

Programmable Transport on SmartNICs

Mina Tahmasbi Arashloo
University of Waterloo

SIGCOMM 2026

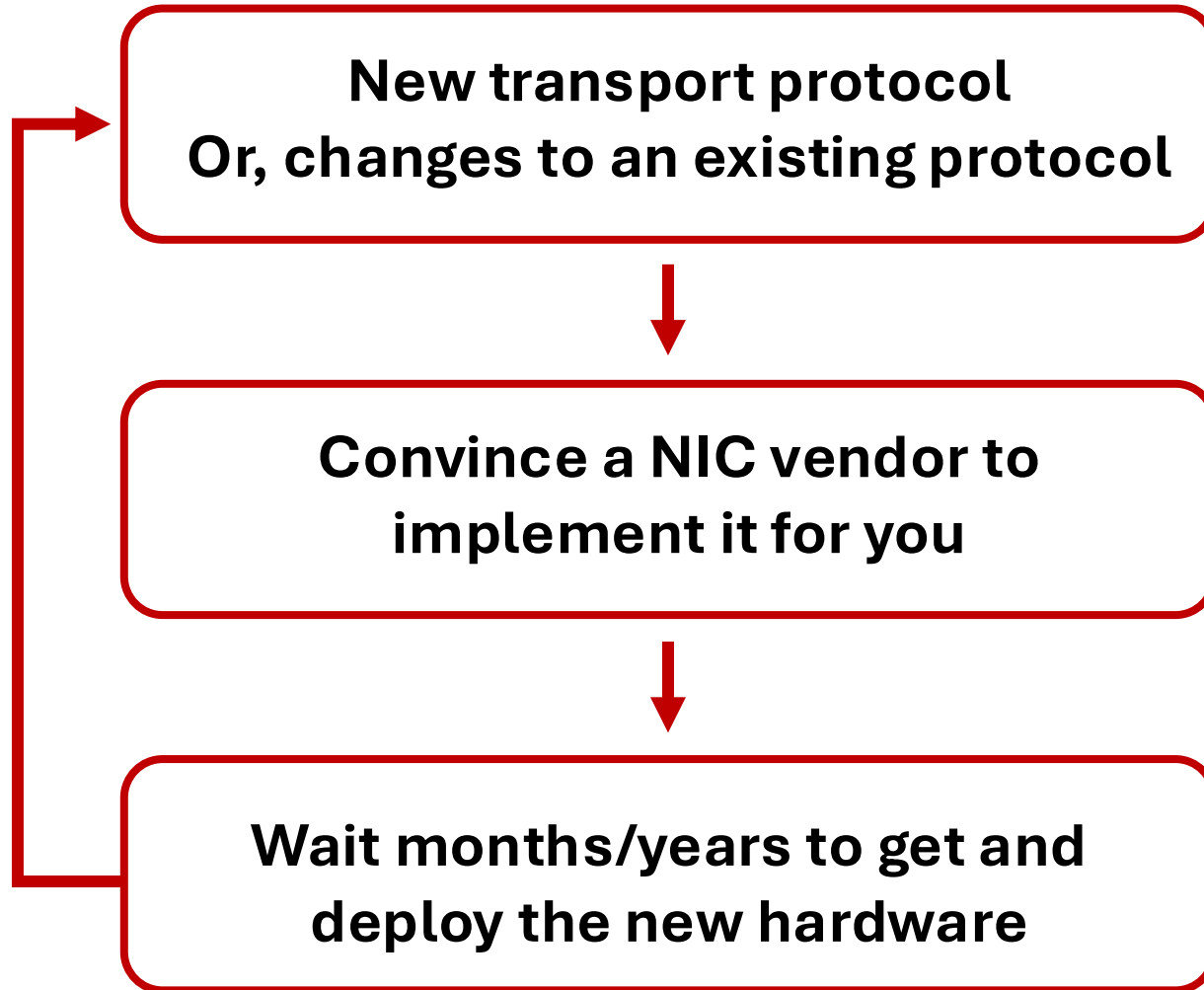
Why transport on NIC?

