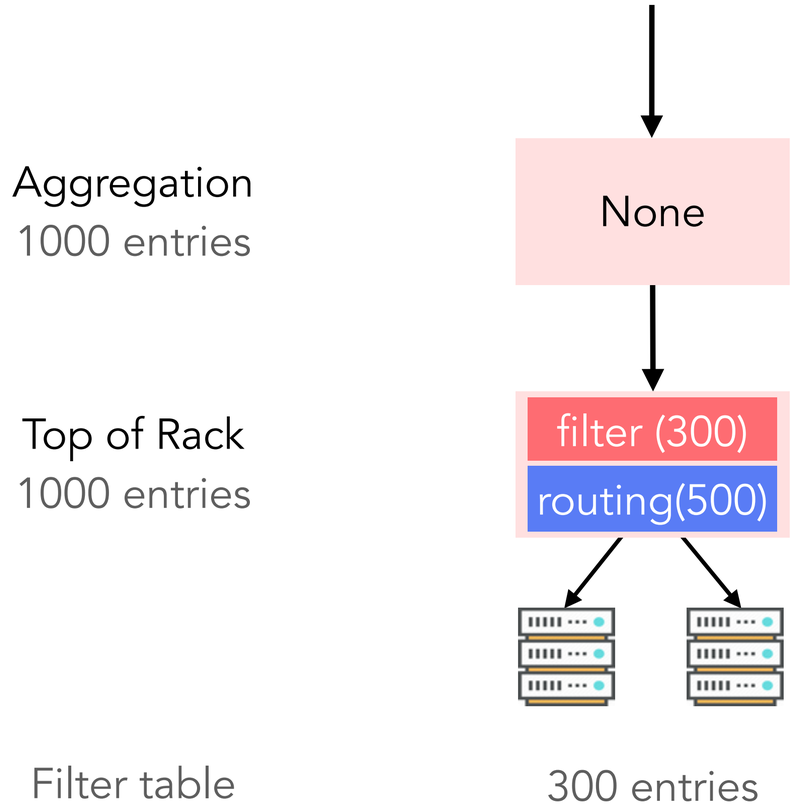
Problem 2: Extensibility

Cannot program across switches



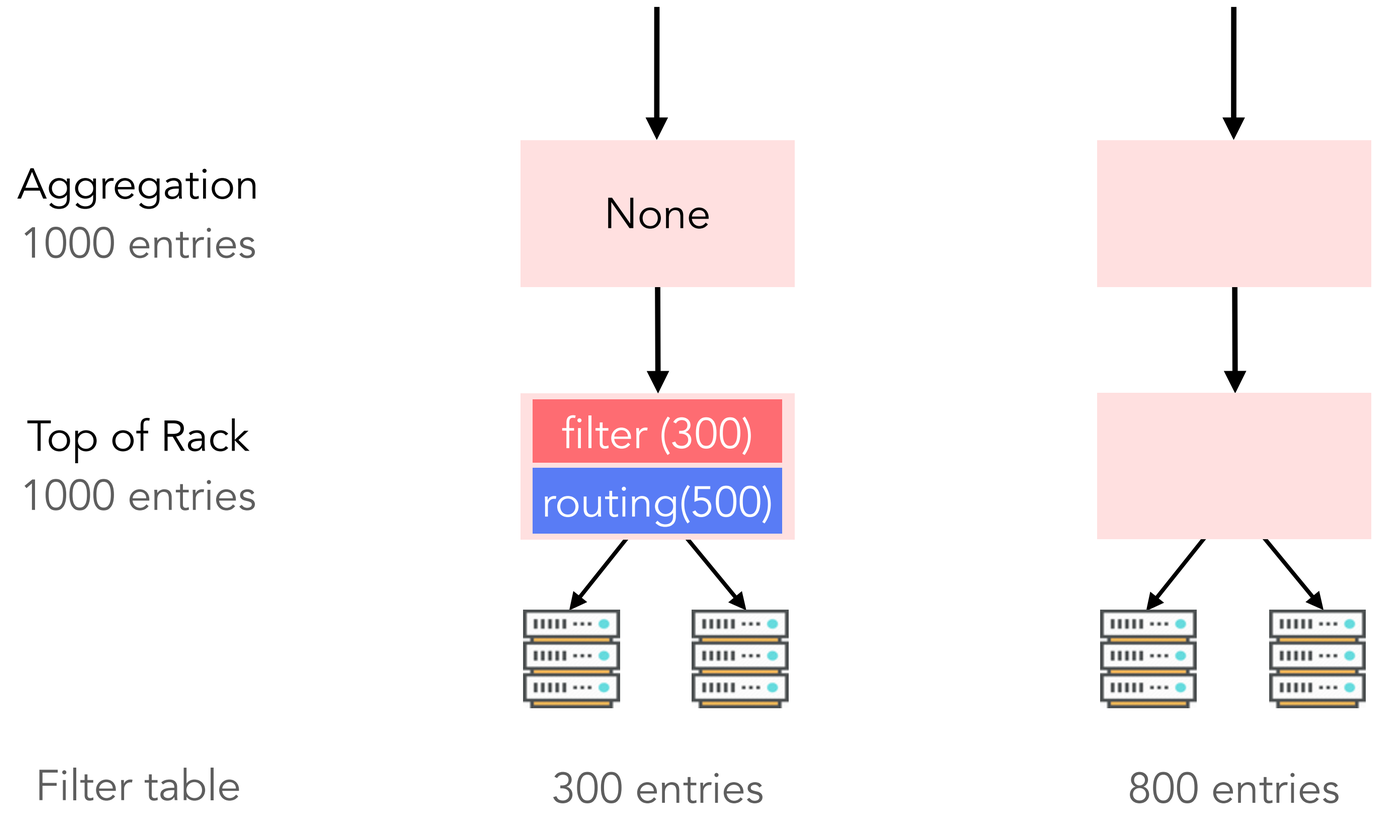
Problem 2: Extensibility

Cannot program across switches



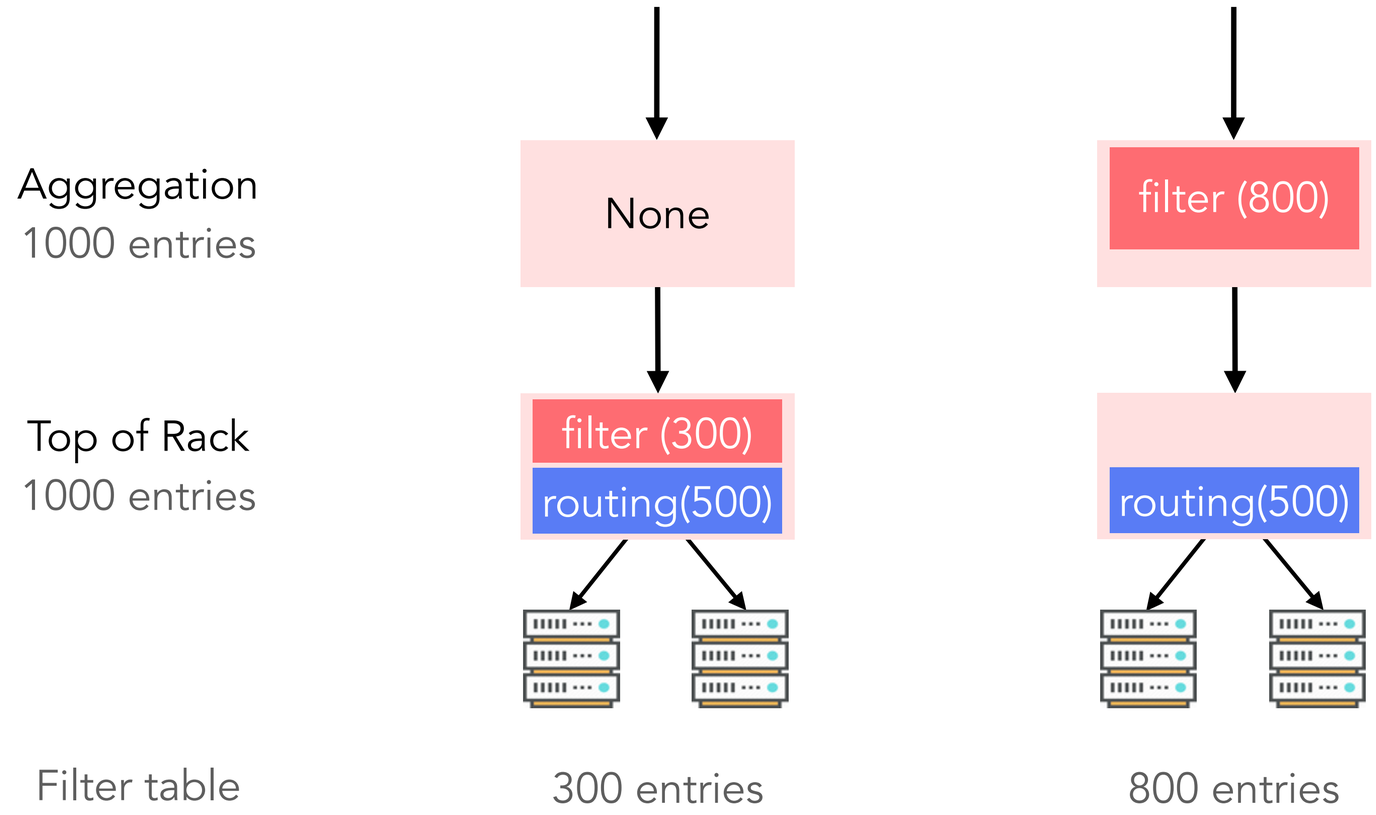
Problem 2: Extensibility

Cannot program across switches



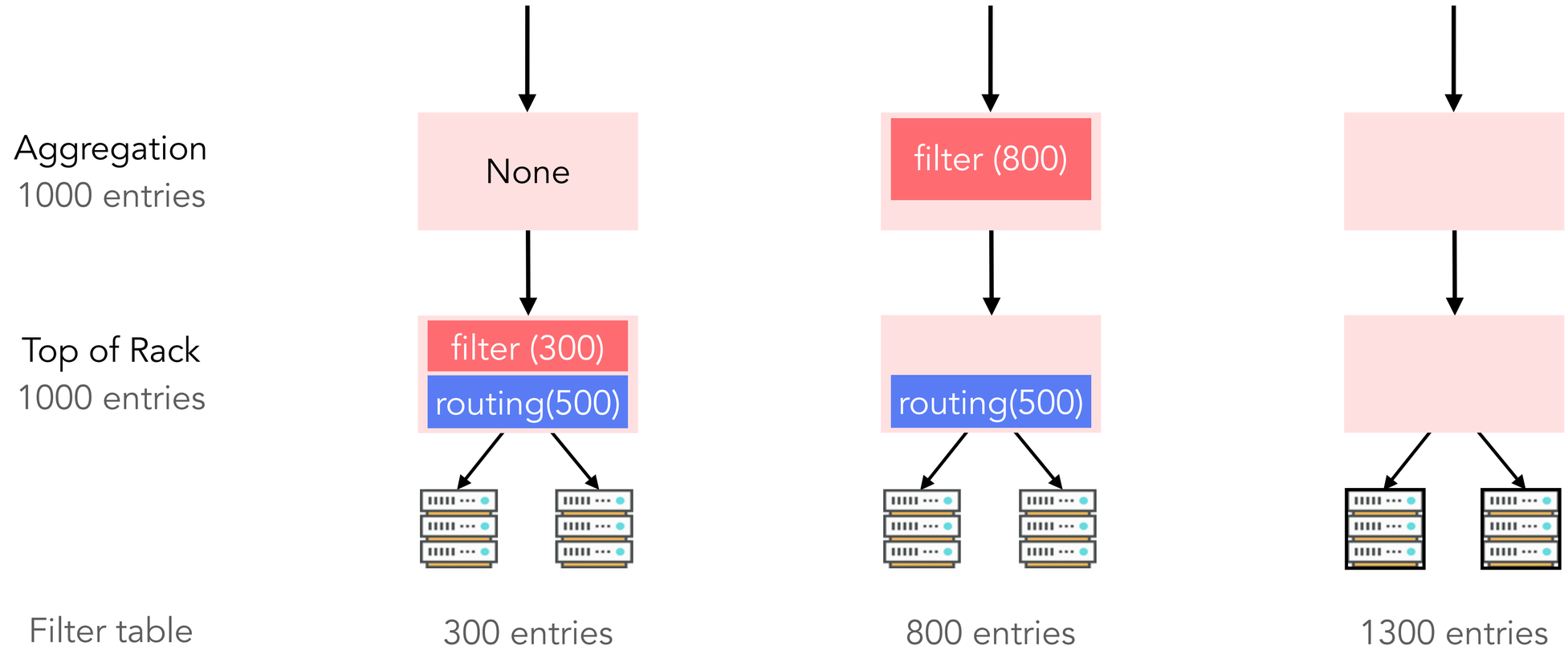
Problem 2: Extensibility

Cannot program across switches



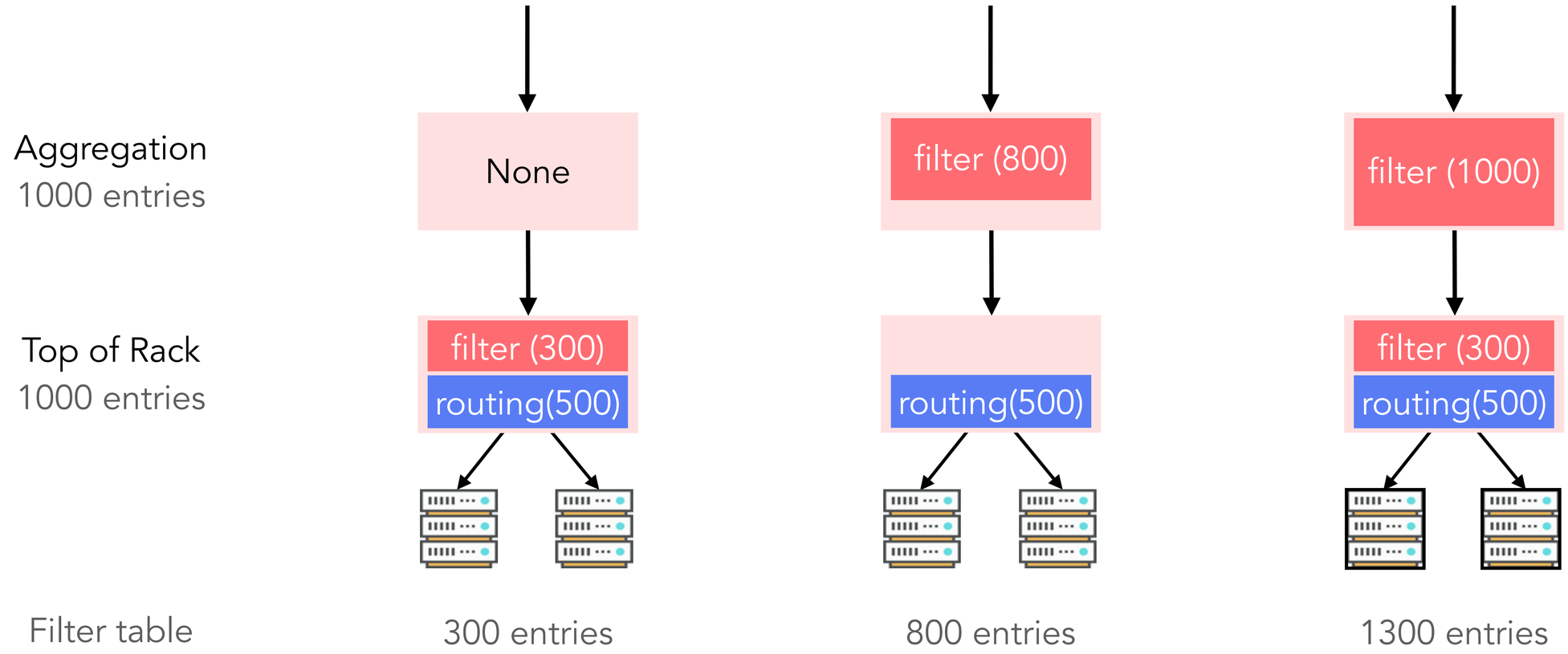
Problem 2: Extensibility

Cannot program across switches



Problem 2: Extensibility

Cannot program across switches



Problem 3: Composition

Co-locate with other programs

Problem 3: Composition

Co-locate with other programs

Add an ARP table with 300 entries to the ToR switch

Problem 3: Composition

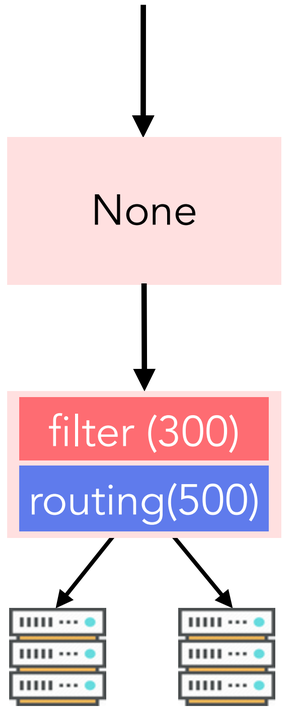
Co-locate with other programs

Alibaba Cloud
Worldwide Cloud Services Partner

Add an ARP table with 300 entries to the ToR switch

Aggregation
1000 entries

Top of Rack
1000 entries



Problem 3: Composition

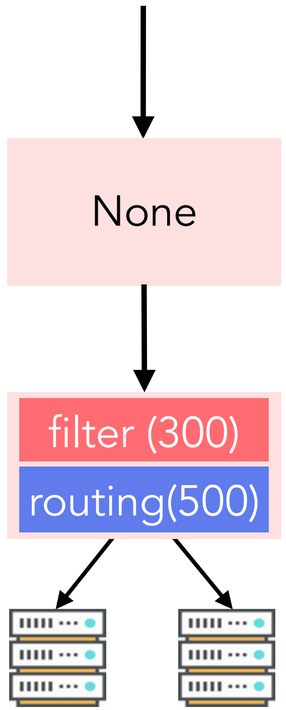
Co-locate with other programs

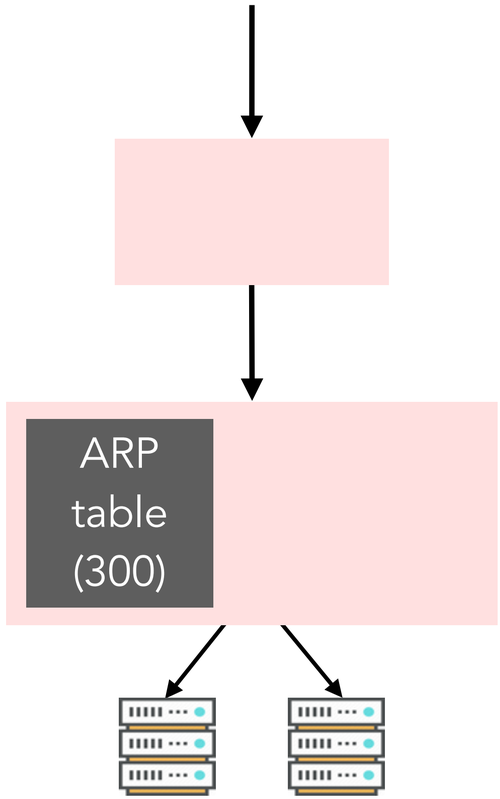
Alibaba Cloud
Worldwide Cloud Services Partner

Add an ARP table with 300 entries to the ToR switch

Aggregation
1000 entries

Top of Rack
1000 entries





Problem 3: Composition

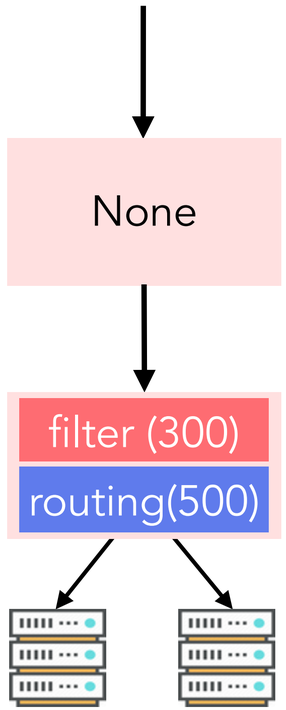
Co-locate with other programs

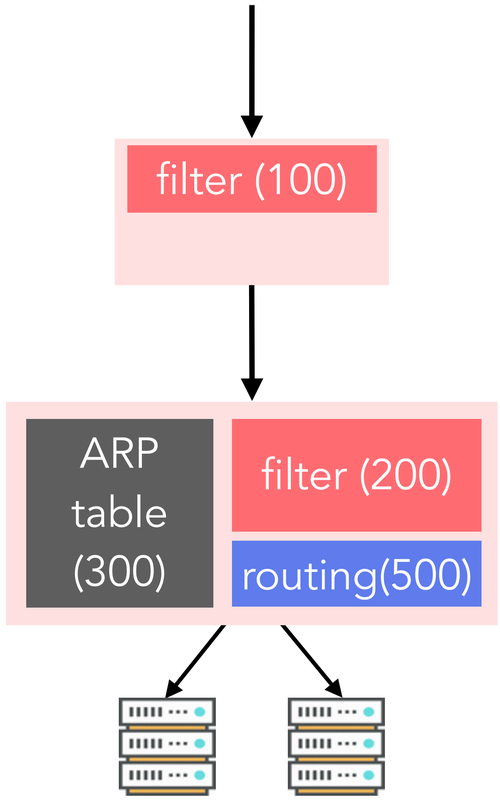
Alibaba Cloud
Worldwide Cloud Services Partner

Add an ARP table with 300 entries to the ToR switch

Aggregation
1000 entries

Top of Rack
1000 entries





Data plane programming is still at an early stage

Portability

Migrate program across switches

Different languages
Different architectures
Different ASIC models

Extensibility

Distribute program across multiple switches

Different roles (In-band network telemetry)
Expand memory or computation (Sequencer)

Composition

Fit multiple programs into one switch

Function overlapping

Data plane programming is still at an early stage

Portability

Migrate program across switches

Different languages
Different architectures
Different ASIC models

Extensibility

Distribute program across multiple switches

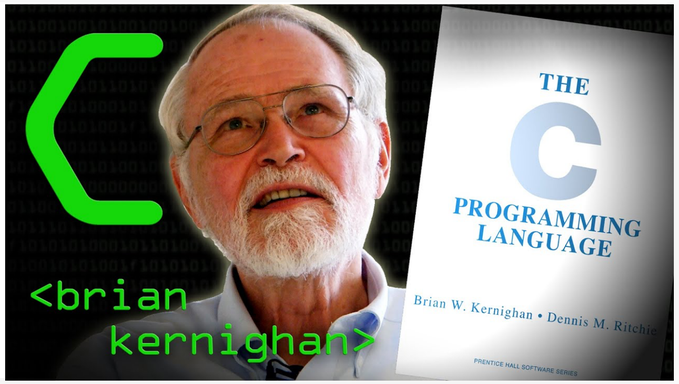
Different roles (In-band network telemetry)
Expand memory or computation (Sequencer)

Composition

Fit multiple programs into one switch

Function overlapping

> C language lets you get close to the machine,
> without getting tied up in the machine
>
> — **Dr. Brain Kernighan**



Lyra: A high-level data plane language & compiler

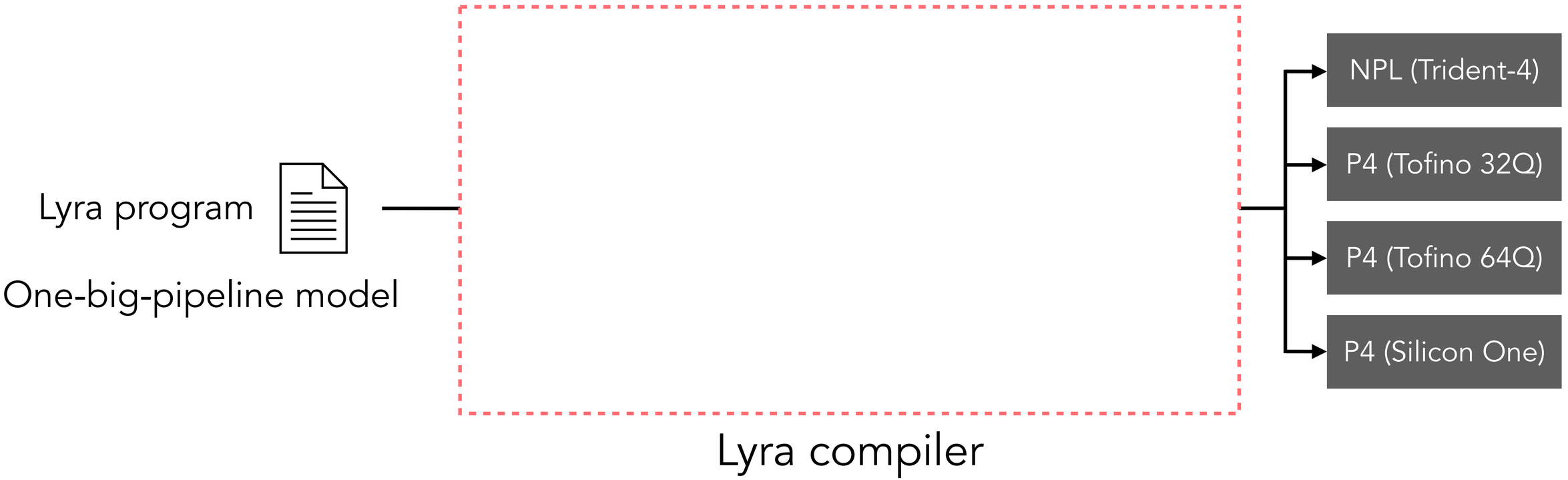
Lyra: A high-level data plane language & compiler