- Higher performance
 - Higher throughput, lower latency, lower jitter
- Higher efficiency
 - Special network processing → Specialized hardware can do “better”
 - CPU freed-up for actual general purpose work
- Strategic location
 - All data transfers to/from CPU and accelerators pass through the NIC
- Needed for any offload operating on message
 - as opposed to individual packets

No “one-size-fits-all” transport protocol

- Depends on workload characteristics and requirements
- Depends on network architecture and properties
- A continuously evolving set of protocols and their variants
 - 40+ years of Internet transport
 - 10+ years of data center transport
 - Growing set of proposals for AI infrastructure transport

Fixed-function ASIC is not the answer

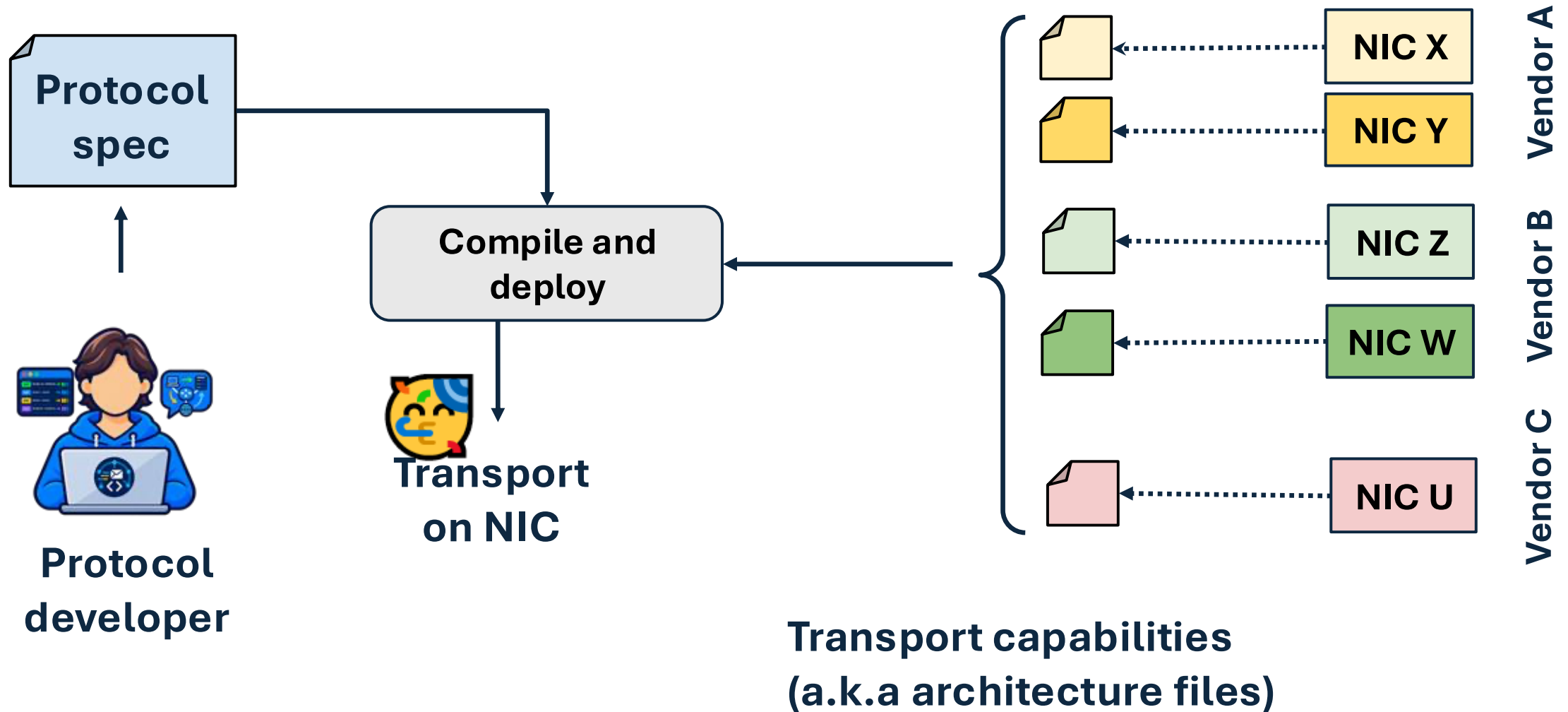


Sounds familiar?