Alibaba Cloud
Worldwide Cloud Services Partner

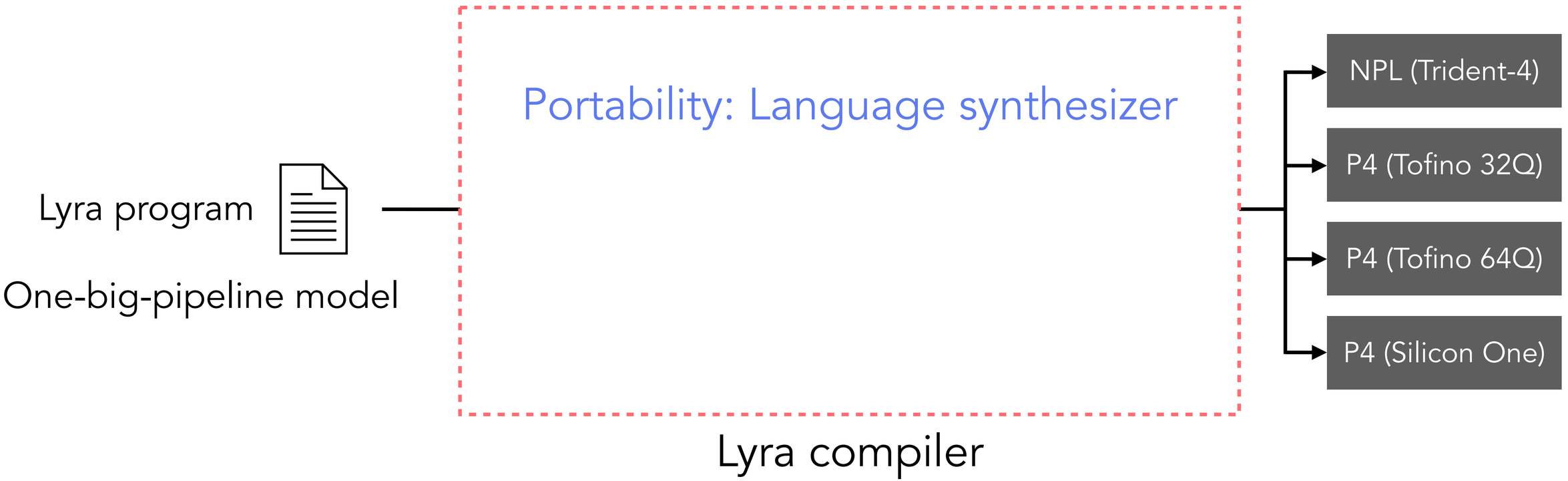
Lyra program 

One-big-pipeline model

Lyra: A high-level data plane language & compiler

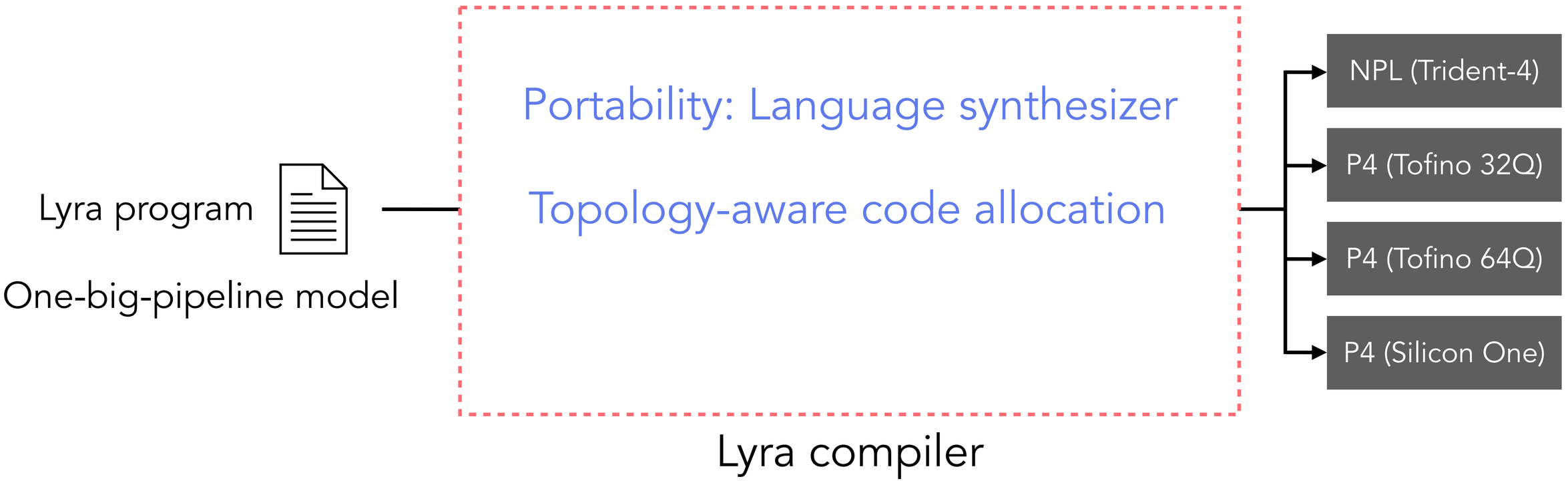


Lyra: A high-level data plane language & compiler

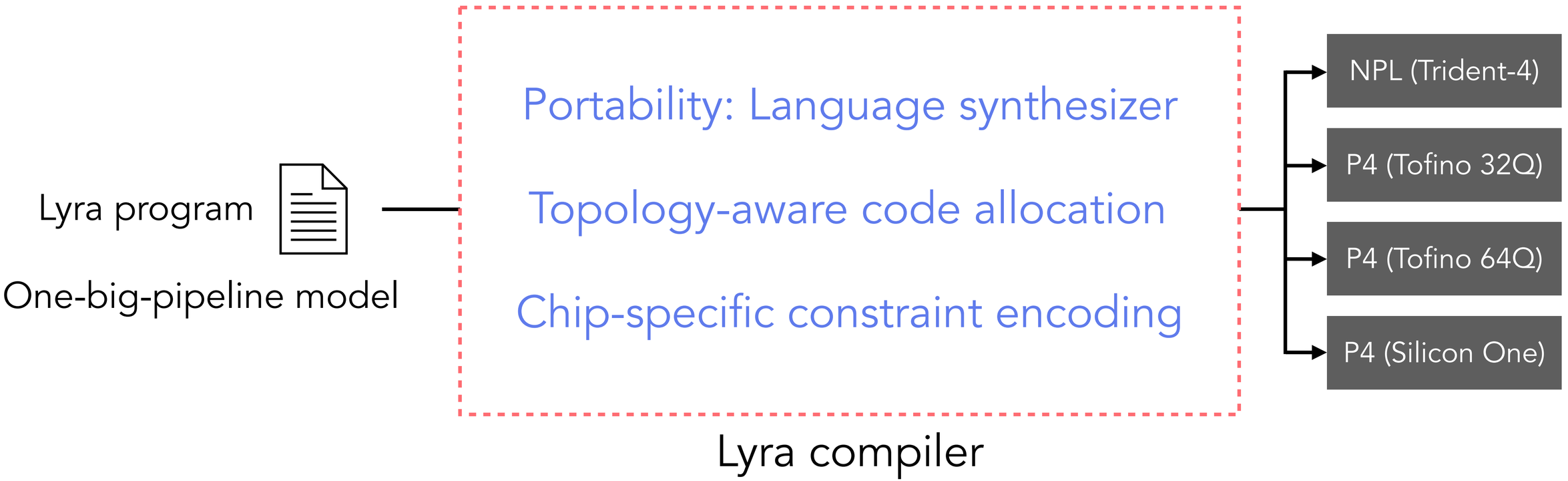


Lyra: A high-level data plane language & compiler

Alibaba Cloud
Worldwide Cloud Services Partner

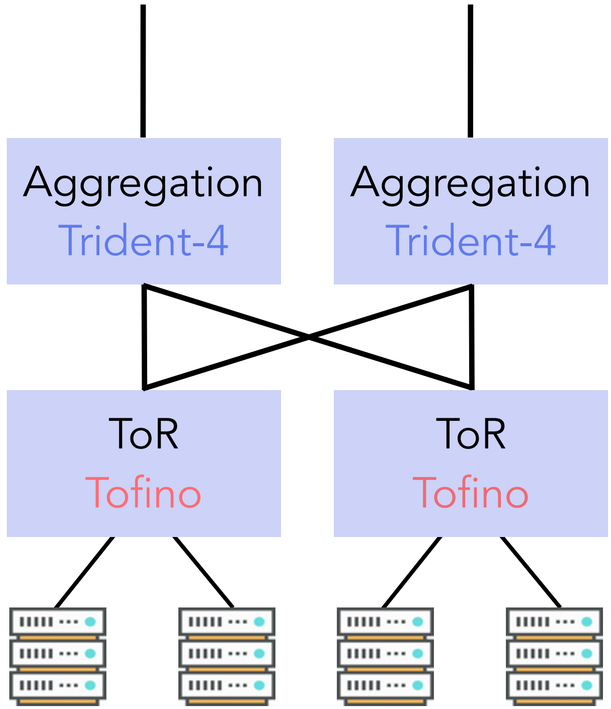


Lyra: A high-level data plane language & compiler



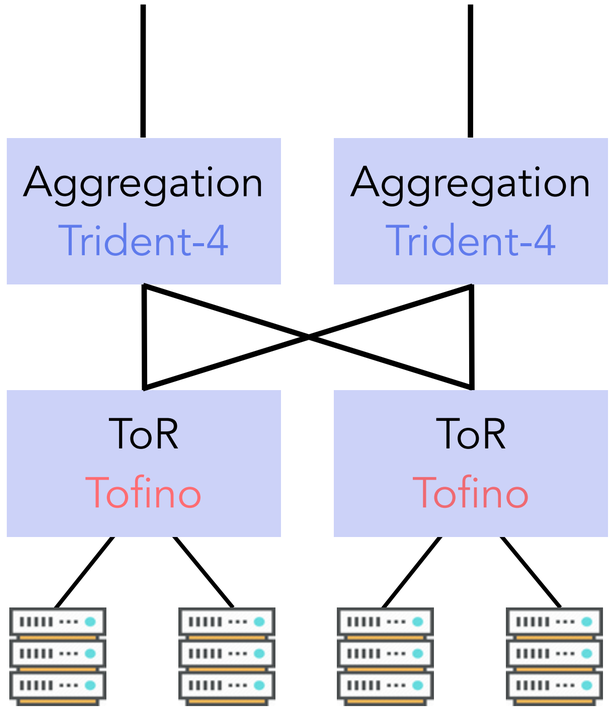
Lyra language: One-big-pipeline model

Lyra language: One-big-pipeline model

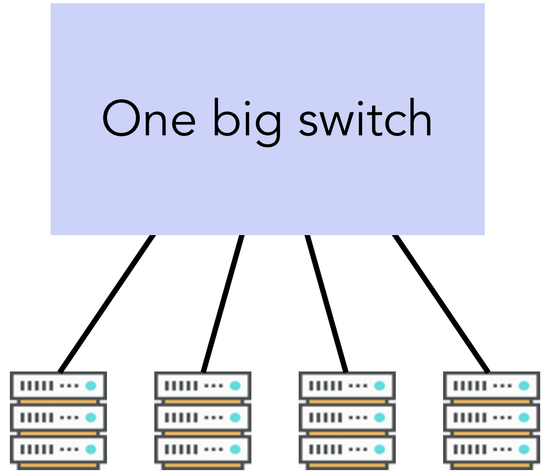


Per-switch model

Lyra language: One-big-pipeline model

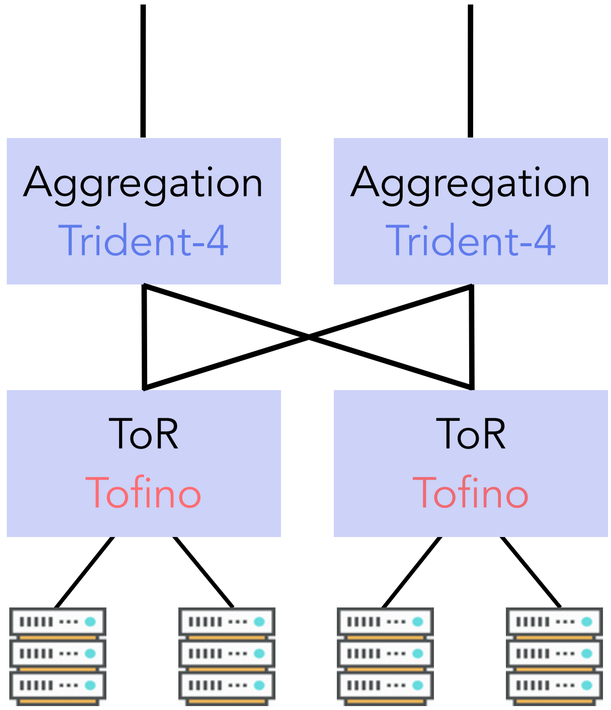


Per-switch model

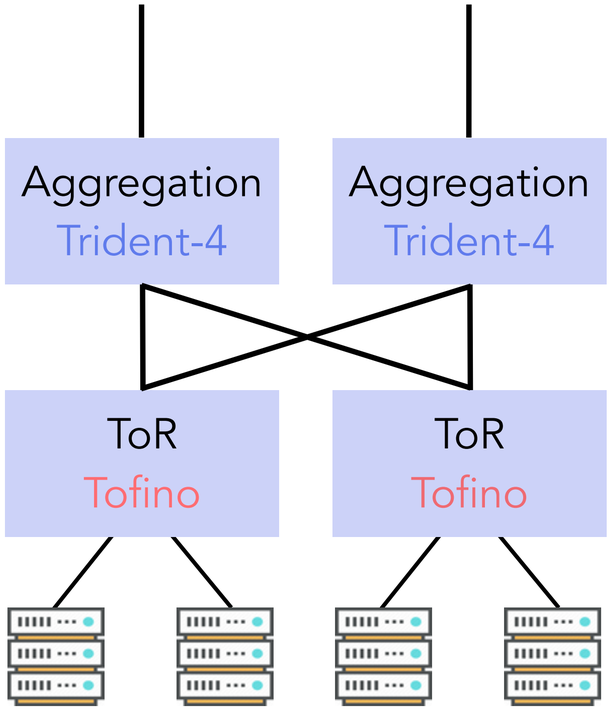


One-big-switch model

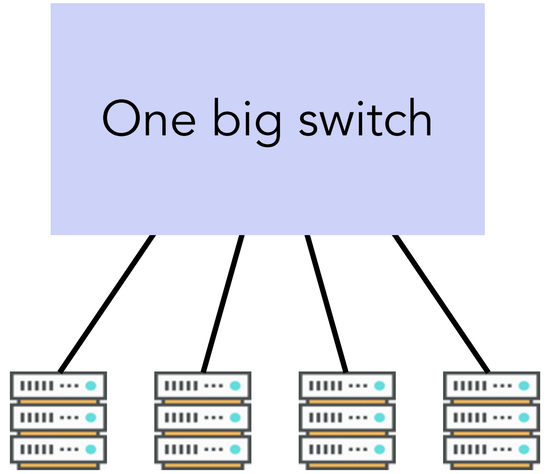
Lyra language: One-big-pipeline model



Per-switch model

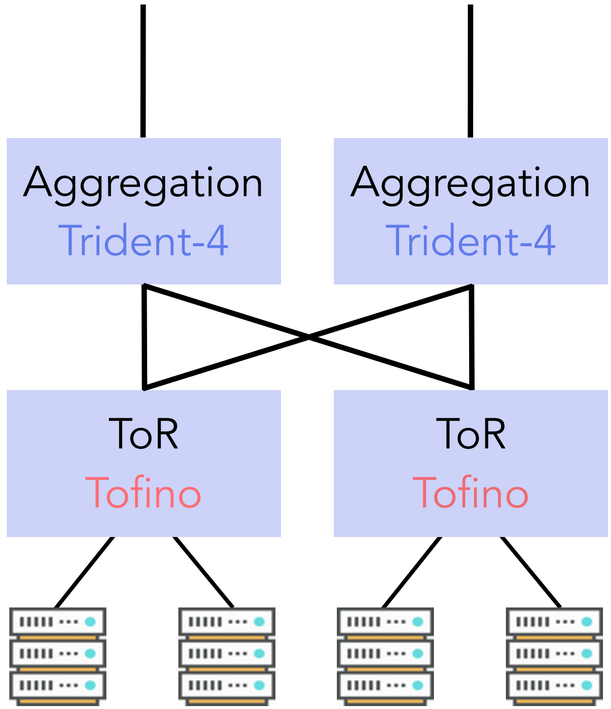


One-big-pipeline model

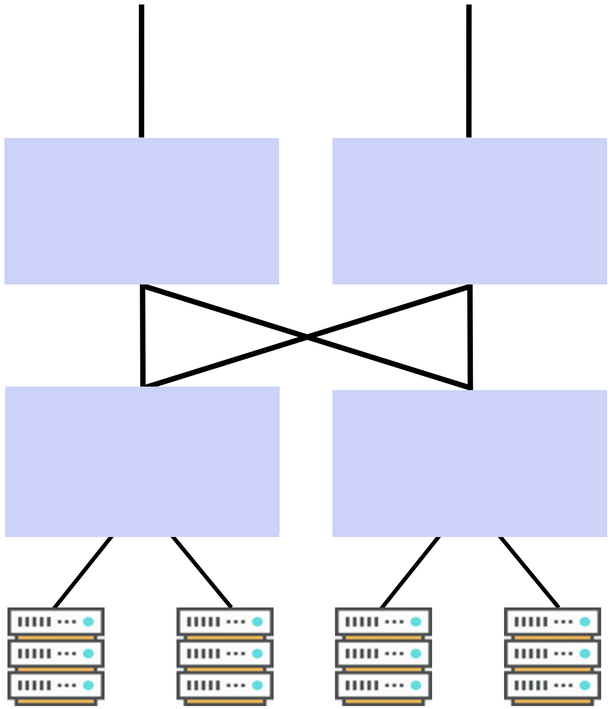


One-big-switch model

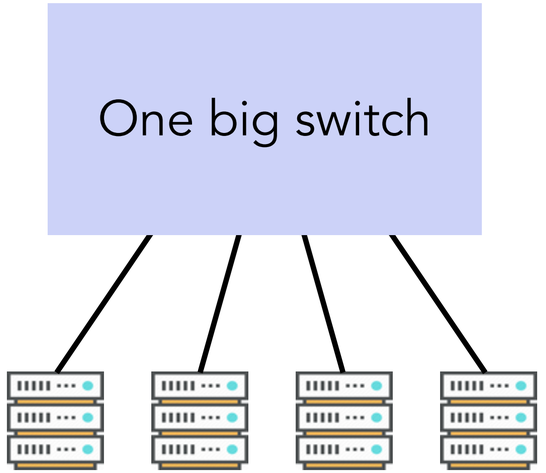
Lyra language: One-big-pipeline model



Per-switch model

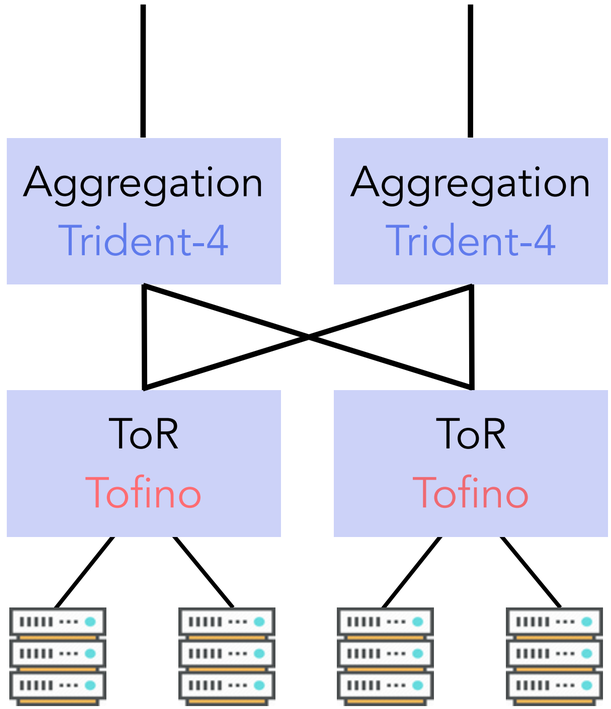


One-big-pipeline model

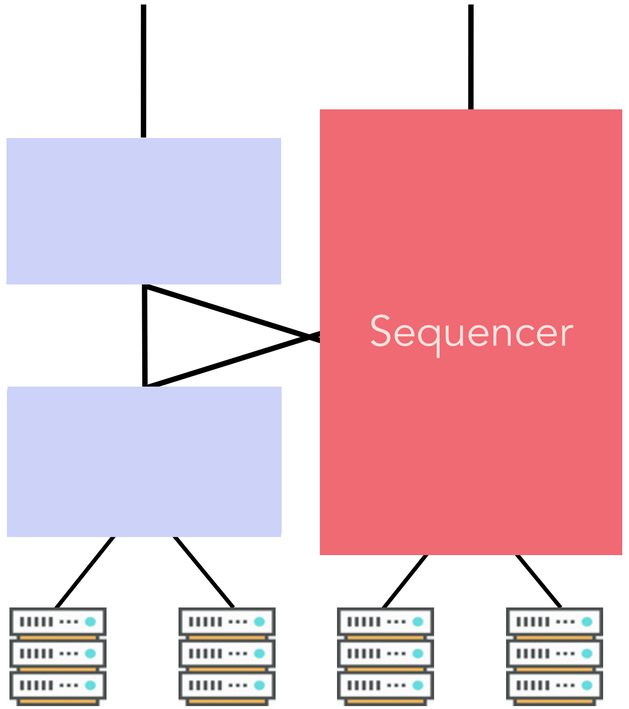


One-big-switch model

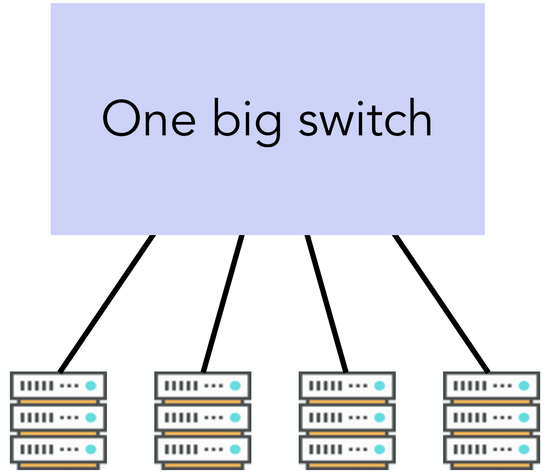
Lyra language: One-big-pipeline model



Per-switch model

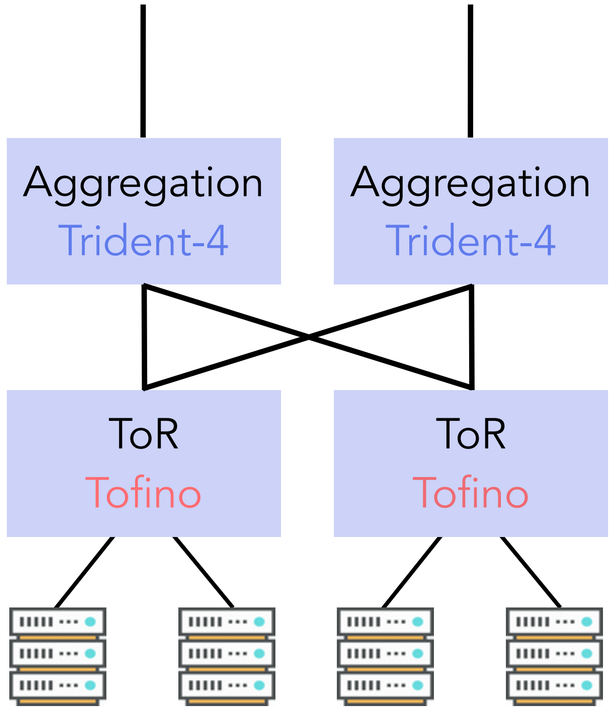


One-big-pipeline model

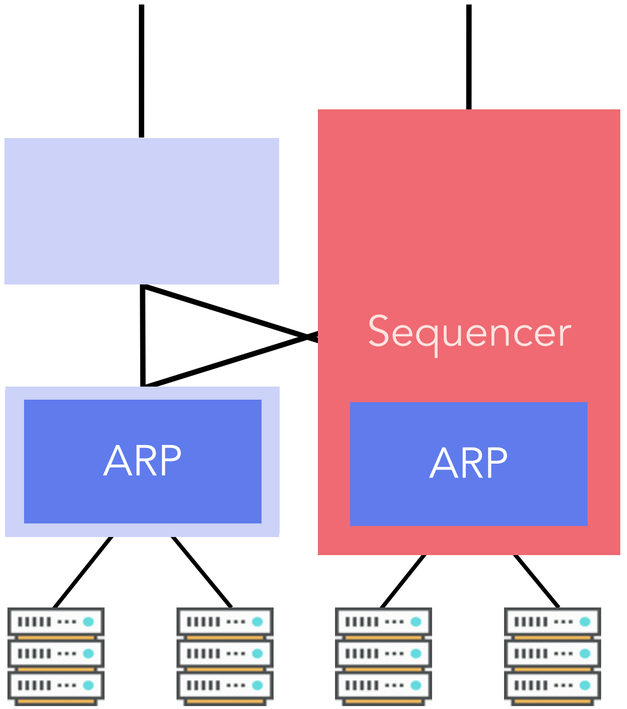


One-big-switch model

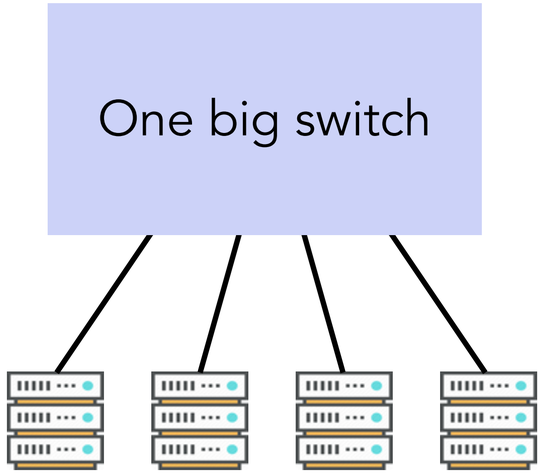
Lyra language: One-big-pipeline model



Per-switch model



One-big-pipeline model



One-big-switch model

Lyra language: Program

Alibaba Cloud
Worldwide Cloud Services Partner

```
// Sequencer.lyra
header_type sequence_t{
    bit[32] seq;
}
parser_node parse_sequence{
    extract_fields(sequence);
}

pipeline[SEQ] {sequencer};

algorithm sequencer {
    add_sequence();
    routing();
}

func add_sequence() {
    extern dict<bit[32] ip, bit[4] id>[300] filter;
    if (ipv4.src_ip in filter) {
        add_header(sequence);
        sequence.seq = (filter[ipv4.src_ip] << 28) \
                        & (ig_ts & 0x0FFFFFFF);
    }
    else {
        drop();
    }
}
```

Lyra language: Program

Alibaba Cloud
Worldwide Cloud Services Partner

```
// Sequencer.lyra
header_type sequence_t{
    bit[32] seq;
}
parser_node parse_sequence{
    extract_fields(sequence);
}

pipeline[SEQ] {sequencer};

algorithm sequencer {
    add_sequence();
    routing();
}

func add_sequence() {
    extern dict<bit[32] ip, bit[4] id>[300] filter;
    if (ipv4.src_ip in filter) {
        add_header(sequence);
        sequence.seq = (filter[ipv4.src_ip] << 28) \
                       & (ig_ts & 0x0FFFFFFF);
    }
    else {
        drop();
    }
}
```

Lyra language: Program

Alibaba Cloud
Worldwide Cloud Services Partner

```
// Sequencer.lyra
header_type sequence_t{
    bit[32] seq;
}
parser_node parse_sequence{
    extract_fields(sequence);
}

pipeline[SEQ] {sequencer};

algorithm sequencer {
    add_sequence();
    routing();
}

func add_sequence() {
    extern dict<bit[32] ip, bit[4] id>[300] filter;
    if (ipv4.src_ip in filter) {
        add_header(sequence);
        sequence.seq = (filter[ipv4.src_ip] << 28) \
                       & (ig_ts & 0x0FFFFFFF);
    }
    else {
        drop();
    }
}
```

- Packet
  - Header
  - Parser

Lyra language: Program


```
// Sequencer.lyra
header_type sequence_t{
    bit[32] seq;
}
parser_node parse_sequence{
    extract_fields(sequence);
}

pipeline[SEQ] {sequencer};

algorithm sequencer {
    add_sequence();
    routing();
}

func add_sequence() {
    extern dict<bit[32] ip, bit[4] id>[300] filter;
    if (ipv4.src_ip in filter) {
        add_header(sequence);
        sequence.seq = (filter[ipv4.src_ip] << 28) \
                       & (ig_ts & 0x0FFFFFFF);
    }
    else {
        drop();
    }
}
```

- Packet
  - Header
  - Parser

Lyra language: Program

Alibaba Cloud
Worldwide Cloud Services Partner

```
// Sequencer.lyra
header_type sequence_t{
    bit[32] seq;
}
parser_node parse_sequence{
    extract_fields(sequence);
}

pipeline[SEQ] {sequencer};

algorithm sequencer {
    add_sequence();
    routing();
}

func add_sequence() {
    extern dict<bit[32] ip, bit[4] id>[300] filter;
    if (ipv4.src_ip in filter) {
        add_header(sequence);
        sequence.seq = (filter[ipv4.src_ip] << 28) \
                       & (ig_ts & 0x0FFFFFFF);
    }
    else {
        drop();
    }
}
```

- Packet
  - Header
  - Parser
- Pipeline
  - Algorithm

Lyra language: Program

Alibaba Cloud
Worldwide Cloud Services Partner

```
// Sequencer.lyra
header_type sequence_t{
    bit[32] seq;
}
parser_node parse_sequence{
    extract_fields(sequence);
}

pipeline[SEQ] {sequencer};

algorithm sequencer {
    add_sequence();
    routing();
}

func add_sequence() {
    extern dict<bit[32] ip, bit[4] id>[300] filter;
    if (ipv4.src_ip in filter) {
        add_header(sequence);
        sequence.seq = (filter[ipv4.src_ip] << 28) \
                        & (ig_ts & 0x0FFFFFFF);
    }
    else {
        drop();
    }
}
```

- Packet
  - Header
  - Parser
- Pipeline
  - Algorithm

Lyra language: Program

Alibaba Cloud
Worldwide Cloud Services Partner

```
// Sequencer.lyra
header_type sequence_t{
    bit[32] seq;
}
parser_node parse_sequence{
    extract_fields(sequence);
}

pipeline[SEQ] {sequencer};

algorithm sequencer {
    add_sequence();
    routing();
}

func add_sequence() {
    extern dict<bit[32] ip, bit[4] id>[300] filter;
    if (ipv4.src_ip in filter) {
        add_header(sequence);
        sequence.seq = (filter[ipv4.src_ip] << 28) \
                        & (ig_ts & 0x0FFFFFFF);
    }
    else {
        drop();
    }
}
```

- Packet
  - Header
  - Parser
- Pipeline
  - Algorithm
- Function

Lyra language: Program

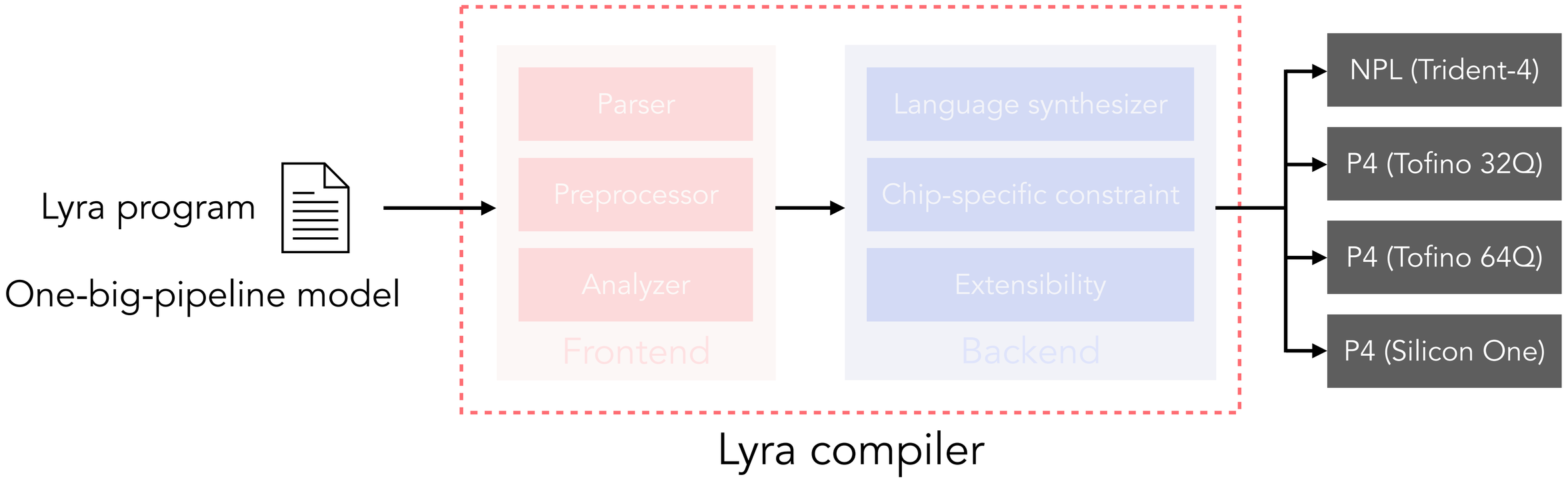
Alibaba Cloud
Worldwide Cloud Services Partner

```
// Sequencer.lyra
header_type sequence_t{
    bit[32] seq;
}
parser_node parse_sequence{
    extract_fields(sequence);
}

pipeline[SEQ] {sequencer};

algorithm sequencer {
    add_sequence();
    routing();
}

func add_sequence() {
    extern dict<bit[32] ip, bit[4] id>[300] filter;
    if (ipv4.src_ip in filter) {
        add_header(sequence);
        sequence.seq = (filter[ipv4.src_ip] << 28) \
                       & (ig_ts & 0x0FFFFFFF);
    }
    else {
        drop();
    }
}
```

- Packet
  - Header
  - Parser
- Pipeline
  - Algorithm
- Function

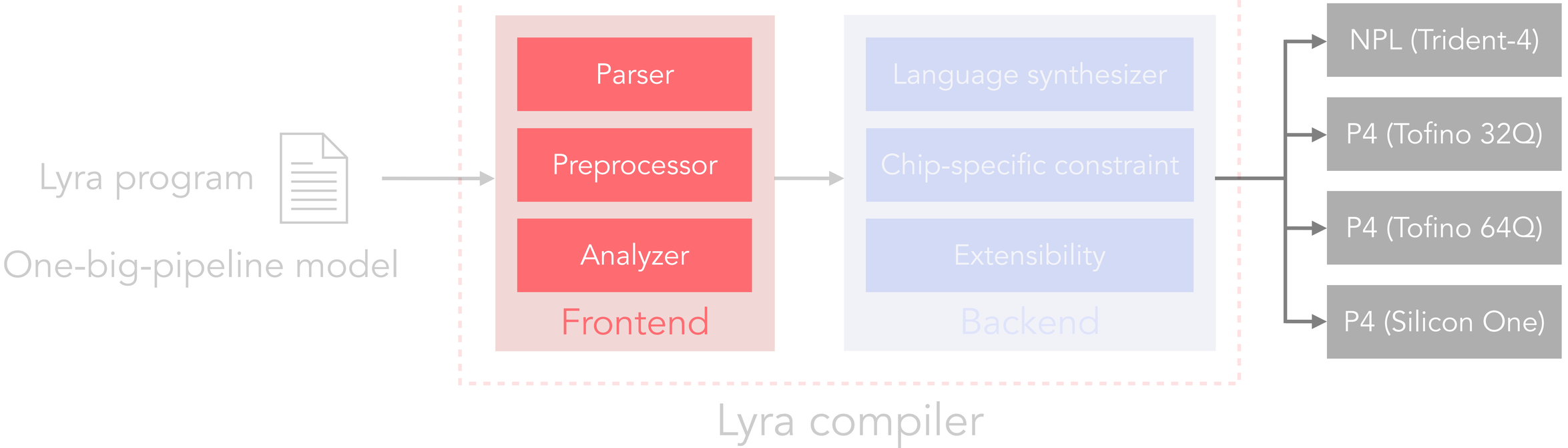
Lyra: A high-level data plane language & compiler

Alibaba Cloud
Worldwide Cloud Services Partner



Lyra: A high-level data plane language & compiler

Alibaba Cloud
Worldwide Cloud Services Partner



Frontend

Frontend


```
extern list<bit[32] ip>[300] filter;
if (ipv4.src_ip in filter) {
    add_header(sequence);
    sequence.seq = (filter[ipv4.src_ip] << 28) \
                   & (ig_ts & 0x0FFFFFFF);
}
else {
    drop();
}
```

Lyra program

One big pipeline model

Frontend


```
extern list<bit[32] ip>[300] filter;
if (ipv4.src_ip in filter) {
    add_header(sequence);
    sequence.seq = (filter[ipv4.src_ip] << 28) \
                     & (ig_ts & 0x0FFFFFFF);
}
else {
    drop();
}
```

Lyra program

```
p1 = ipv4.src_ip in filter;
    add_header(sequence);
    tmp1 = filter[ipv4.src_ip] << 28;
    tmp2 = ig_ts & 0x0FFFFFFF;
    sequence.seq = tmp1 & tmp2;
p2 = !(ipv4.src_ip in filter);
    drop();
```

Intermediate representation

One big pipeline model

Frontend


```
extern list<bit[32] ip>[300] filter;
if (ipv4.src_ip in filter) {
    add_header(sequence);
    sequence.seq = (filter[ipv4.src_ip] << 28) \
                   & (ig_ts & 0x0FFFFFFF);
}
else {
    drop();
}
```

Lyra program

```
p1 = ipv4.src_ip in filter;
    add_header(sequence);
    tmp1 = filter[ipv4.src_ip] << 28;
    tmp2 = ig_ts & 0x0FFFFFFF;
    sequence.seq = tmp1 & tmp2;
p2 = !(ipv4.src_ip in filter);
    drop();
```

Intermediate representation

One big pipeline model

No complex statements

Frontend


```
extern list<bit[32] ip>[300] filter;
if (ipv4.src_ip in filter) {
    add_header(sequence);
    sequence.seq = (filter[ipv4.src_ip] << 28) \
                   & (ig_ts & 0x0FFFFFFF);
}
else {
    drop();
}
```

Lyra program

```
p1 = ipv4.src_ip in filter;
    add_header(sequence);
    tmp1 = filter[ipv4.src_ip] << 28;
    tmp2 = ig_ts & 0x0FFFFFFF;
    sequence.seq = tmp1 & tmp2;
p2 = !(ipv4.src_ip in filter);
    drop();
```

Intermediate representation

One big pipeline model

No complex statements

No branches

Frontend


```
extern list<bit[32] ip>[300] filter;
if (ipv4.src_ip in filter) {
    add_header(sequence);
    sequence.seq = (filter[ipv4.src_ip] << 28) \
                   & (ig_ts & 0x0FFFFFFF);
}
else {
    drop();
}
```

Lyra program

```
p1 = ipv4.src_ip in filter;
    add_header(sequence);
    tmp1 = filter[ipv4.src_ip] << 28;
    tmp2 = ig_ts & 0x0FFFFFFF;
    sequence.seq = tmp1 & tmp2;
p2 = !(ipv4.src_ip in filter);
    drop();
```

Intermediate representation

One big pipeline model

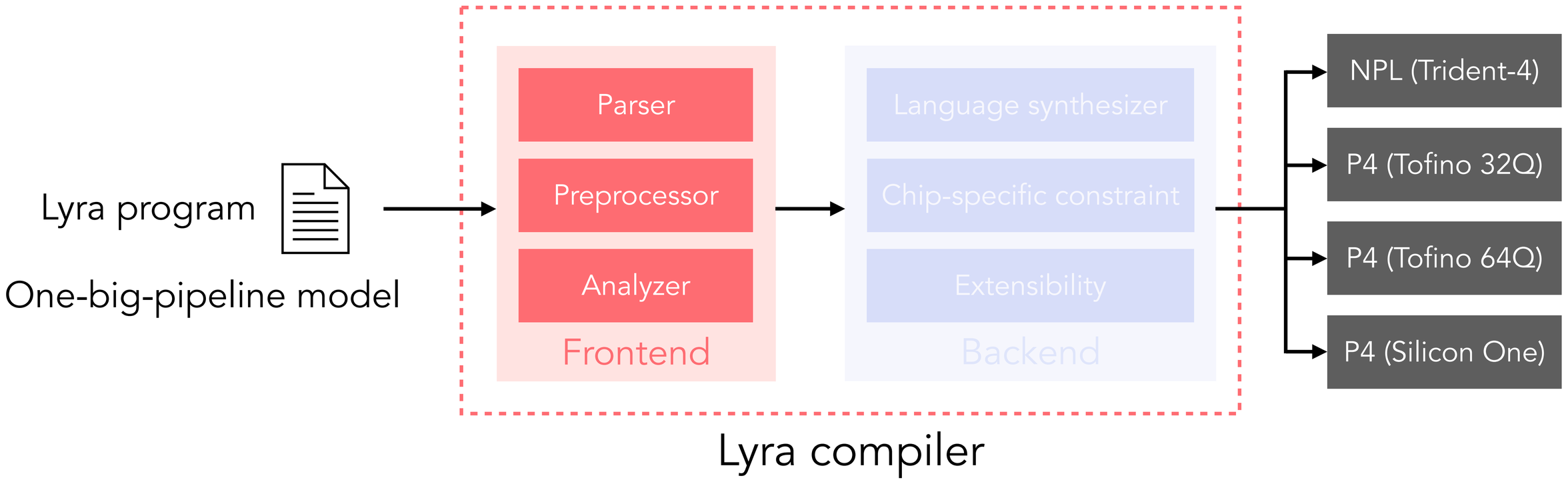
No complex statements

No branches

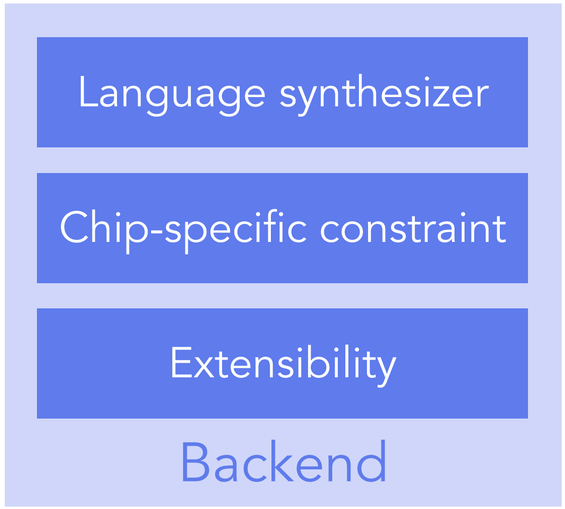
Has statement dependencies

Lyra: A high-level data plane language & compiler

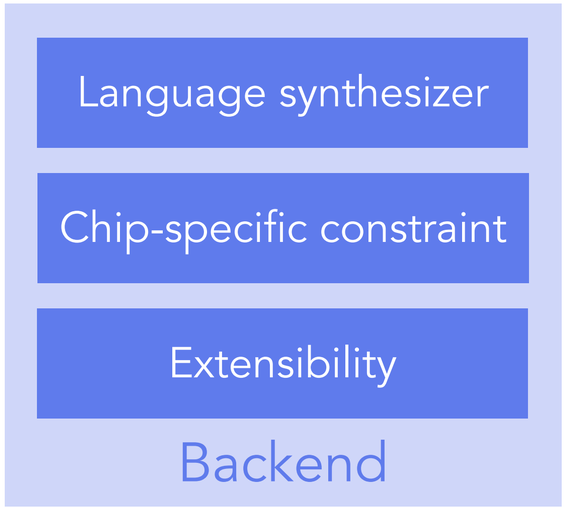
Alibaba Cloud
Worldwide Cloud Services Partner



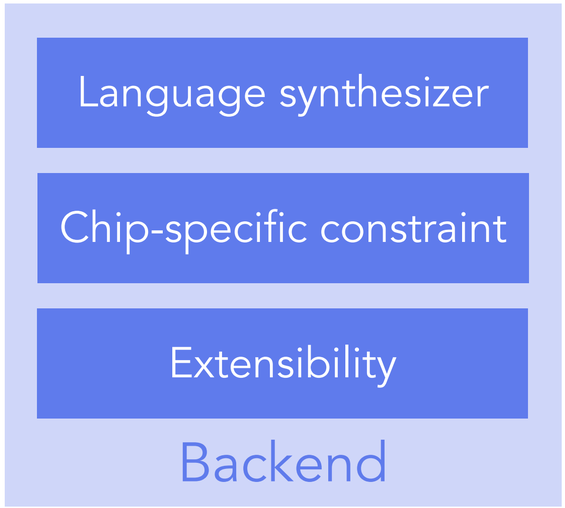
Lyra: A high-level data plane language & compiler



Lyra: A high-level data plane language & compiler

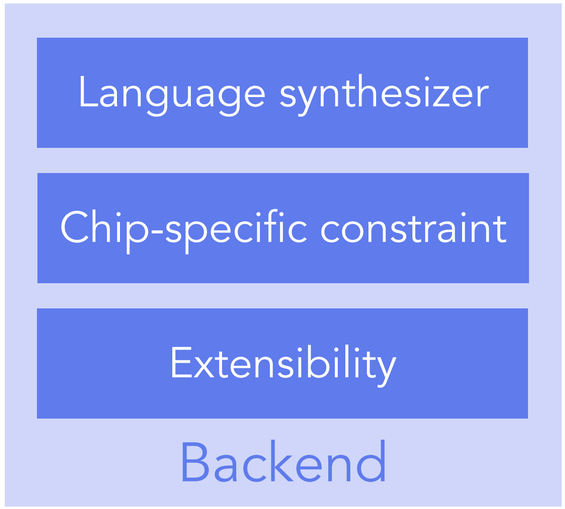


Lyra: A high-level data plane language & compiler



Portability I Compile Lyra into structured low level languages

Lyra: A high-level data plane language & compiler

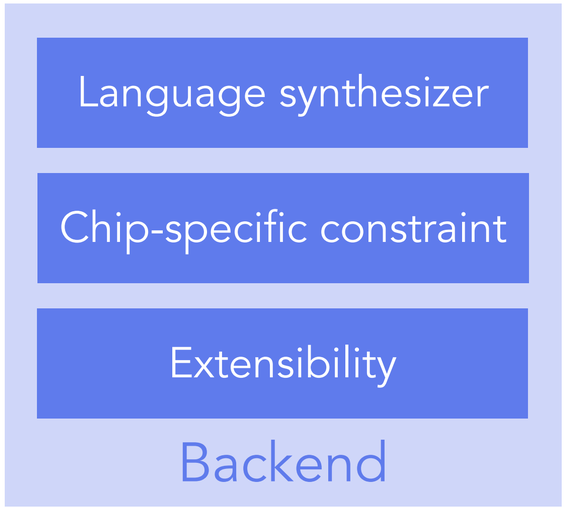


Portability | Compile Lyra into structured low level languages

Composition | Checks whether the program can fit into the switch

Lyra: A high-level data plane language & compiler

Alibaba Cloud
Worldwide Cloud Services Partner



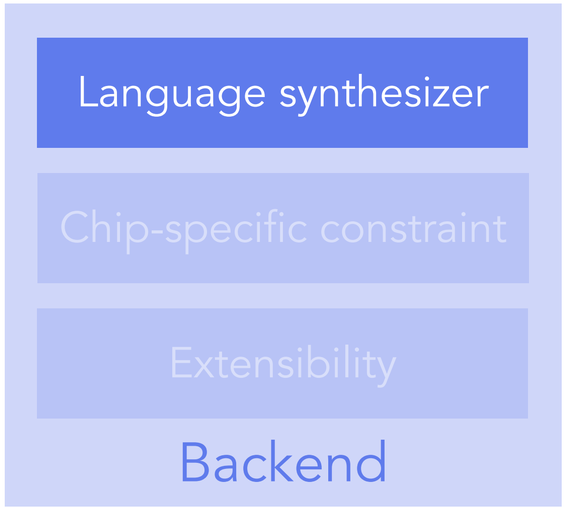
Portability | Compile Lyra into structured low level languages

Composition | Checks whether the program can fit into the switch

Extensibility | Correctly deploy the program in the scope

Lyra: A high-level data plane language & compiler

Alibaba Cloud
Worldwide Cloud Services Partner



Portability | Compile Lyra into structured low level languages

Composition | Checks whether the program can fit into the switch

Extensibility | Correctly deploy the program in the scope

Languages have different programming paradigms

Languages have different programming paradigms

Languages have different programming paradigms

Table oriented programming

Languages have different programming paradigms

Table oriented programming

Languages have different programming paradigms

Table oriented programming

Procedural oriented programming

Languages have different programming paradigms

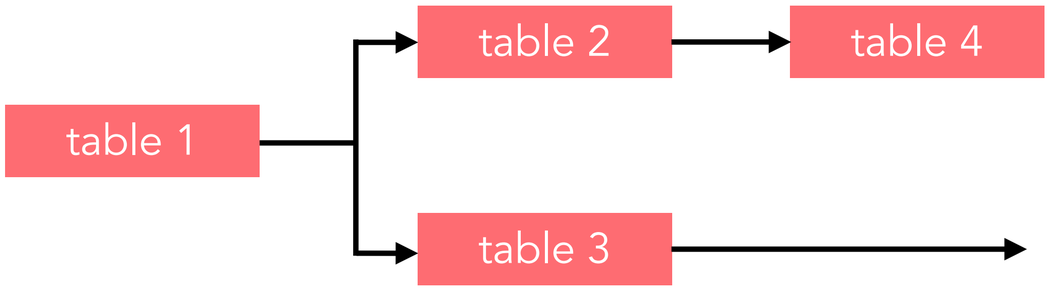
Table oriented programming

Procedural oriented programming

Languages have different programming paradigms

Table oriented programming

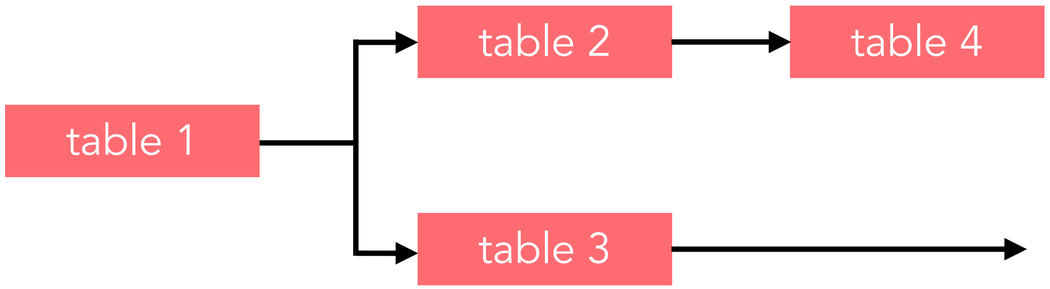
Procedural oriented programming



P4 Program

Languages have different programming paradigms

Table oriented programming



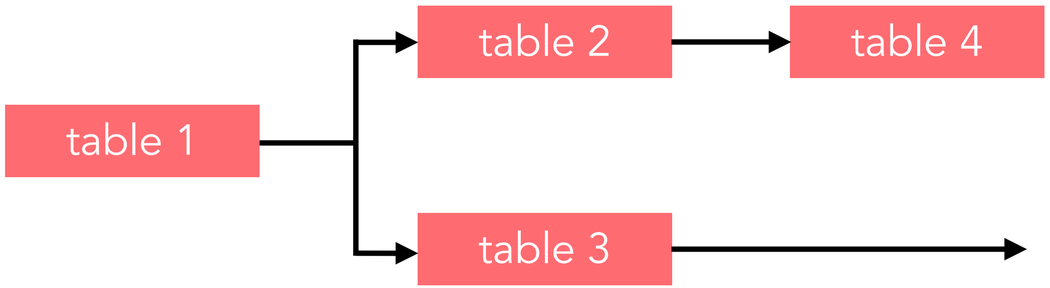
P4 Program

Procedural oriented programming

P4 Table

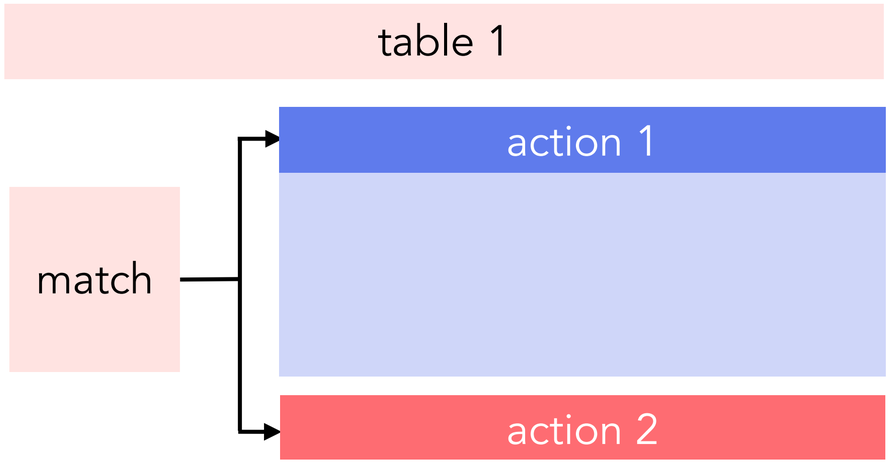
Languages have different programming paradigms

Table oriented programming



P4 Program

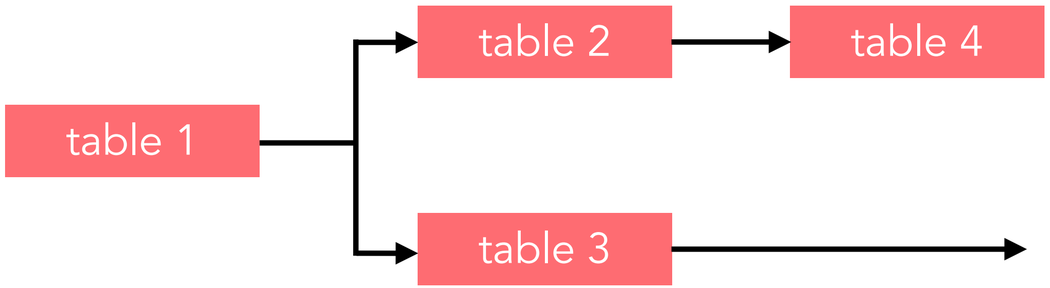
Procedural oriented programming



P4 Table

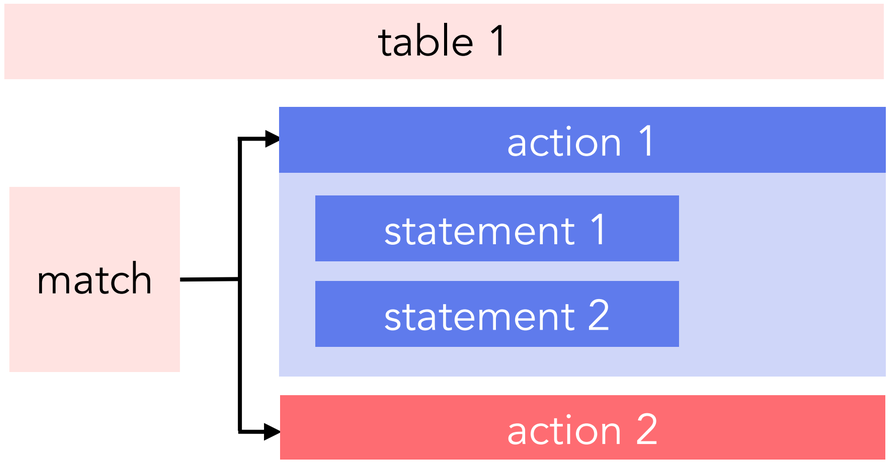
Languages have different programming paradigms

Table oriented programming



P4 Program

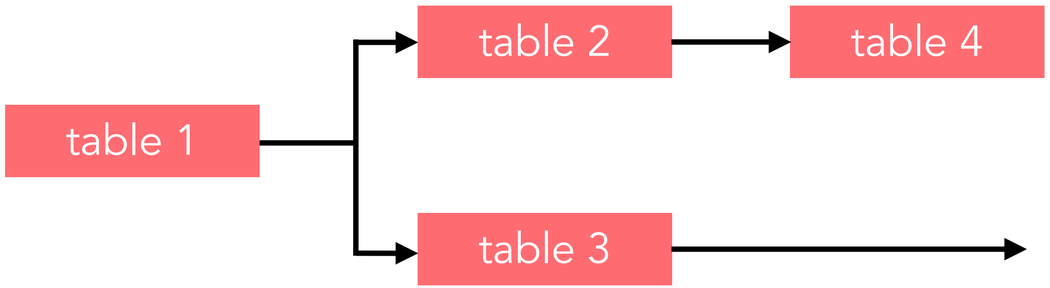
Procedural oriented programming



P4 Table

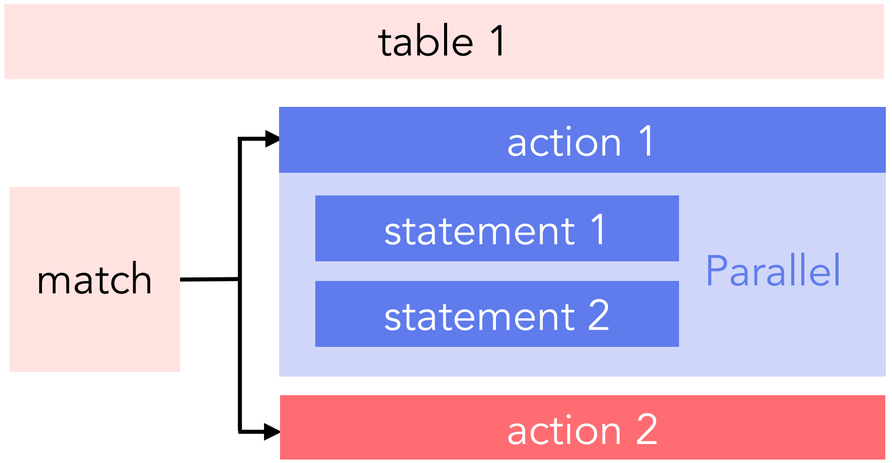
Languages have different programming paradigms

Table oriented programming



P4 Program

Procedural oriented programming



P4 Table

Predicate block

Predicate block

Alibaba Cloud
Worldwide Cloud Services Partner

Group of statements that has the same predicate and no dependency

Predicate block

Group of statements that has the same predicate and no dependency

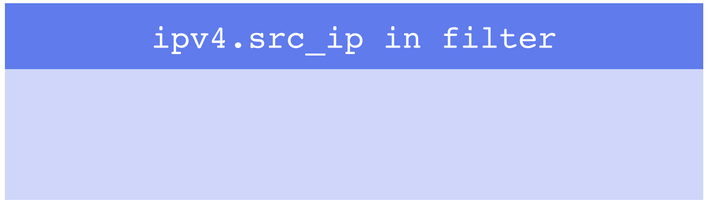
```
p1 = ipv4.src_ip in filter;
     add_header(sequence);
     tmp1 = filter[ipv4.src_ip] << 28;
     tmp2 = ig_ts & 0x0FFFFFFF;
     sequence.seq = tmp1 & tmp2;

p2 = !(ipv4.src_ip in filter);
     drop();
```

Predicate block

Alibaba Cloud
Worldwide Cloud Services Partner

Group of statements that has the same predicate and no dependency

```
p1 = ipv4.src_ip in filter;
     add_header(sequence);
     tmp1 = filter[ipv4.src_ip] << 28;
     tmp2 = ig_ts & 0x0FFFFFFF;
     sequence.seq = tmp1 & tmp2;

p2 = !(ipv4.src_ip in filter);
     drop();
```



Predicate block

Alibaba Cloud
Worldwide Cloud Services Partner

Group of statements that has the same predicate and no dependency

```
p1 = ipv4.src_ip in filter;
     add_header(sequence);
     tmp1 = filter[ipv4.src_ip] << 28;
     tmp2 = ig_ts & 0x0FFFFFFF;
     sequence.seq = tmp1 & tmp2;

p2 = !(ipv4.src_ip in filter);
     drop();
```

```
ipv4.src_ip in filter
add_header(sequence);
```

Predicate block

Alibaba Cloud
Worldwide Cloud Services Partner

Group of statements that has the same predicate and no dependency

```
p1 = ipv4.src_ip in filter;
     add_header(sequence);
     tmp1 = filter[ipv4.src_ip] << 28;
     tmp2 = ig_ts & 0x0FFFFFFF;
     sequence.seq = tmp1 & tmp2;

p2 = !(ipv4.src_ip in filter);
     drop();
```

```
ipv4.src_ip in filter
```

```
add_header(sequence);
tmp1 = filter[ipv4.src_ip] << 28;
```

Predicate block

Alibaba Cloud
Worldwide Cloud Services Partner

Group of statements that has the same predicate and no dependency

```
p1 = ipv4.src_ip in filter;
     add_header(sequence);
     tmp1 = filter[ipv4.src_ip] << 28;
     tmp2 = ig_ts & 0x0FFFFFFF;
     sequence.seq = tmp1 & tmp2;

p2 = !(ipv4.src_ip in filter);
     drop();
```

```
ipv4.src_ip in filter
```

```
add_header(sequence);
tmp1 = filter[ipv4.src_ip] << 28;
tmp2 = ig_ts & 0x0FFFFFFF;
```

Predicate block

Alibaba Cloud
Worldwide Cloud Services Partner

Group of statements that has the same predicate and no dependency

```
p1 = ipv4.src_ip in filter;
     add_header(sequence);
     tmp1 = filter[ipv4.src_ip] << 28;
     tmp2 = ig_ts & 0x0FFFFFFF;
     sequence.seq = tmp1 & tmp2;

p2 = !(ipv4.src_ip in filter);
     drop();
```

```
ipv4.src_ip in filter
add_header(sequence);
tmp1 = filter[ipv4.src_ip] << 28;
tmp2 = ig_ts & 0x0FFFFFFF;
```

```
ipv4.src_ip in filter
```

Predicate block

Group of statements that has the same predicate and no dependency

```
p1 = ipv4.src_ip in filter;
     add_header(sequence);
     tmp1 = filter[ipv4.src_ip] << 28;
     tmp2 = ig_ts & 0x0FFFFFFF;
     sequence.seq = tmp1 & tmp2;

p2 = !(ipv4.src_ip in filter);
     drop();
```

```
ipv4.src_ip in filter
add_header(sequence);
tmp1 = filter[ipv4.src_ip] << 28;
tmp2 = ig_ts & 0x0FFFFFFF;
```

```
ipv4.src_ip in filter
sequence.seq = tmp1 & tmp2;
```

Predicate block

Group of statements that has the same predicate and no dependency

```
p1 = ipv4.src_ip in filter;
     add_header(sequence);
     tmp1 = filter[ipv4.src_ip] << 28;
     tmp2 = ig_ts & 0x0FFFFFFF;
     sequence.seq = tmp1 & tmp2;

p2 = !(ipv4.src_ip in filter);
     drop();
```

```
ipv4.src_ip in filter
add_header(sequence);
tmp1 = filter[ipv4.src_ip] << 28;
tmp2 = ig_ts & 0x0FFFFFFF;
```

```
!(ipv4.src_ip in filter)
```

```
ipv4.src_ip in filter
sequence.seq = tmp1 & tmp2;
```

Predicate block

Alibaba Cloud
Worldwide Cloud Services Partner

Group of statements that has the same predicate and no dependency

```
p1 = ipv4.src_ip in filter;
     add_header(sequence);
     tmp1 = filter[ipv4.src_ip] << 28;
     tmp2 = ig_ts & 0x0FFFFFFF;
     sequence.seq = tmp1 & tmp2;

p2 = !(ipv4.src_ip in filter);
     drop();
```

```
ipv4.src_ip in filter
add_header(sequence);
tmp1 = filter[ipv4.src_ip] << 28;
tmp2 = ig_ts & 0x0FFFFFFF;
```