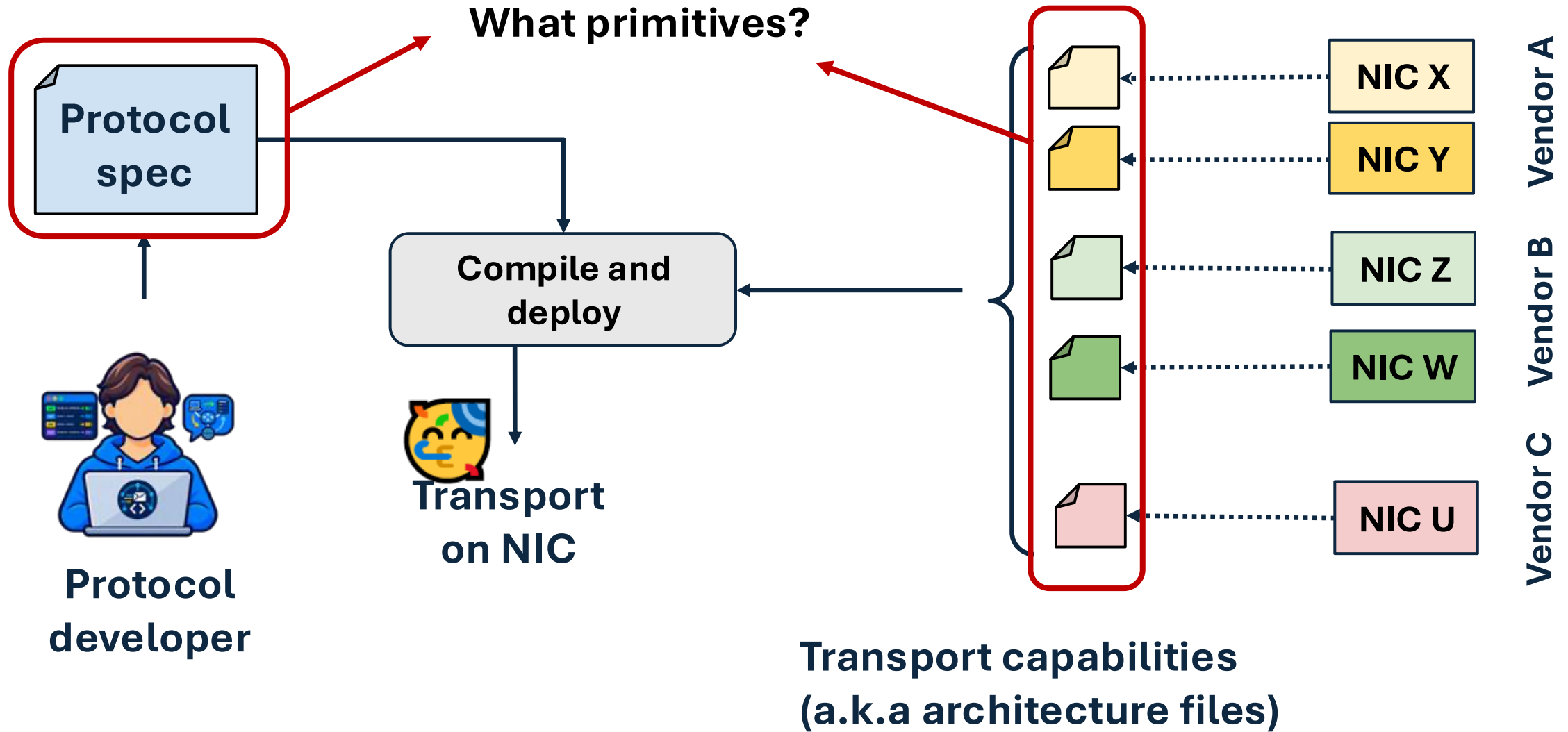
This is where we were with
L2/L3 packet processing in
early 2010s

Led to programmable data
planes and P4

Programmable transport on SmartNICs



Programmable transport on SmartNICs



What “transport primitives” do we need?



Transport control path (fast)

Congestion control: Get stats from data path, revisit data-path params



Transport management path (slow)

option negotiation, key exchange, exceptions, ...



Transport data path (very fast)

Reassemble data packets

Generate
acks

Generate
data packets

Path selection

Loss detection
and recovery

Enforce
pacing rules

What programmability exists today?



Transport control path (fast)

Congestion control: Get stats from data path, revisit data-path params



Transport management path (slow)

option negotiation, key exchange, exceptions, ...



Transport data path (very fast)

Reassemble data packets

Generate
acks

Generate
data packets

Path selection

Loss detection
and recovery

Enforce
pacing rules

What programmability exists today?



Transport control path (fast)

Congestion control: Get stats from data path, revisit data-path params



Transport management path (slow)

option negotiation, key exchange, exceptions, ...



- Runs on off-path cores
- Programmability often exposed to users
- Not where the programmability-performance tension often manifests
- Not the focus today

data packets

data packets

path selection

pacing rules

What programmability exists today?



Transport control path (fast)

Congestion control: Get stats from data path, revisit data-path params



Transport management path (slow)

option negotiation, key exchange, exceptions, ...



Transport data path (very fast)

Reassemble data packets

Generate
acks

Generate
data packets

Loss detection
and recovery

Path selection

Enforce
pacing rules

What programmability exists today?