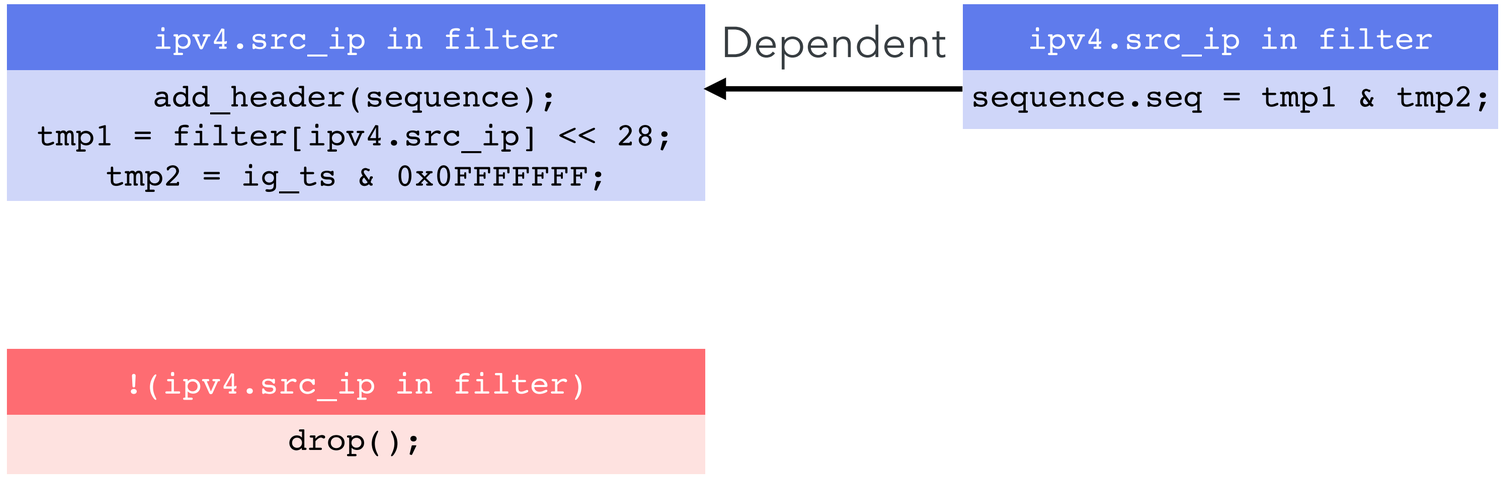
```
!(ipv4.src_ip in filter)
drop();
```

```
ipv4.src_ip in filter
sequence.seq = tmp1 & tmp2;
```

Predicate block

Group of statements that has the same predicate and no dependency

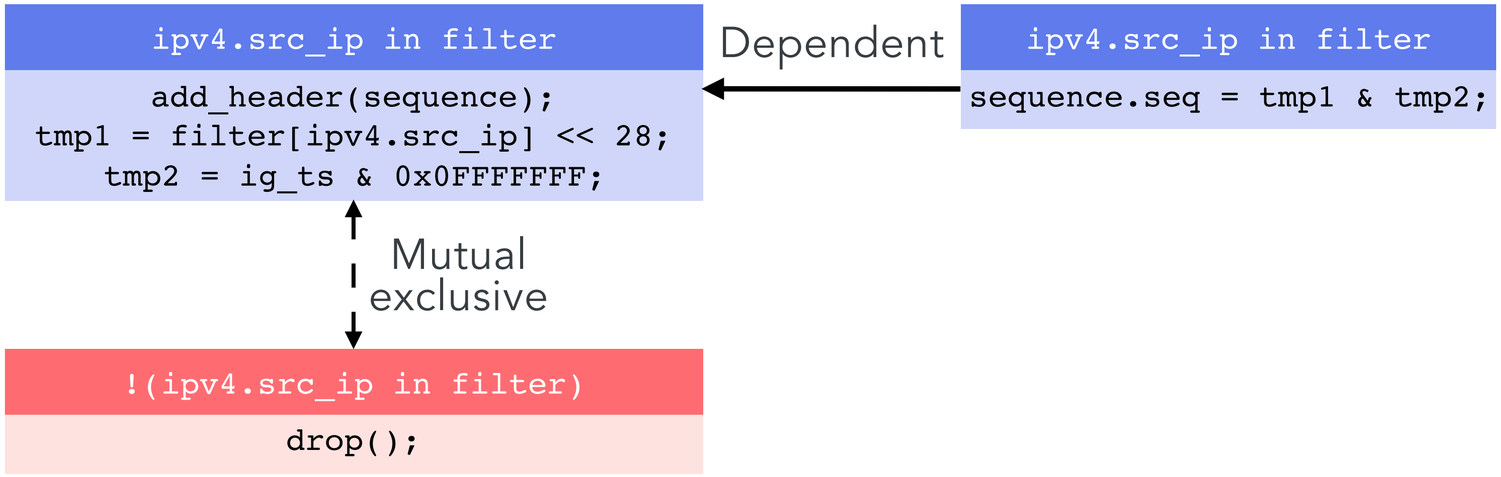
```
p1 = ipv4.src_ip in filter;
     add_header(sequence);
     tmp1 = filter[ipv4.src_ip] << 28;
     tmp2 = ig_ts & 0x0FFFFFFF;
     sequence.seq = tmp1 & tmp2;

p2 = !(ipv4.src_ip in filter);
     drop();
```



Predicate block

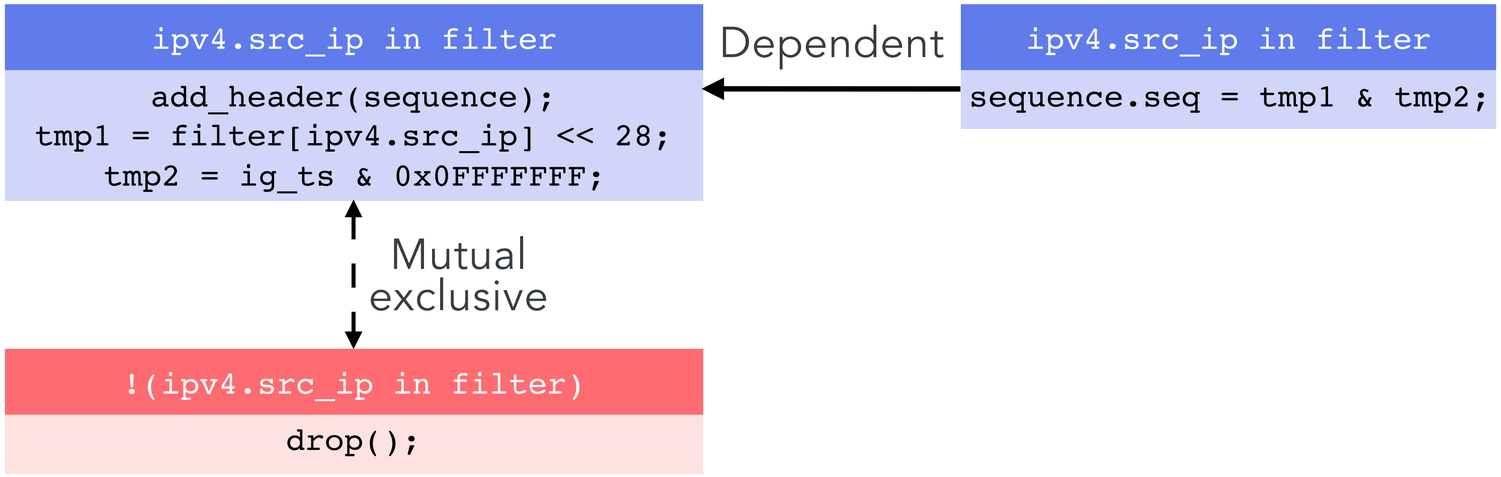
Group of statements that has the same predicate and no dependency

```
p1 = ipv4.src_ip in filter;
     add_header(sequence);
     tmp1 = filter[ipv4.src_ip] << 28;
     tmp2 = ig_ts & 0x0FFFFFFF;
     sequence.seq = tmp1 & tmp2;

p2 = !(ipv4.src_ip in filter);
     drop();
```



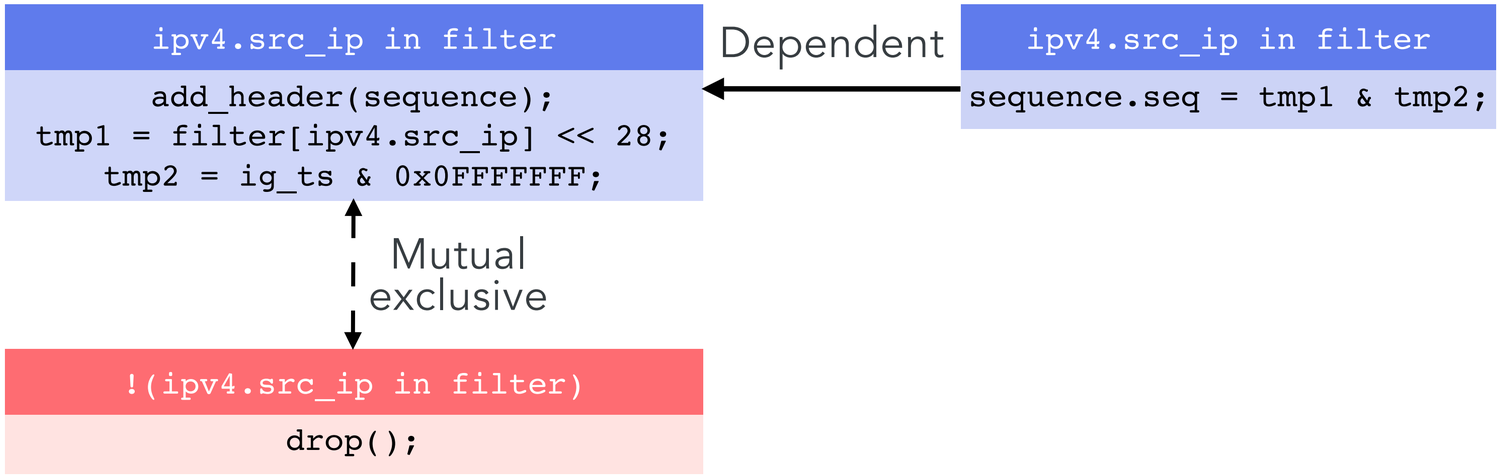
Predicate block

Group of statements that has the same predicate and no dependency



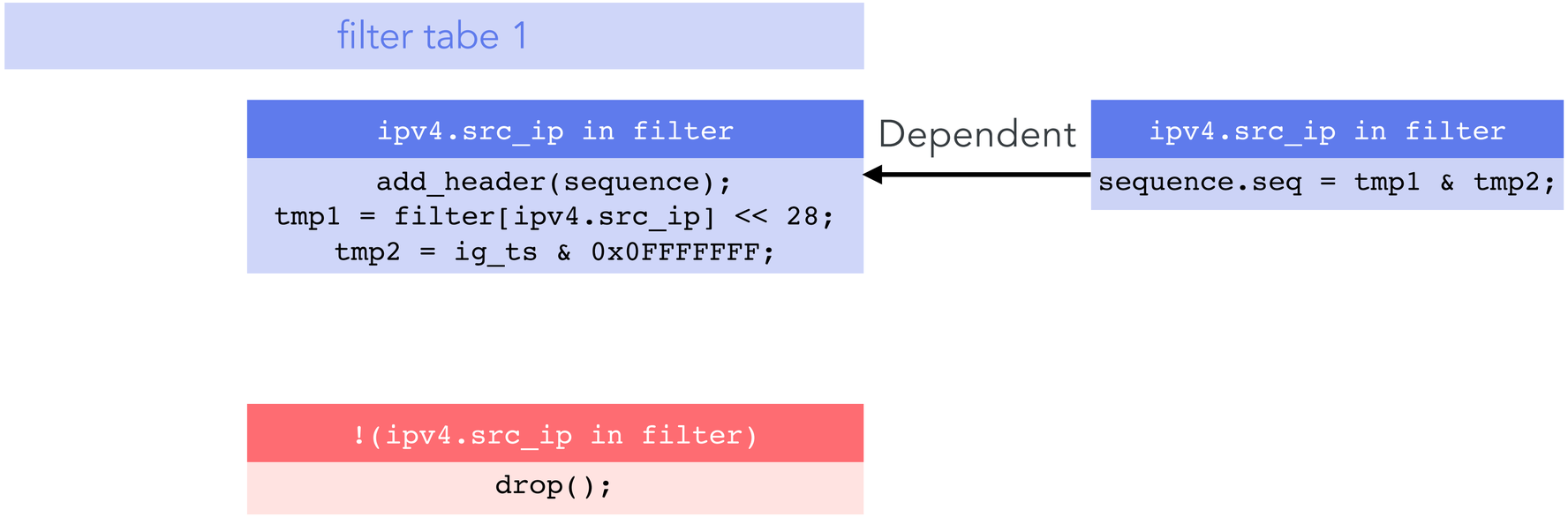
Predicate block

Group of statements that has the same predicate and no dependency



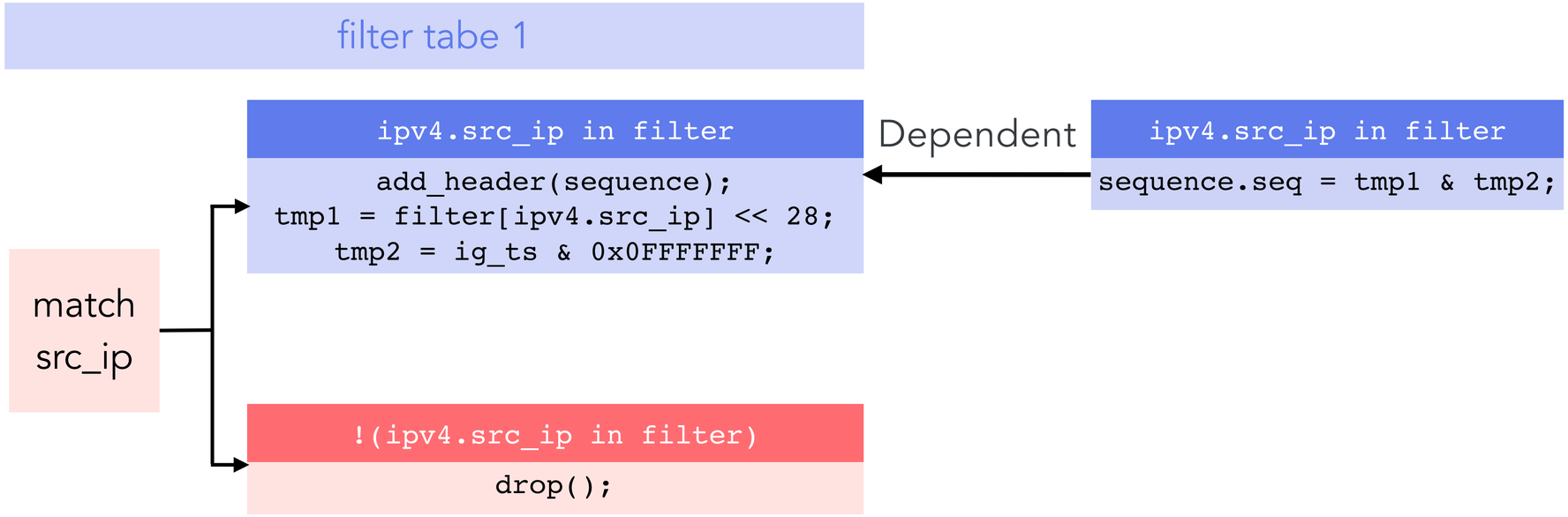
Predicate block

Group of statements that has the same predicate and no dependency



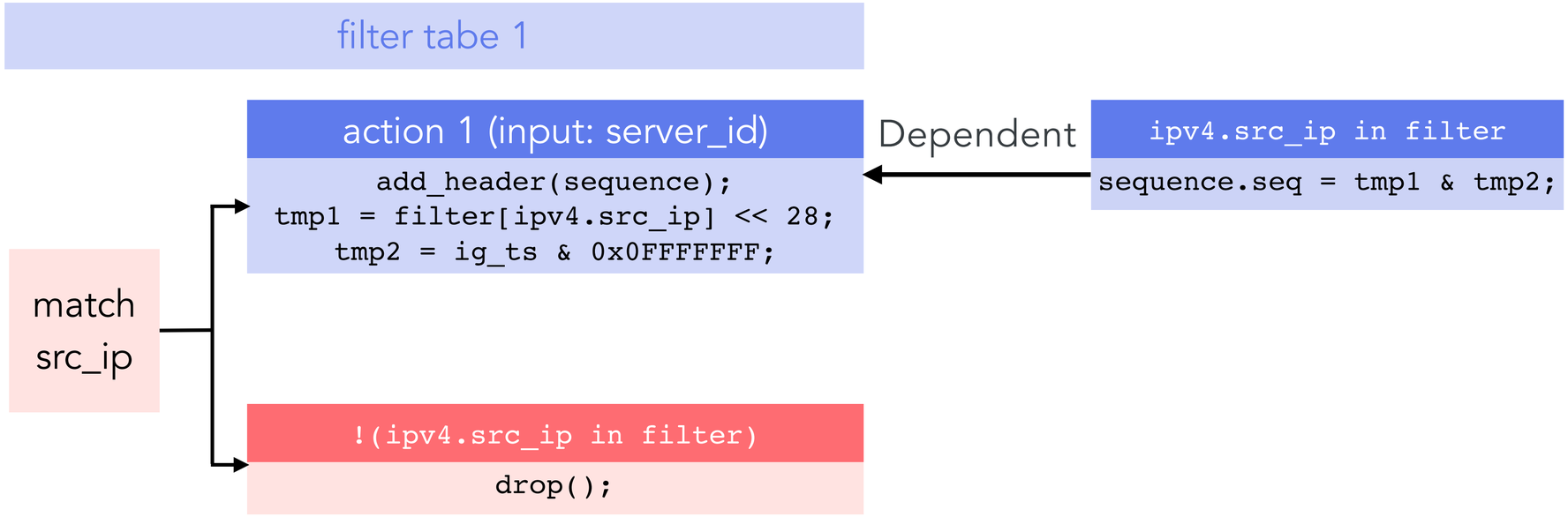
Predicate block

Group of statements that has the same predicate and no dependency



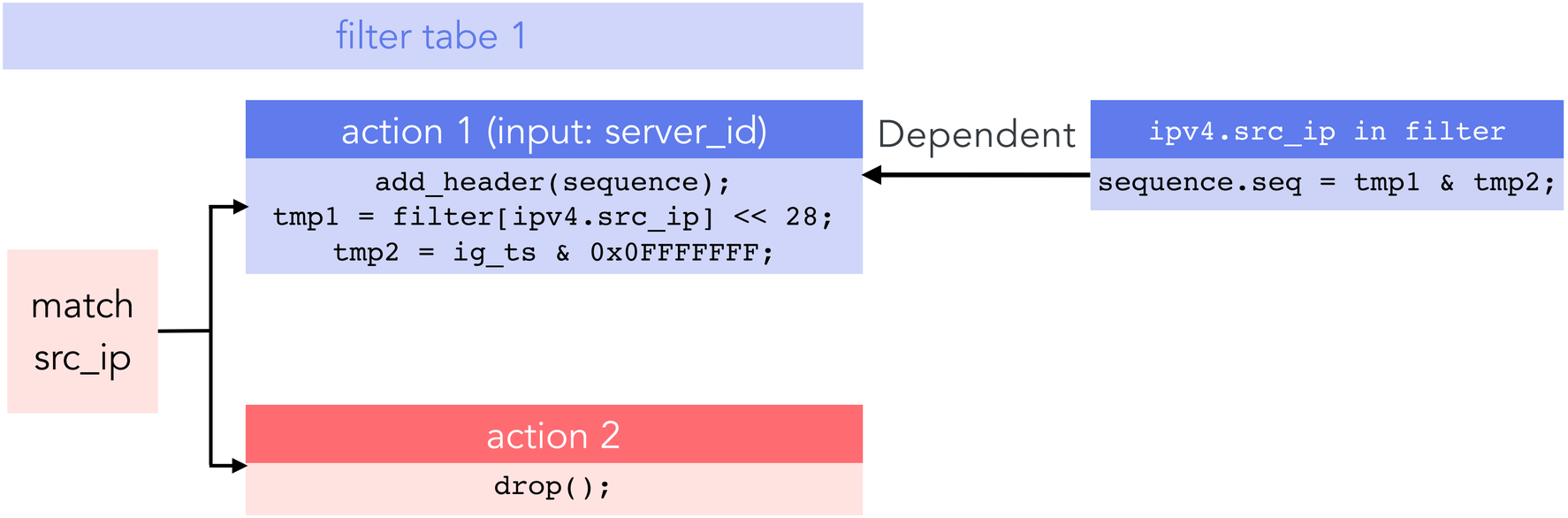
Predicate block

Group of statements that has the same predicate and no dependency



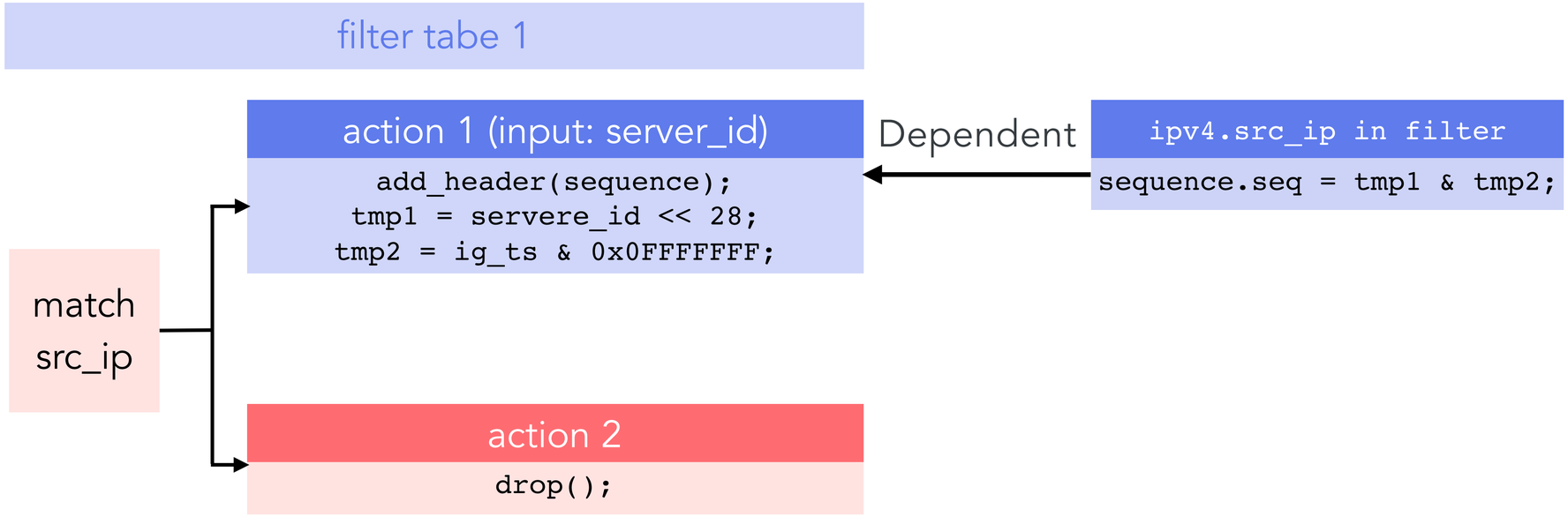
Predicate block

Group of statements that has the same predicate and no dependency



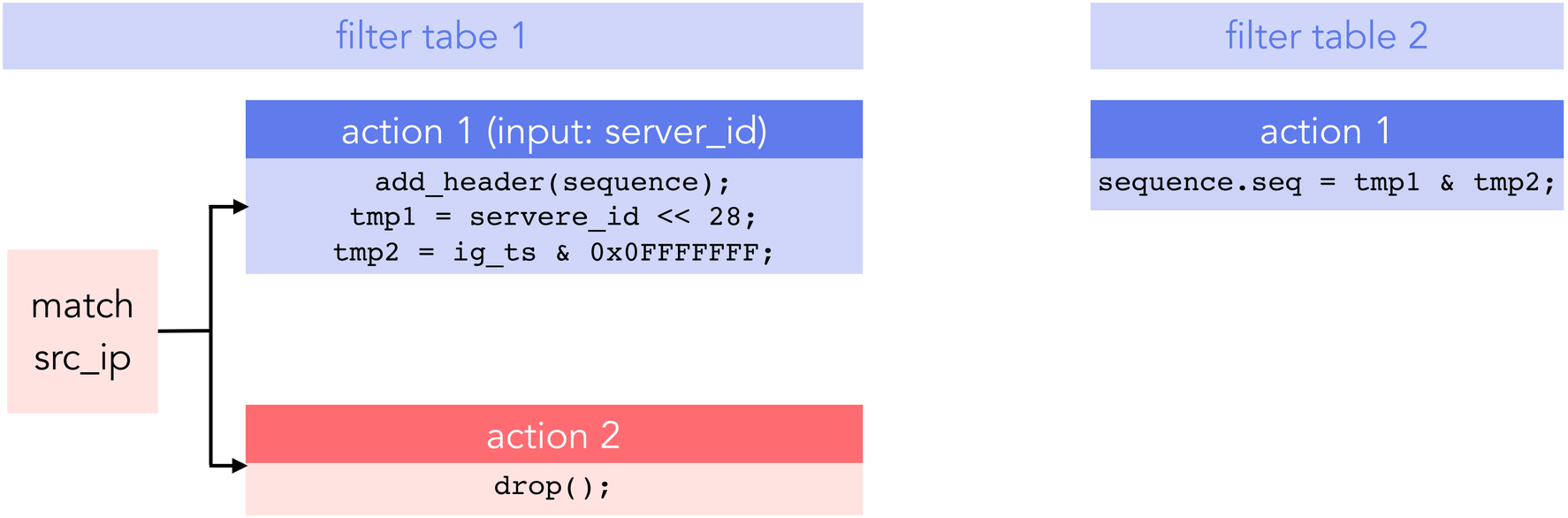
Predicate block

Group of statements that has the same predicate and no dependency



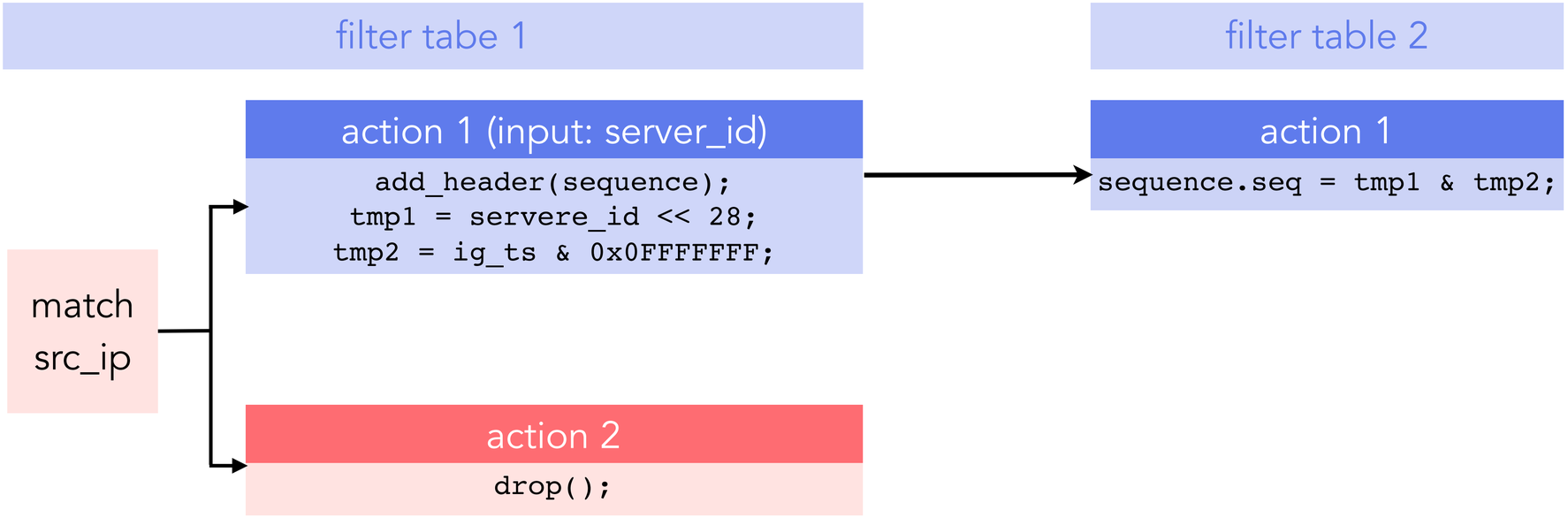
Predicate block

Group of statements that has the same predicate and no dependency



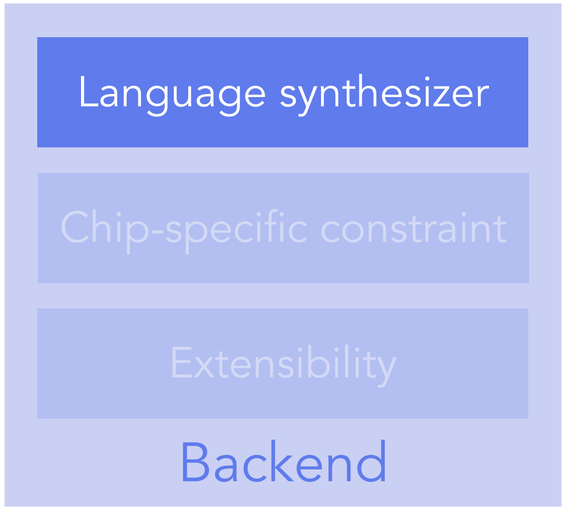
Predicate block

Group of statements that has the same predicate and no dependency



Lyra: A high-level data plane language & compiler

Alibaba Cloud
Worldwide Cloud Services Partner



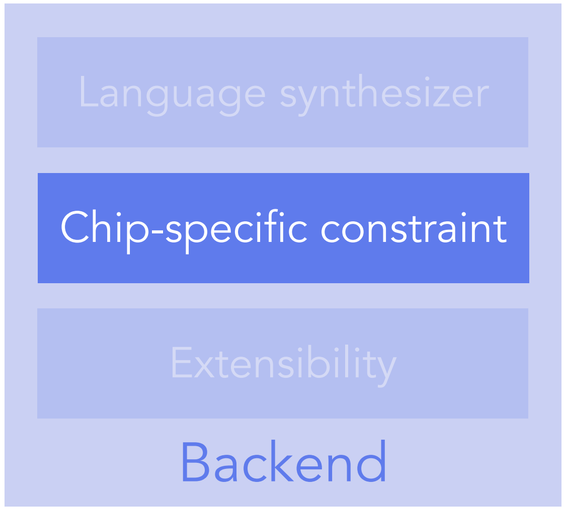
→ **Portability |** Compile Lyra into structured low level languages

→ **Composition |** Checks whether the program can fit into the switch

→ **Extensibility |** Correctly deploy the program in the scope

Lyra: A high-level data plane language & compiler

Alibaba Cloud
Worldwide Cloud Services Partner

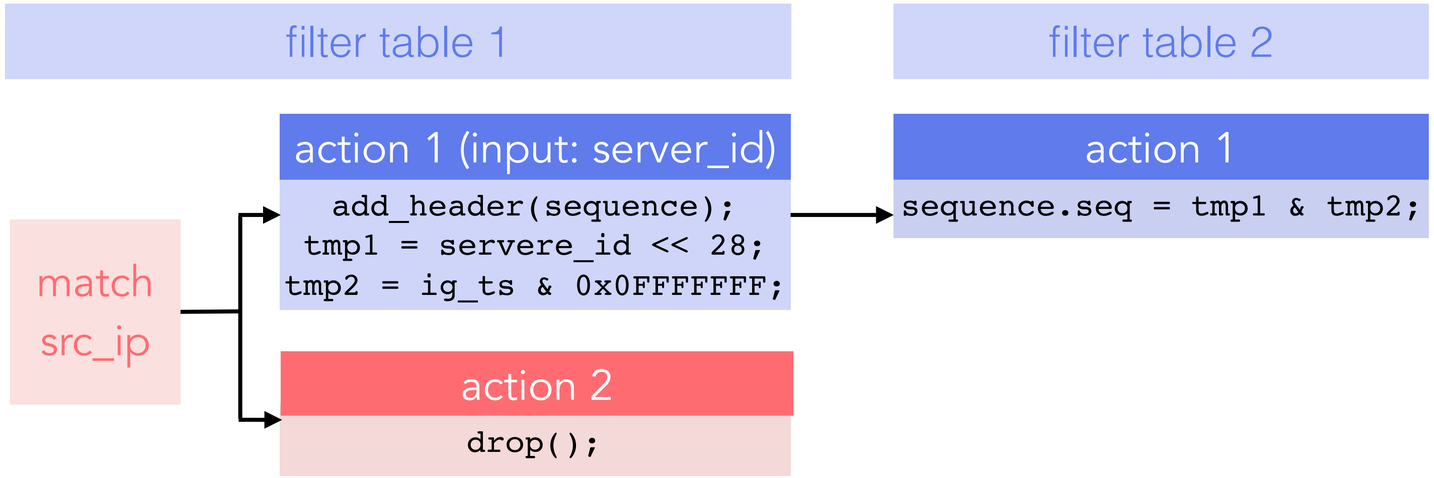


Portability | Compile Lyra into structured low level languages

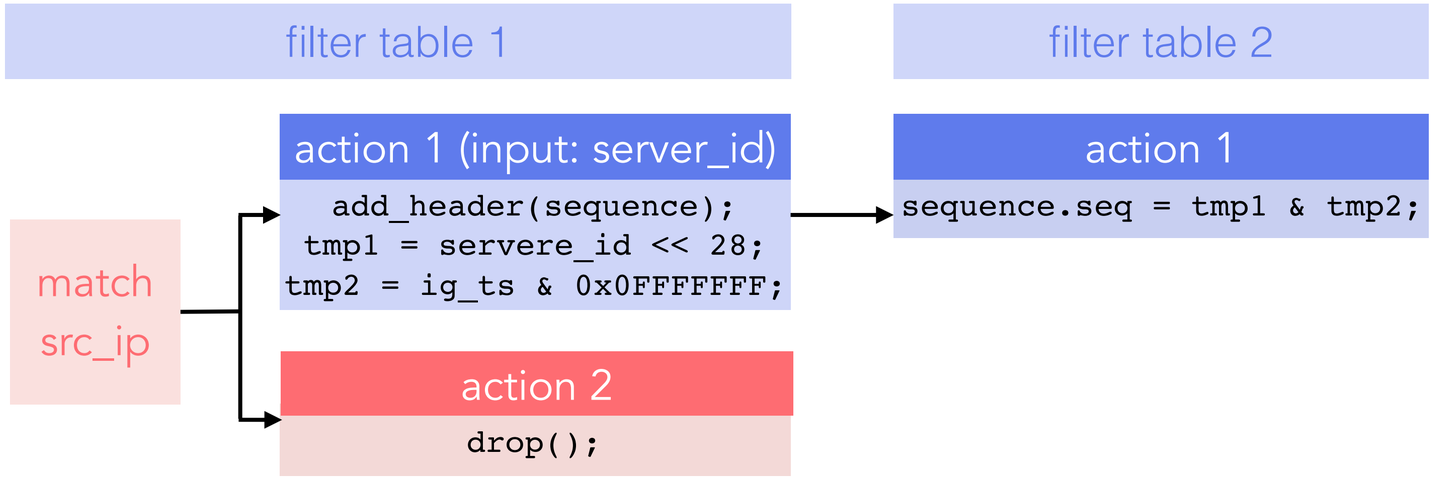
Composition | Checks whether the program can fit into the switch