Transport control path (fast)

Congestion control: Get stats from data path, revisit data-path params

- Most common arrangement today
 - Intel IPU, BF3, ...
- Parametrized but otherwise **fixed-function** transport **data path**
 - Falcon, RDMA, UEC, ...
- Tightly integrated on-path cores for **control-path reconfiguration**.



Transport data path (very fast)

Reassemble
data packets

Generate
acks

Generate
data packets

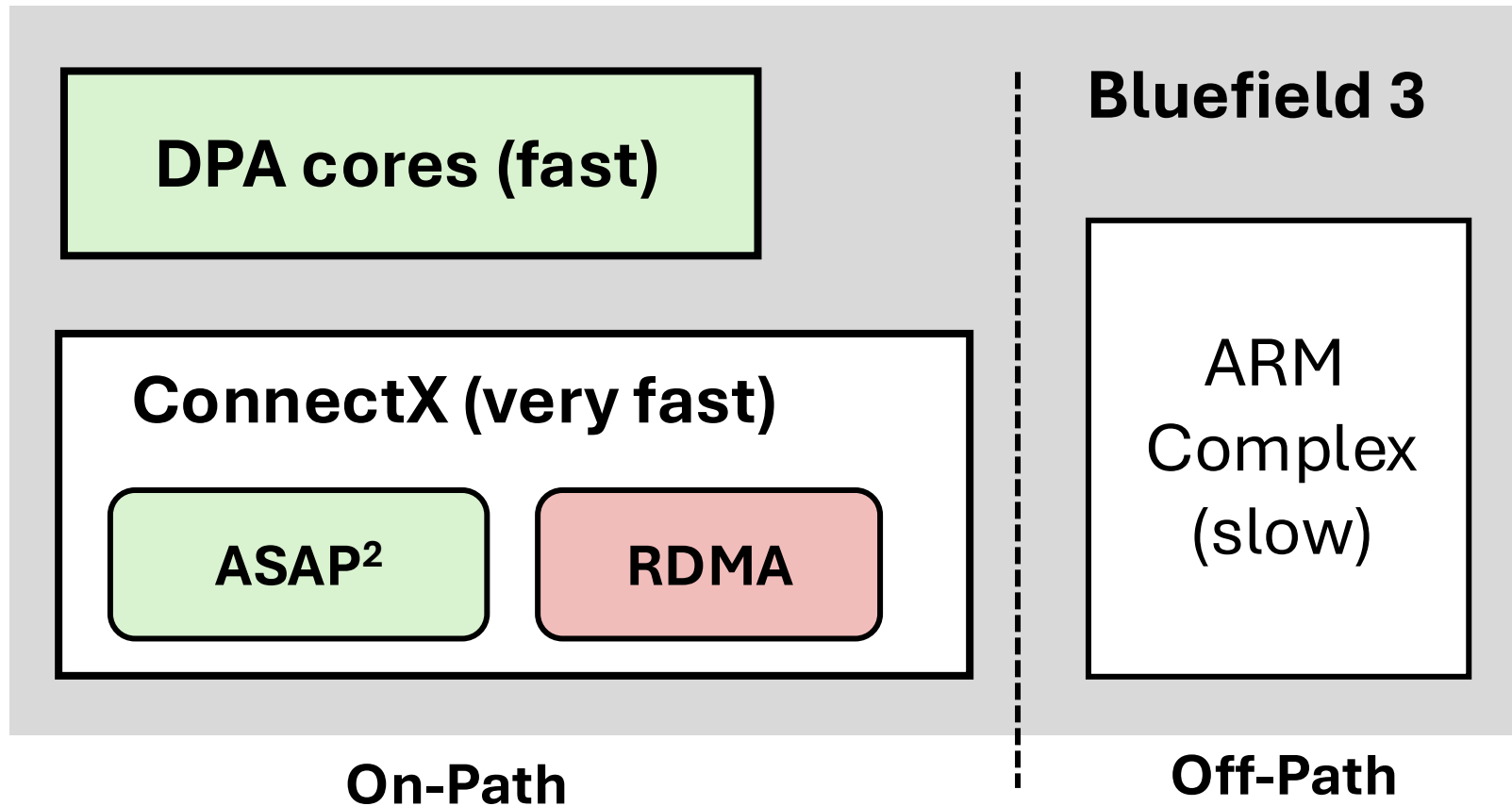
Loss detection
and recovery

Path selection

Enforce