Extensibility | Correctly deploy the program in the scope

Target-specific resource encoding: RMT

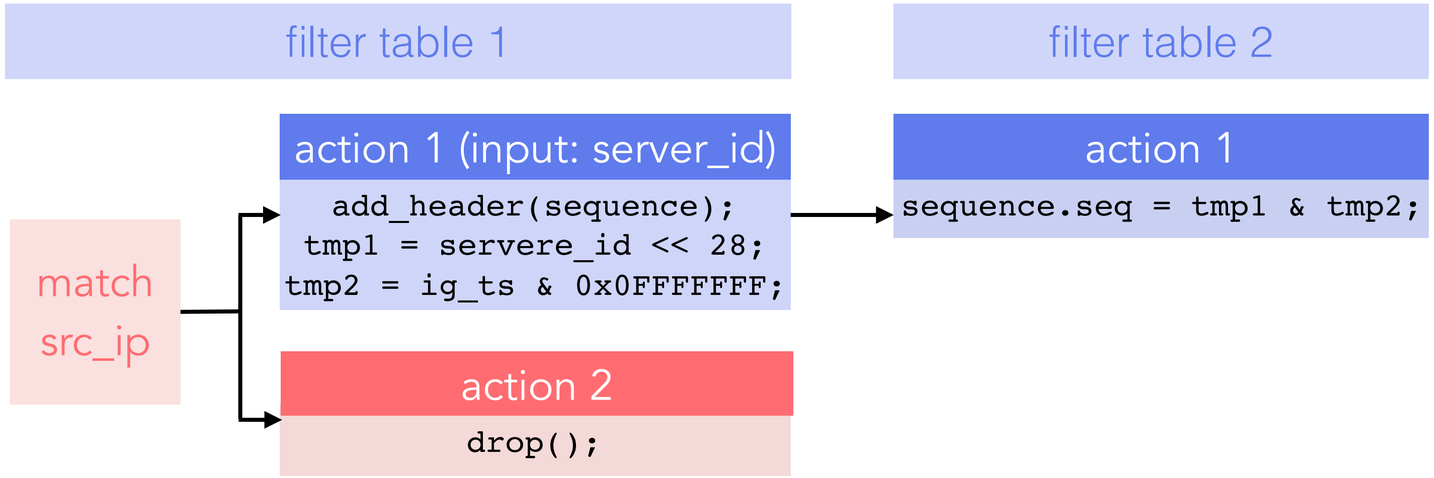


Target-specific resource encoding: RMT



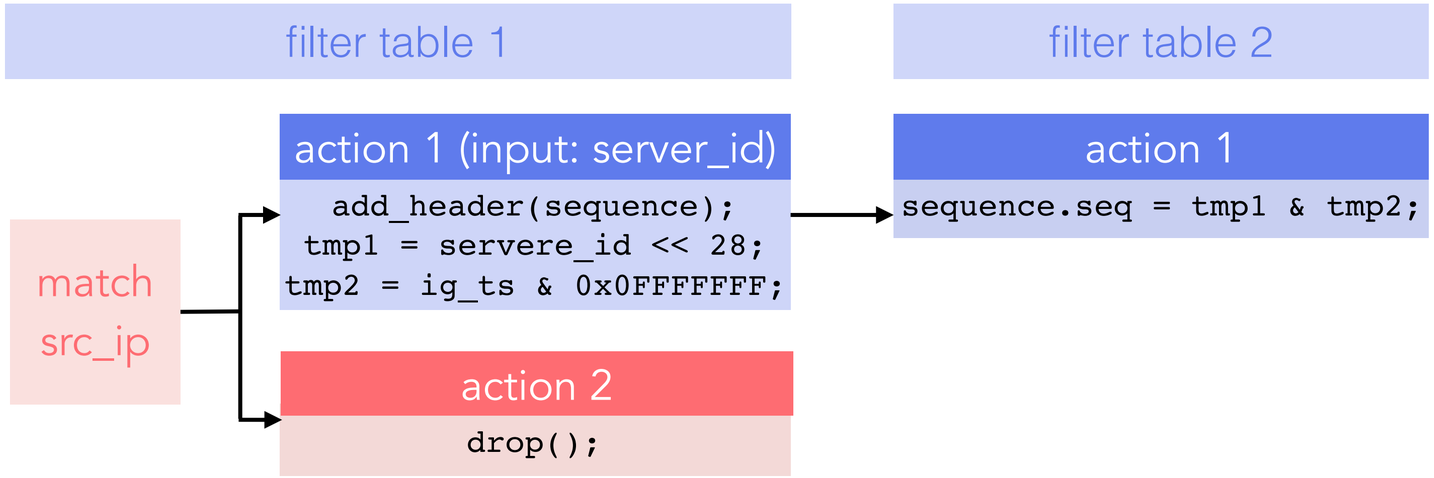
S_{filter_1} S_{filter_2}

Target-specific resource encoding: RMT



$$S_{filter_1} < S_{filter_2}$$

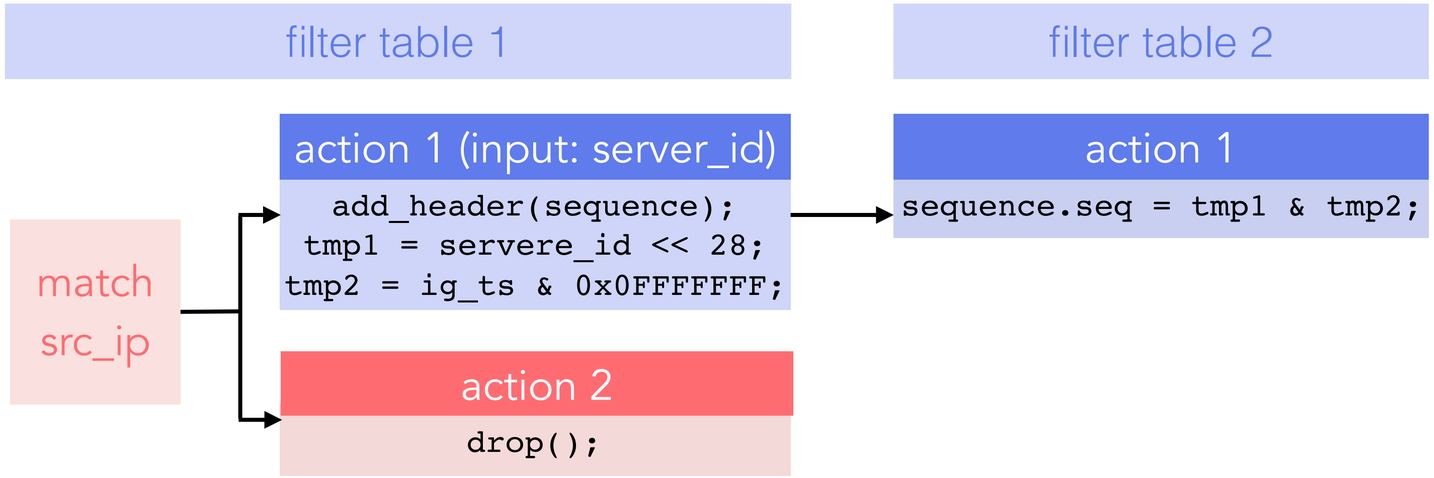
Target-specific resource encoding: RMT



$$S_{filter_1} < S_{filter_2}$$

$$0 \leq S_{filter_1} \leq 31$$

Target-specific resource encoding: RMT

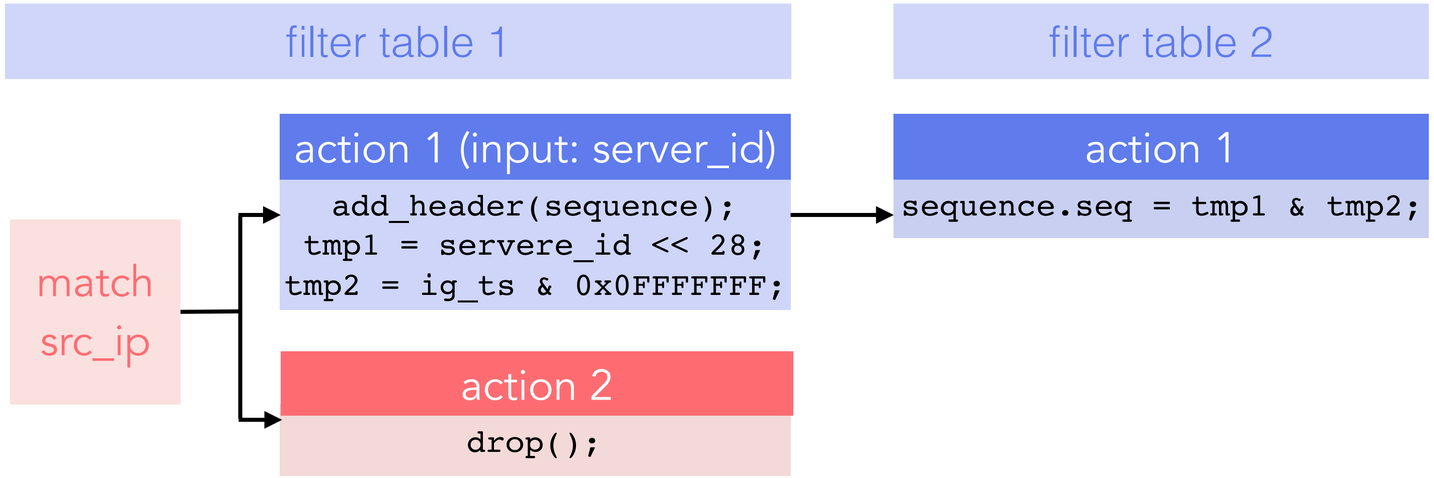


$$S_{filter_1} < S_{filter_2}$$

$$0 \leq S_{filter_1} \leq 31$$

$$0 \leq S_{filter_2} \leq 31$$

Target-specific resource encoding: RMT

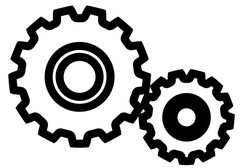


$$S_{filter_1} < S_{filter_2}$$

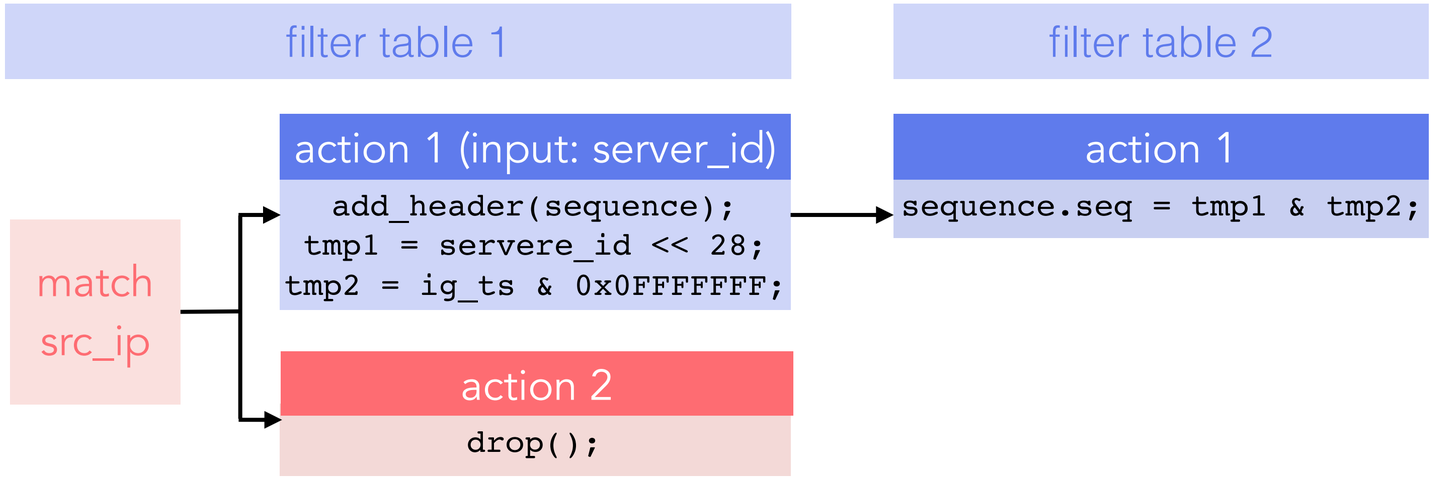
$$0 \leq S_{filter_1} \leq 31$$

$$0 \leq S_{filter_2} \leq 31$$

SMT solver



Target-specific resource encoding: RMT

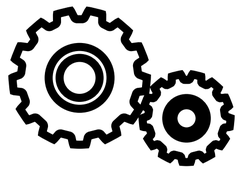


$$S_{filter_1} < S_{filter_2}$$

$$0 \leq S_{filter_1} \leq 31$$

$$0 \leq S_{filter_2} \leq 31$$

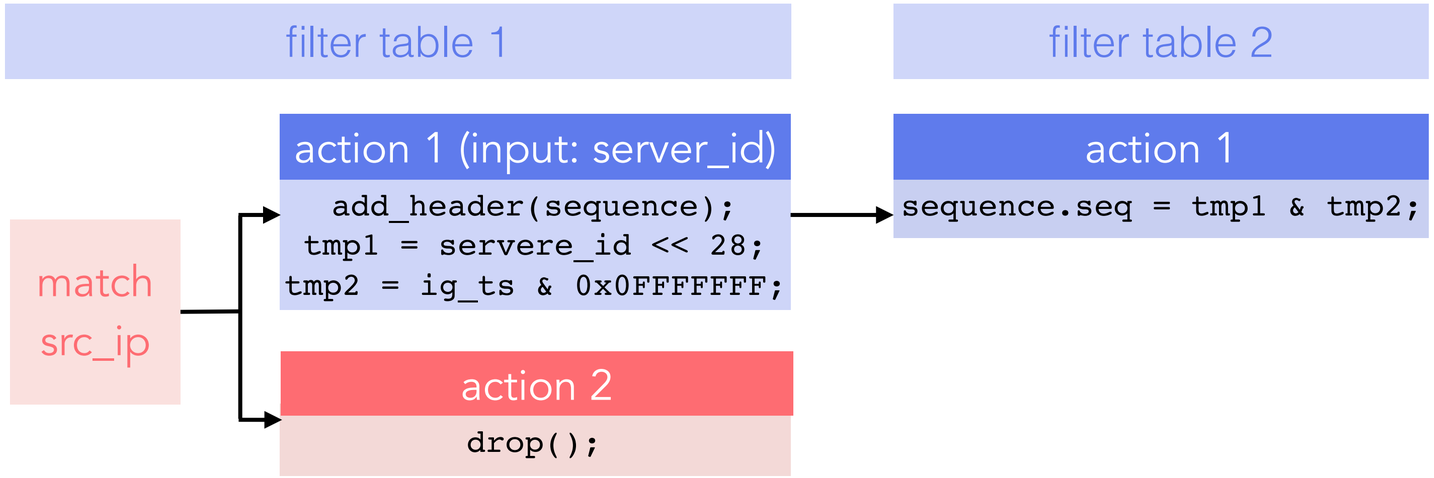
SMT solver



$$S_{filter_1} = 0$$

$$S_{filter_2} = 1$$

Target-specific resource encoding: RMT



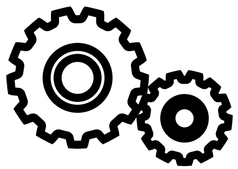
- SRAM memory
- TCAM memory
- Packet Header Vector
- Table num per stage

$$S_{filter_1} < S_{filter_2}$$

$$0 \leq S_{filter_1} \leq 31$$

$$0 \leq S_{filter_2} \leq 31$$

SMT solver

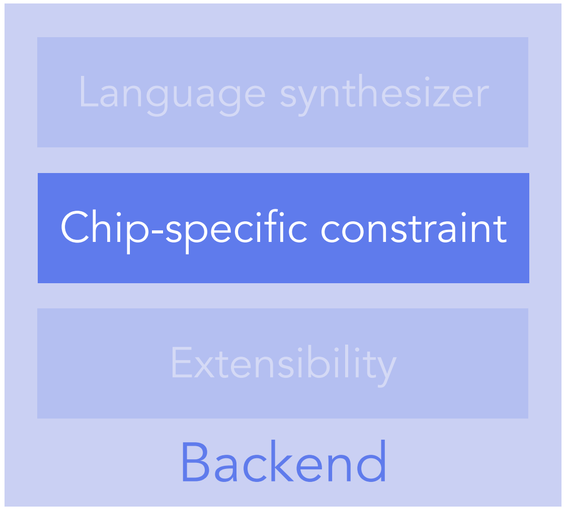


$$S_{filter_1} = 0$$

$$S_{filter_2} = 1$$

Lyra: A high-level data plane language & compiler

Alibaba Cloud
Worldwide Cloud Services Partner



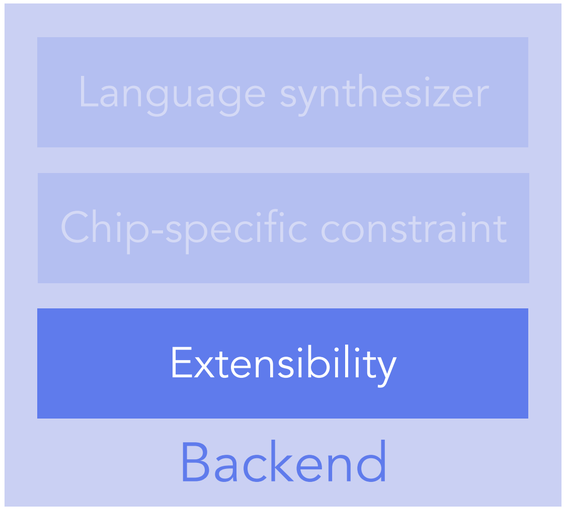
→ **Portability |** Compile Lyra into structured low level languages

→ **Composition |** Checks whether the program can fit into the switch

→ **Extensibility |** Correctly deploy the program in the scope

Lyra: A high-level data plane language & compiler

Alibaba Cloud
Worldwide Cloud Services Partner



→ **Portability I** Compile Lyra into structured low level languages

→ **Composition I** Checks whether the program can fit into the switch

→ **Extensibility I** Correctly deploy the program in the scope

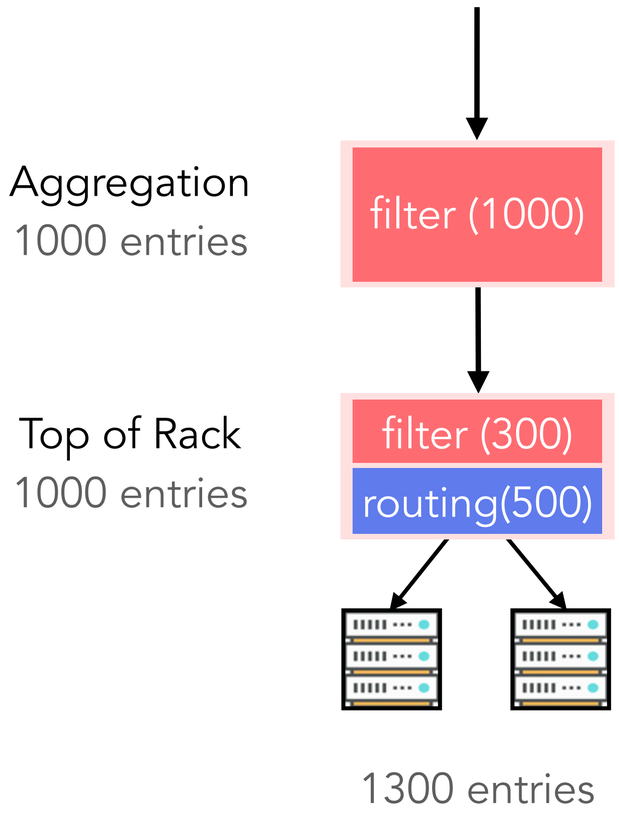
Pass information between switches

Pass information between switches

Pack downstream switches' required data into packet header

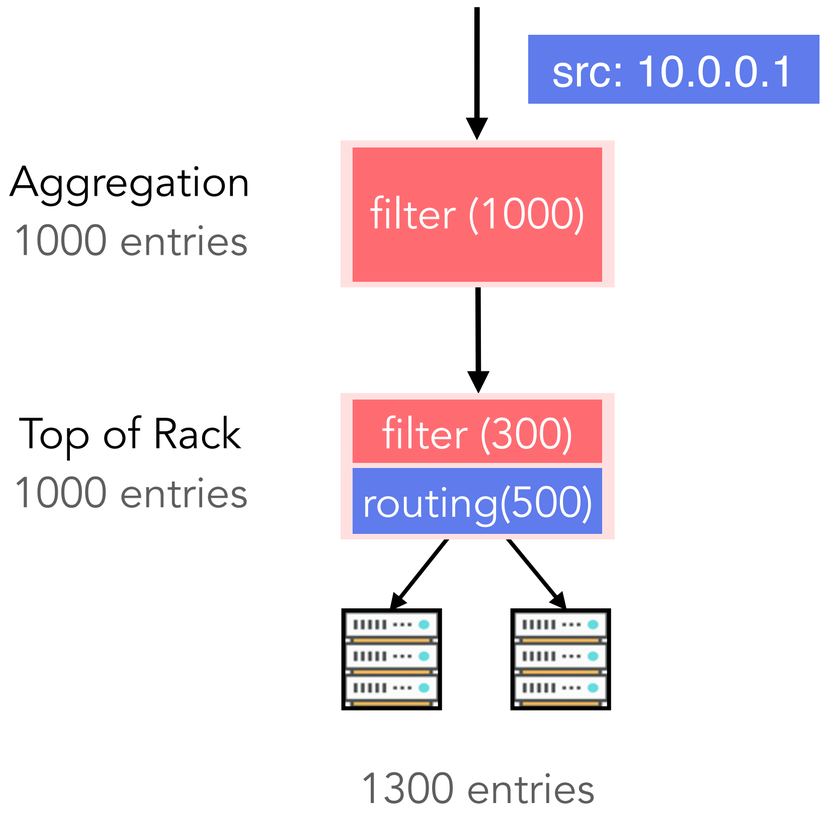
Pass information between switches

Pack downstream switches' required data into packet header



Pass information between switches

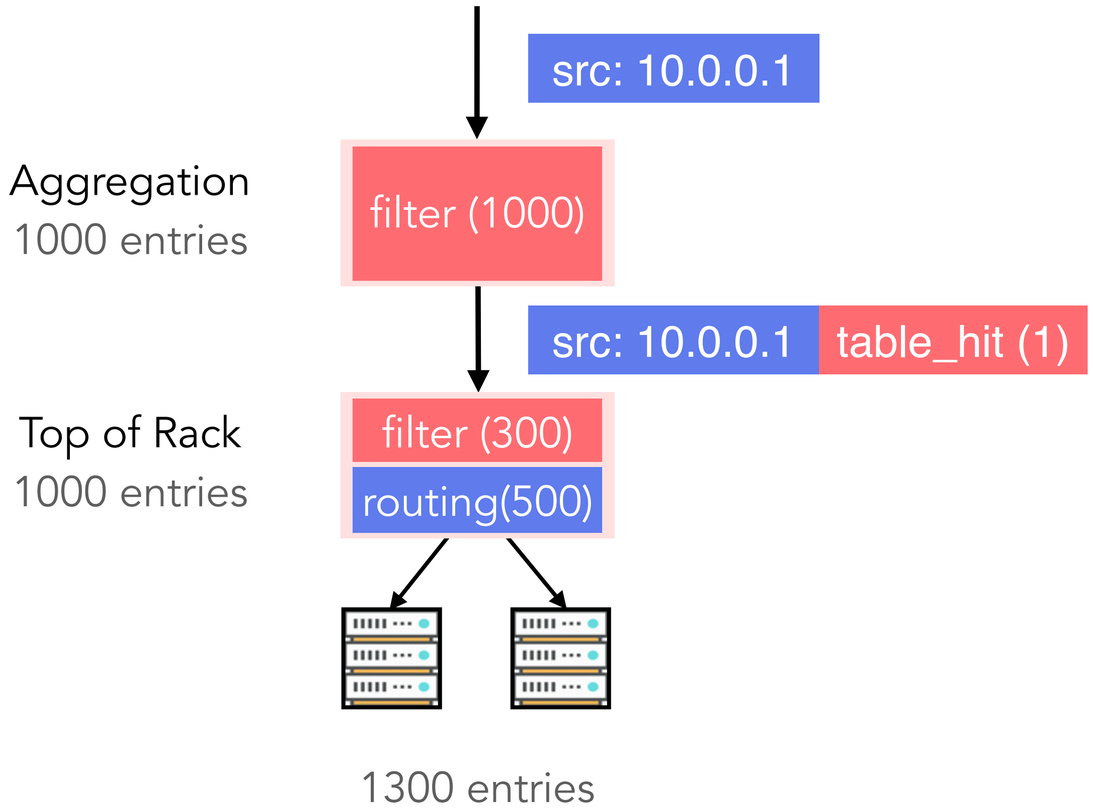
Pack downstream switches' required data into packet header



Pass information between switches

Alibaba Cloud
Worldwide Cloud Services Partner

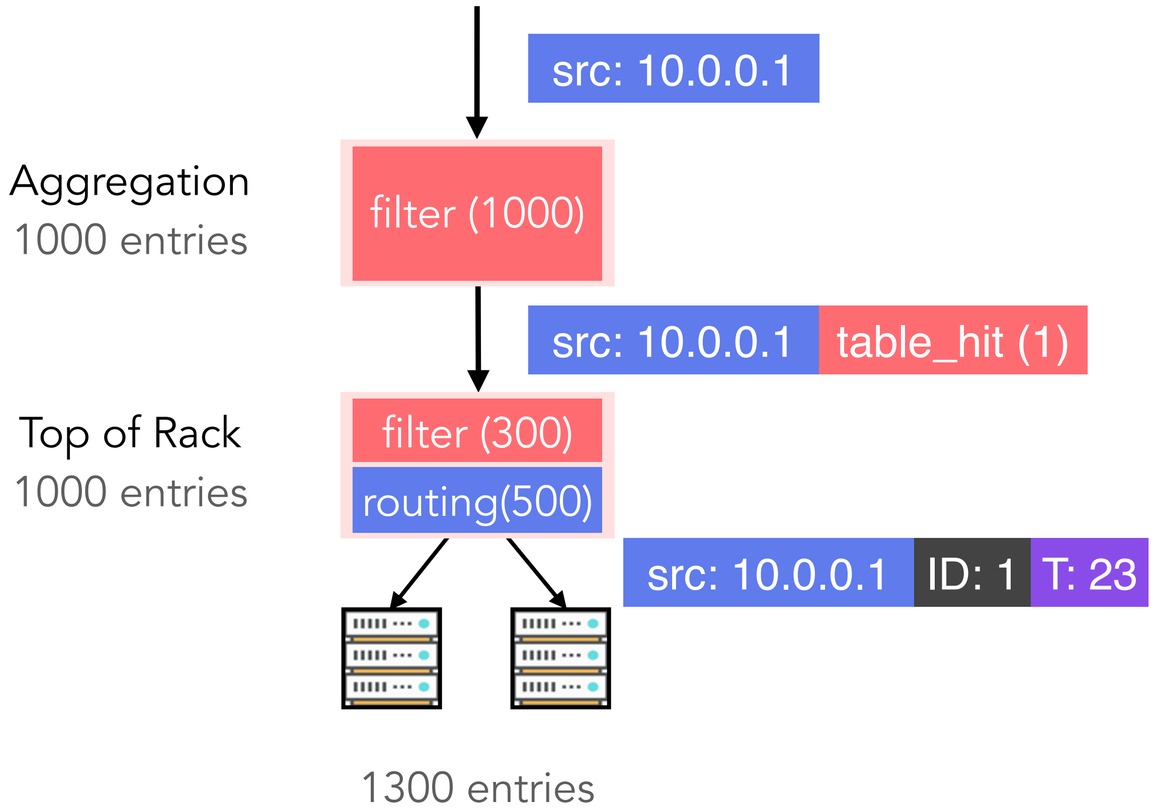
Pack downstream switches' required data into packet header



Pass information between switches

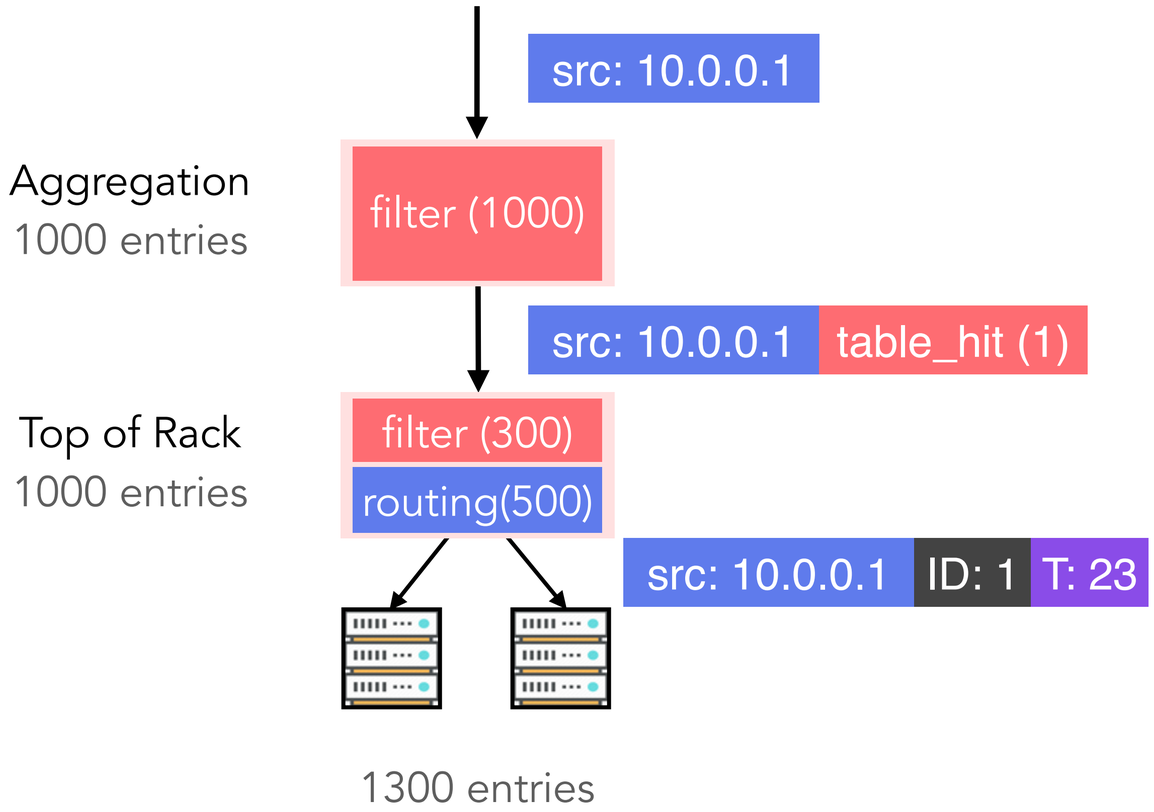
Alibaba Cloud
Worldwide Cloud Services Partner

Pack downstream switches' required data into packet header



Pass information between switches

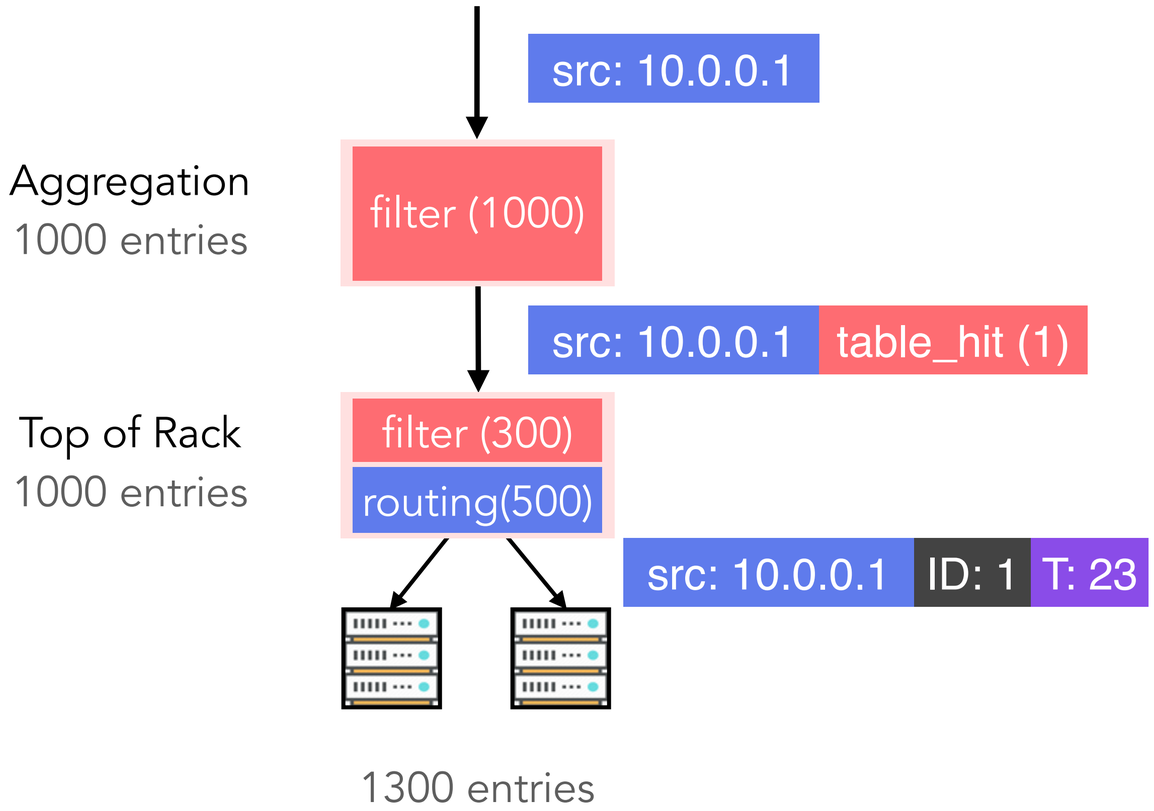
Pack downstream switches' required data into packet header