pacing rules

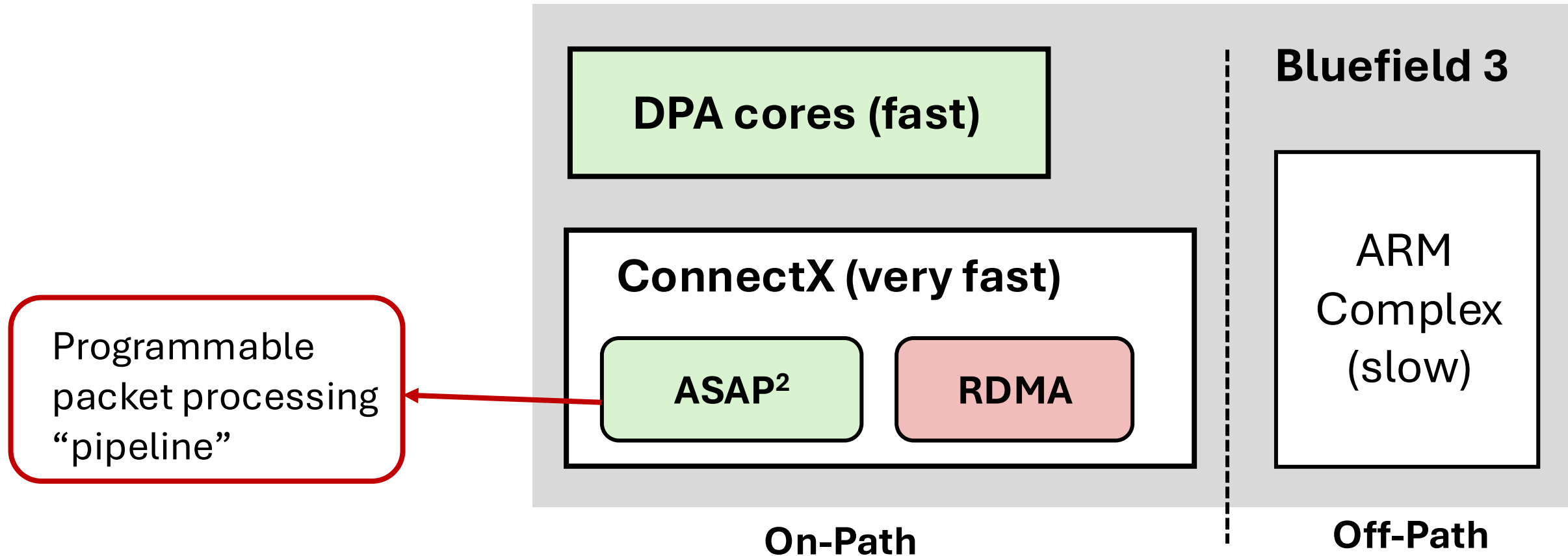
This tutorial – Exercise 2

Programmable congestion control on BlueField3



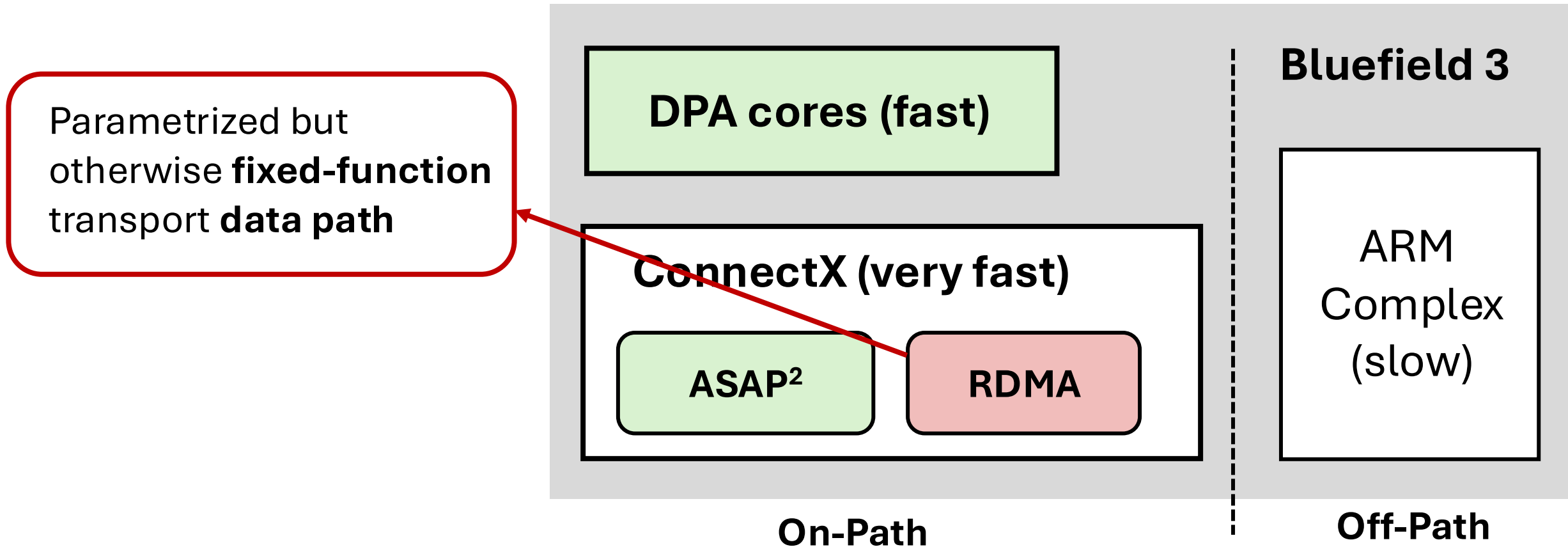
This tutorial – Exercise 2

Programmable congestion control on BlueField3



This tutorial – Exercise 2

Programmable congestion control on BlueField3

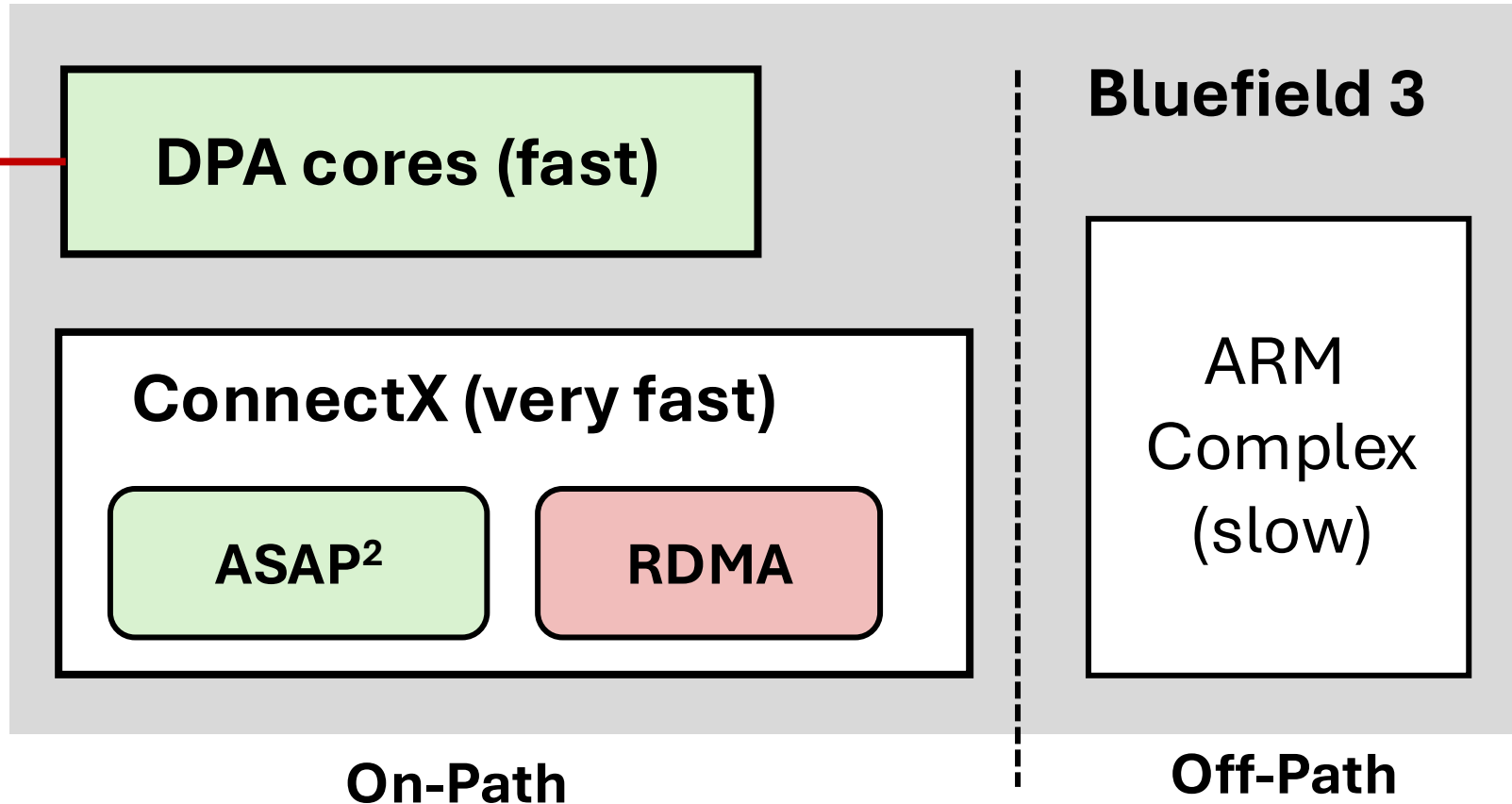


This tutorial – Exercise 2

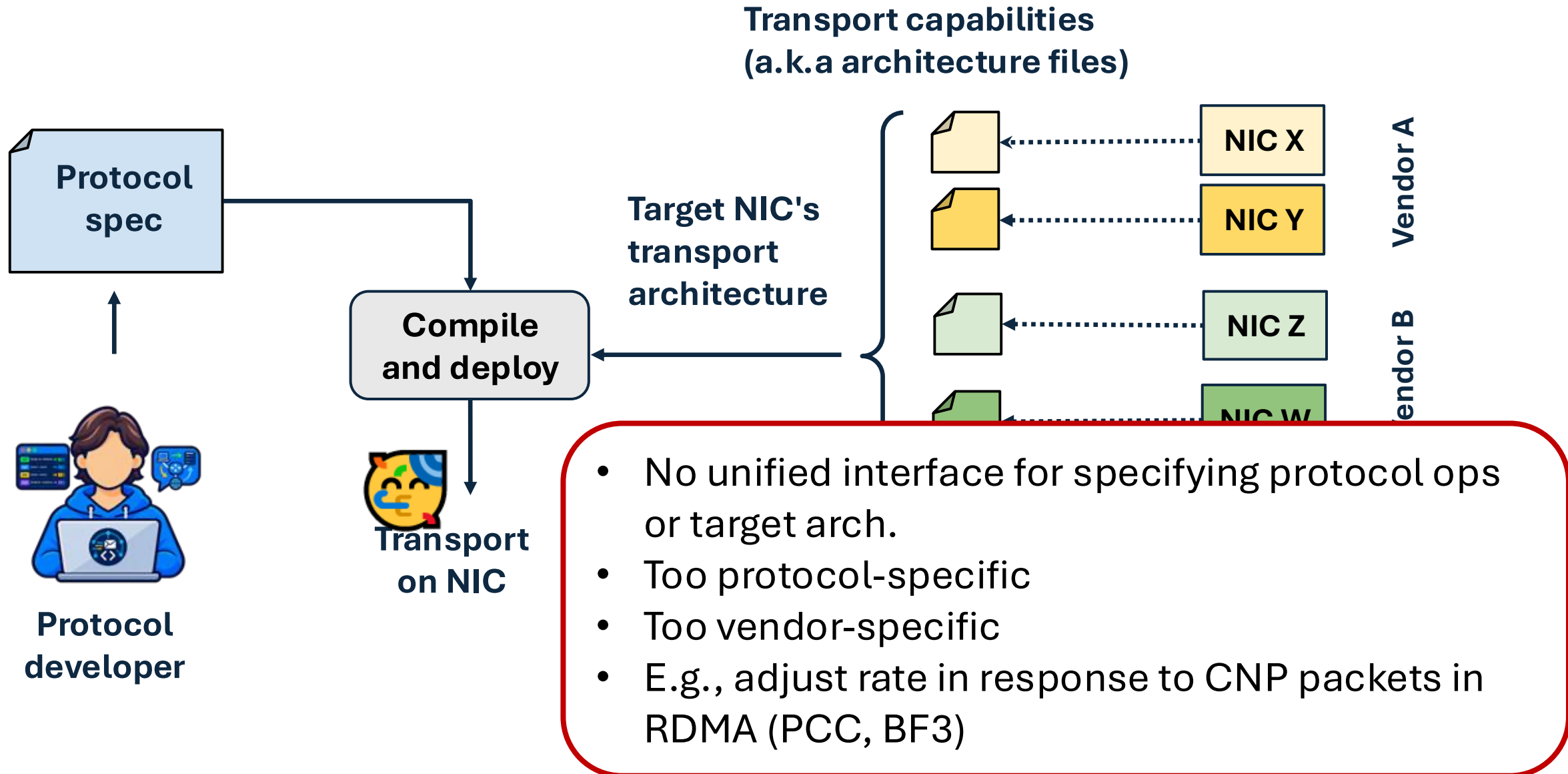
Programmable congestion control on BlueField3

Programmable congestion control (PCC)