No code is duplicated

Pass information between switches

Pack downstream switches' required data into packet header

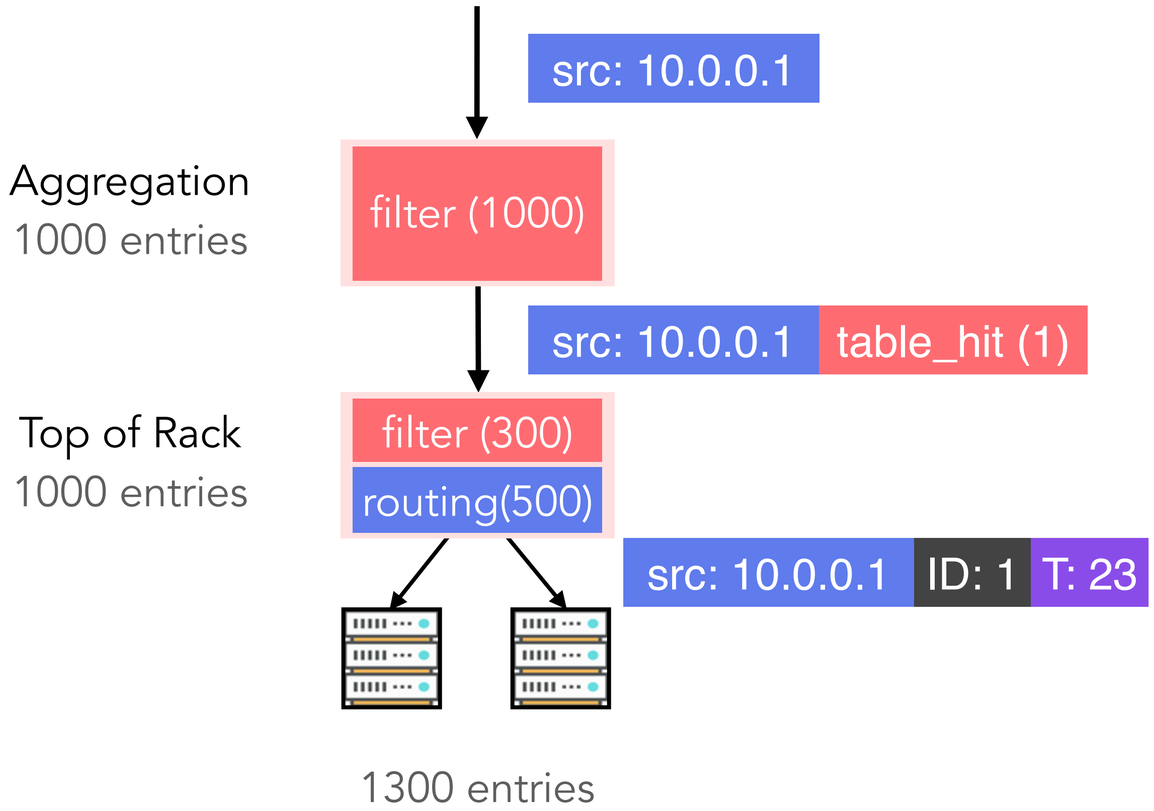


No code is duplicated

No code is missing

Pass information between switches

Pack downstream switches' required data into packet header



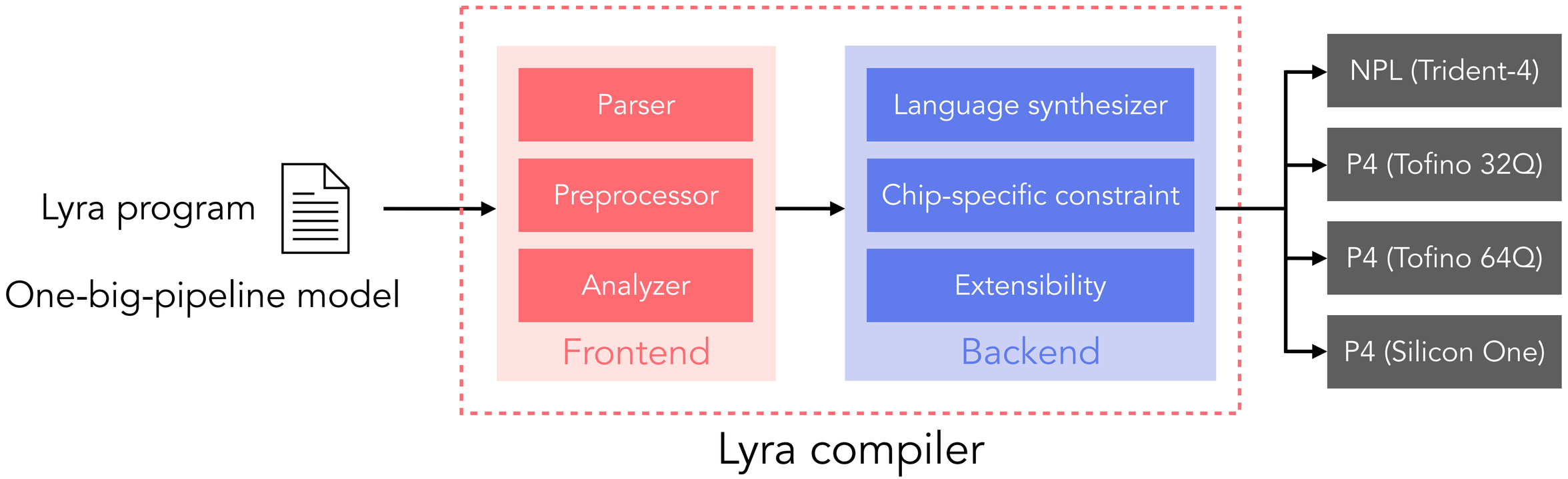
No code is duplicated

No code is missing

No reordering

Lyra: A high-level data plane language & compiler

Alibaba Cloud
Worldwide Cloud Services Partner



Evaluation

Demo for portability, extensibility and composition

Lyra can reduce resource usage

Alibaba Cloud
Worldwide Cloud Services Partner

Program	P4$_{14}$				Lyra								
						Synthesized P4$_{14}$				Synthesized NPL			
	LoC / Logic LoC	Tables	Actions	Registers	Lyra LoC / Logic LoC	Lyra Compile Time	Tables	Actions	Registers	Lyra Compile Time	Tables	Registers	Longest Code Path
Ingress INT	308/99	9	8	0	207/62	0.987s	8	7	0	0.78s	4	0	9
Transit INT	275/66	6	6	0	193/46	0.914s	5	5	0	0.72s	2	0	4
Egress INT	282/73	7	7	0	197/47	0.897s	6	6	0	0.73s	2	0	4
Speedlight	453/351	21	23	6	194/97	1.352s	16	20	6	0.95s	9	6	18
NetCache	1137/937	96	96	40	372/153	1.909s	12	14	40	1.17s	3	40	20
NetChain	319/211	16	16	2	177/73	1.530s	13	16	2	0.85s	6	2	18
NetPaxos	241/140	6	11	5	150/69	1.158s	6	11	5	0.84s	3	5	4
flowlet_switching	195/130	8	7	2	113/43	0.91s	8	7	2	0.70s	4	2	12
simple_router	101/66	4	4	0	72/31	0.852s	4	4	0	0.67s	3	0	10
switch	4924/3876	131	363	0	4151/2563	33.6s	131	363	0	19.4s	125	0	53

Lyra can reduce resource usage

Alibaba Cloud
Worldwide Cloud Services Partner

Program	$P4_{14}$				Lyra								
						Synthesized $P4_{14}$				Synthesized NPL			
	LoC / Logic LoC	Tables	Actions	Registers	Lyra LoC / Logic LoC	Lyra Compile Time	Tables	Actions	Registers	Lyra Compile Time	Tables	Registers	Longest Code Path
Ingress INT	308/99	9	8	0	207/62	0.987s	8	7	0	0.78s	4	0	9
Transit INT	275/66	6	6	0	193/46	0.914s	5	5	0	0.72s	2	0	4
Egress INT	282/73	7	7	0	197/47	0.897s	6	6	0	0.73s	2	0	4
Speedlight	453/351	21	23	6	194/97	1.352s	16	20	6	0.95s	9	6	18
NetCache	1137/937	96	96	40	372/153	1.909s	12	14	40	1.17s	3	40	20
NetChain	319/211	16	16	2	177/73	1.530s	13	16	2	0.85s	6	2	18
NetPaxos	241/140	6	11	5	150/69	1.158s	6	11	5	0.84s	3	5	4
flowlet_switching	195/130	8	7	2	113/43	0.91s	8	7	2	0.70s	4	2	12
simple_router	101/66	4	4	0	72/31	0.852s	4	4	0	0.67s	3	0	10
switch	4924/3876	131	363	0	4151/2563	33.6s	131	363	0	19.4s	125	0	53

Lyra can reduce resource usage

Alibaba Cloud
Worldwide Cloud Services Partner

Program	$P4_{14}$				Lyra								
						Synthesized $P4_{14}$				Synthesized NPL			
	LoC / Logic LoC	Tables	Actions	Registers	Lyra LoC / Logic LoC	Lyra Compile Time	Tables	Actions	Registers	Lyra Compile Time	Tables	Registers	Longest Code Path
Ingress INT	308/99	9	8	0	207/62	0.987s	8	7	0	0.78s	4	0	9
Transit INT	275/66	6	6	0	193/46	0.914s	5	5	0	0.72s	2	0	4
Egress INT	282/73	7	7	0	197/47	0.897s	6	6	0	0.73s	2	0	4
Speedlight	453/351	21	23	6	194/97	1.352s	16	20	6	0.95s	9	6	18
NetCache	1137/937	96	96	40	372/153	1.909s	12	14	40	1.17s	3	40	20
NetChain	319/211	16	16	2	177/73	1.530s	13	16	2	0.85s	6	2	18
NetPaxos	241/140	6	11	5	150/69	1.158s	6	11	5	0.84s	3	5	4
flowlet_switching	195/130	8	7	2	113/43	0.91s	8	7	2	0.70s	4	2	12
simple_router	101/66	4	4	0	72/31	0.852s	4	4	0	0.67s	3	0	10
switch	4924/3876	131	363	0	4151/2563	33.6s	131	363	0	19.4s	125	0	53

Lyra can reduce resource usage

Alibaba Cloud
Worldwide Cloud Services Partner

Program	P4$_{14}$				Lyra								
						Synthesized P4$_{14}$				Synthesized NPL			
	LoC / Logic LoC	Tables	Actions	Registers	Lyra LoC / Logic LoC	Lyra Compile Time	Tables	Actions	Registers	Lyra Compile Time	Tables	Registers	Longest Code Path
Ingress INT	308/99	9	8	0	207/62	0.987s	8	7	0	0.78s	4	0	9
Transit INT	275/66	6	6	0	193/46	0.914s	5	5	0	0.72s	2	0	4
Egress INT	282/73	7	7	0	197/47	0.897s	6	6	0	0.73s	2	0	4
Speedlight	453/351	21	23	6	194/97	1.352s	16	20	6	0.95s	9	6	18
NetCache	1137/937	96	96	40	372/153	1.909s	12	14	40	1.17s	3	40	20
NetChain	319/211	16	16	2	177/73	1.530s	13	16	2	0.85s	6	2	18
NetPaxos	241/140	6	11	5	150/69	1.158s	6	11	5	0.84s	3	5	4
flowlet_switching	195/130	8	7	2	113/43	0.91s	8	7	2	0.70s	4	2	12
simple_router	101/66	4	4	0	72/31	0.852s	4	4	0	0.67s	3	0	10
switch	4924/3876	131	363	0	4151/2563	33.6s	131	363	0	19.4s	125	0	53

Lyra can reduce resource usage

Alibaba Cloud
Worldwide Cloud Services Partner

Program	$P4_{14}$				Lyra								
						Synthesized $P4_{14}$				Synthesized NPL			
	LoC / Logic LoC	Tables	Actions	Registers	Lyra LoC / Logic LoC	Lyra Compile Time	Tables	Actions	Registers	Lyra Compile Time	Tables	Registers	Longest Code Path
Ingress INT	308/99	9	8	0	207/62	0.987s	8	7	0	0.78s	4	0	9
Transit INT	275/66	6	6	0	193/46	0.914s	5	5	0	0.72s	2	0	4
Egress INT	282/73	7	7	0	197/47	0.897s	6	6	0	0.73s	2	0	4
Speedlight	453/351	21	23	6	194/97	1.352s	16	20	6	0.95s	9	6	18
NetCache	1137/937	96	96	40	372/153	1.909s	12	14	40	1.17s	3	40	20
NetChain	319/211	16	16	2	177/73	1.530s	13	16	2	0.85s	6	2	18
NetPaxos	241/140	6	11	5	150/69	1.158s	6	11	5	0.84s	3	5	4
flowlet_switching	195/130	8	7	2	113/43	0.91s	8	7	2	0.70s	4	2	12
simple_router	101/66	4	4	0	72/31	0.852s	4	4	0	0.67s	3	0	10
switch	4924/3876	131	363	0	4151/2563	33.6s	131	363	0	19.4s	125	0	53

Lyra can reduce resource usage

Alibaba Cloud
Worldwide Cloud Services Partner

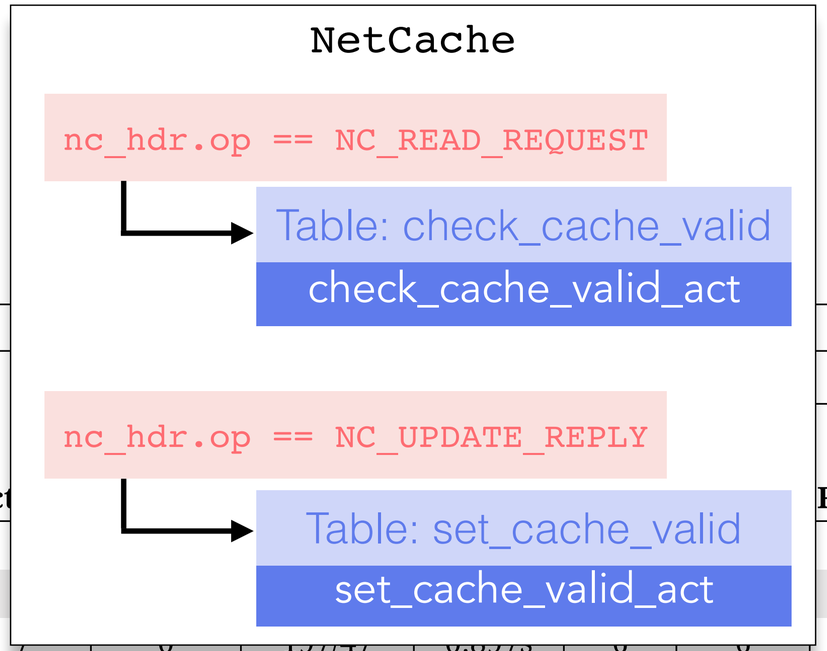
Program	P4$_{14}$				Lyra								
						Synthesized P4$_{14}$				Synthesized NPL			
	LoC / Logic LoC	Tables	Actions	Registers	Lyra LoC / Logic LoC	Lyra Compile Time	Tables	Actions	Registers	Lyra Compile Time	Tables	Registers	Longest Code Path
Ingress INT	308/99	9	8	0	207/62	0.987s	8	7	0	0.78s	4	0	9
Transit INT	275/66	6	6	0	193/46	0.914s	5	5	0	0.72s	2	0	4
Egress INT	282/73	7	7	0	197/47	0.897s	6	6	0	0.73s	2	0	4
Speedlight	453/351	21	23	6	194/97	1.352s	16	20	6	0.95s	9	6	18
NetCache	1137/937	96	96	40	372/153	1.909s	12	14	40	1.17s	3	40	20
NetChain	319/211	16	16	2	177/73	1.530s	13	16	2	0.85s	6	2	18
NetPaxos	241/140	6	11	5	150/69	1.158s	6	11	5	0.84s	3	5	4
flowlet_switching	195/130	8	7	2	113/43	0.91s	8	7	2	0.70s	4	2	12
simple_router	101/66	4	4	0	72/31	0.852s	4	4	0	0.67s	3	0	10
switch	4924/3876	131	363	0	4151/2563	33.6s	131	363	0	19.4s	125	0	53

Lyra can reduce resource usage

Alibaba Cloud
Worldwide Cloud Services Partner

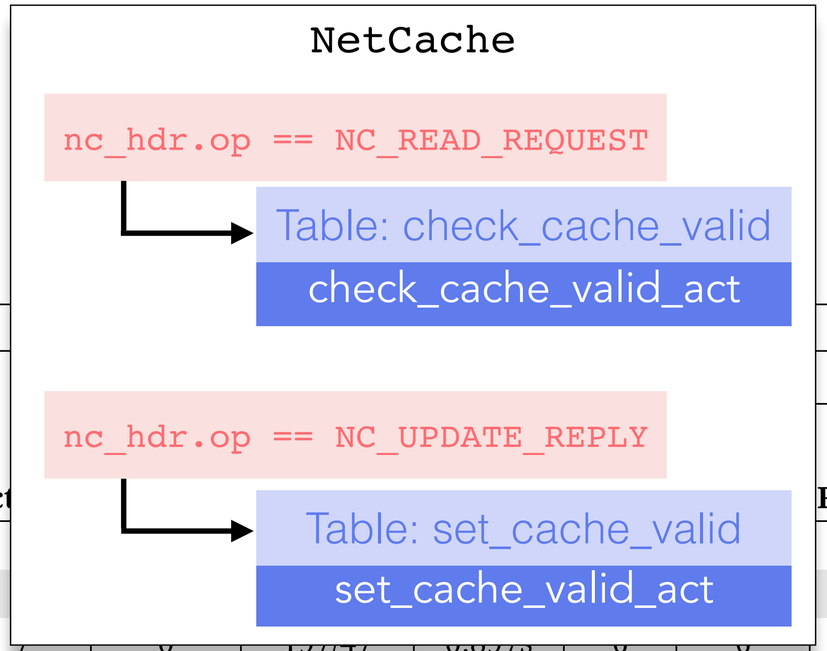
Program	$P4_{14}$				Lyra								
						Synthesized $P4_{14}$				Synthesized NPL			
	LoC / Logic LoC	Tables	Actions	Registers	Lyra LoC / Logic LoC	Lyra Compile Time	Tables	Actions	Registers	Lyra Compile Time	Tables	Registers	Longest Code Path
Ingress INT	308/99	9	8	0	207/62	0.987s	8	7	0	0.78s	4	0	9
Transit INT	275/66	6	6	0	193/46	0.914s	5	5	0	0.72s	2	0	4
Egress INT	282/73	7	7	0	197/47	0.897s	6	6	0	0.73s	2	0	4
Speedlight	453/351	21	23	6	194/97	1.352s	16	20	6	0.95s	9	6	18
NetCache	1137/937	96	96	40	372/153	1.909s	12	14	40	1.17s	3	40	20
NetChain	319/211	16	16	2	177/73	1.530s	13	16	2	0.85s	6	2	18
NetPaxos	241/140	6	11	5	150/69	1.158s	6	11	5	0.84s	3	5	4
flowlet_switching	195/130	8	7	2	113/43	0.91s	8	7	2	0.70s	4	2	12
simple_router	101/66	4	4	0	72/31	0.852s	4	4	0	0.67s	3	0	10
switch	4924/3876	131	363	0	4151/2563	33.6s	131	363	0	19.4s	125	0	53

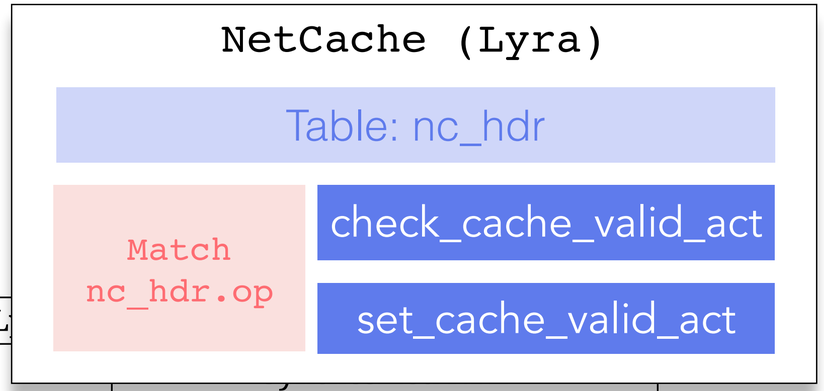
Lyra can reduce resource usage



Program	$P4_{14}$								Lyra				
										Synthesized NPL			
	LoC / Logic LoC	Tables	Act						Registers	Lyra Compile Time	Tables	Registers	Longest Code Path
Ingress INT	308/99	9							0	0.78s	4	0	9
Transit INT	275/66	6							0	0.72s	2	0	4
Egress INT	282/73	7							0	0.73s	2	0	4
Speedlight	453/351	21	23	6	194/97	1.352s	16	20	6	0.95s	9	6	18
NetCache	1137/937	96	96	40	372/153	1.909s	12	14	40	1.17s	3	40	20
NetChain	319/211	16	16	2	177/73	1.530s	13	16	2	0.85s	6	2	18
NetPaxos	241/140	6	11	5	150/69	1.158s	6	11	5	0.84s	3	5	4
flowlet_switching	195/130	8	7	2	113/43	0.91s	8	7	2	0.70s	4	2	12
simple_router	101/66	4	4	0	72/31	0.852s	4	4	0	0.67s	3	0	10
switch	4924/3876	131	363	0	4151/2563	33.6s	131	363	0	19.4s	125	0	53

Lyra can reduce resource usage





Program	P4$_{14}$ LoC / Logic LoC	Tables	Act						L Registers	Lyra Compile Time	Tables	Registers	Longest Code Path
Ingress INT	308/99	9							0	0.78s	4	0	9
Transit INT	275/66	6							0	0.72s	2	0	4
Egress INT	282/73	7							0	0.73s	2	0	4
Speedlight	453/351	21	23	6	194/97	1.352s	16	20	6	0.95s	9	6	18
NetCache	1137/937	96	96	40	372/153	1.909s	12	14	40	1.17s	3	40	20
NetChain	319/211	16	16	2	177/73	1.530s	13	16	2	0.85s	6	2	18
NetPaxos	241/140	6	11	5	150/69	1.158s	6	11	5	0.84s	3	5	4
flowlet_switching	195/130	8	7	2	113/43	0.91s	8	7	2	0.70s	4	2	12
simple_router	101/66	4	4	0	72/31	0.852s	4	4	0	0.67s	3	0	10
switch	4924/3876	131	363	0	4151/2563	33.6s	131	363	0	19.4s	125	0	53

Conclusion

- Lyra is the first high-level data plane language and compiler that achieves portability, extensibility and composition.
- Lyra offers a one-big-pipeline programming model and can generate runnable chip-specific code across multiple switches.
- The programs generated by Lyra use fewer hardware resources than human-written programs.

Thanks !