- Get CNPs
- Decide rate
- Use API call to set rate on data path



We are not quite at this state yet



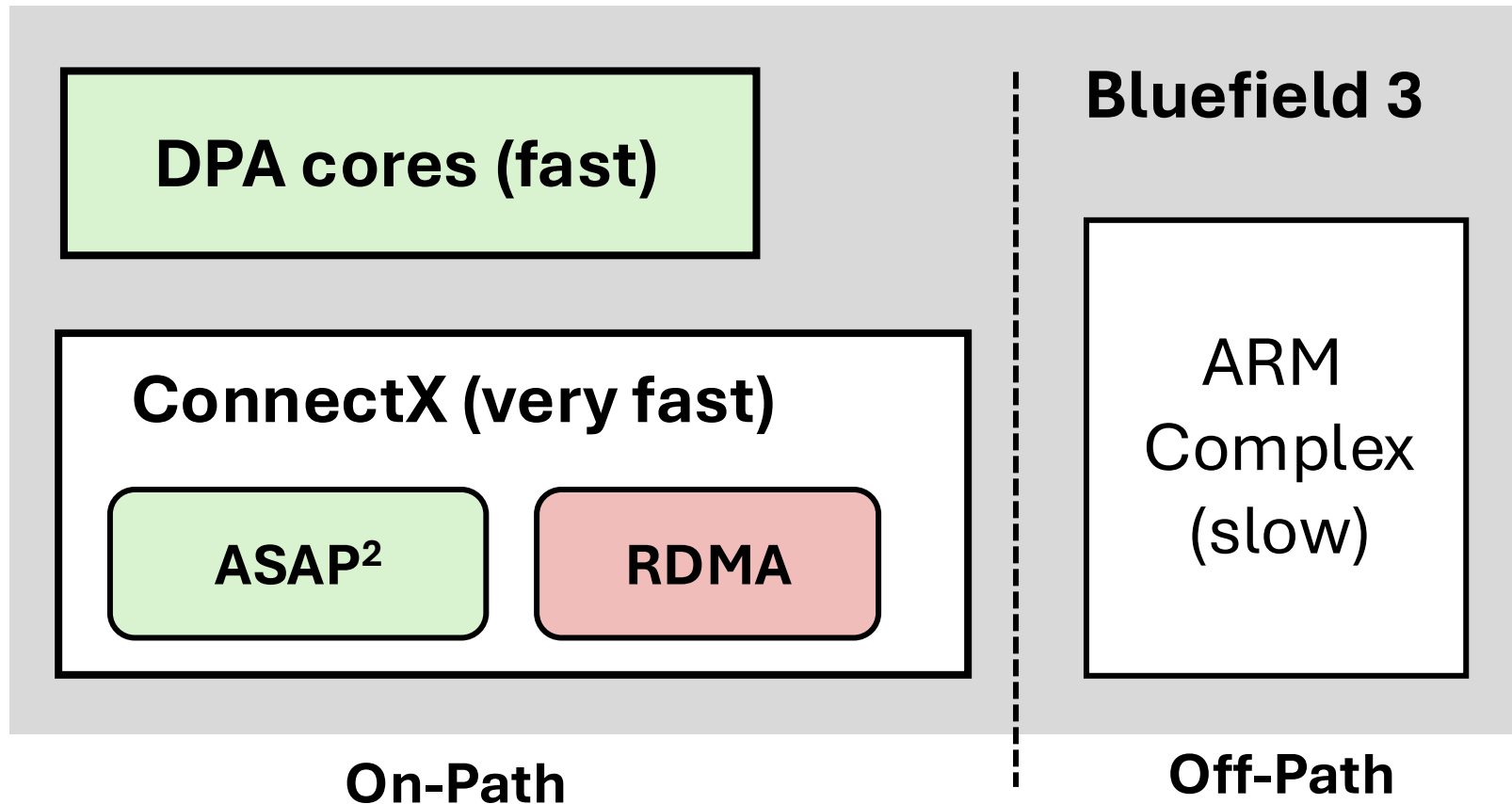
This tutorial – (Bonus) Exercise 3

Programmable path selection on BlueField3

- Existing AI transport heavily uses multi-path
 - Each “connection” maintains a set of viable paths
 - Packets are sprayed along multiple paths
 - The sender tracks congestion signals along paths
 - It tags packets with paths based on their “score”
- This exercise “emulates” this setup on BlueField3
 - PCC can only “set-rate”, not change other configurations
 - Congestion signal can only come in the form of per-queue-pair CNPs
 - Example of transport programming APIs still being too protocol- and vendor-specific

This tutorial – (Bonus) Exercise 3

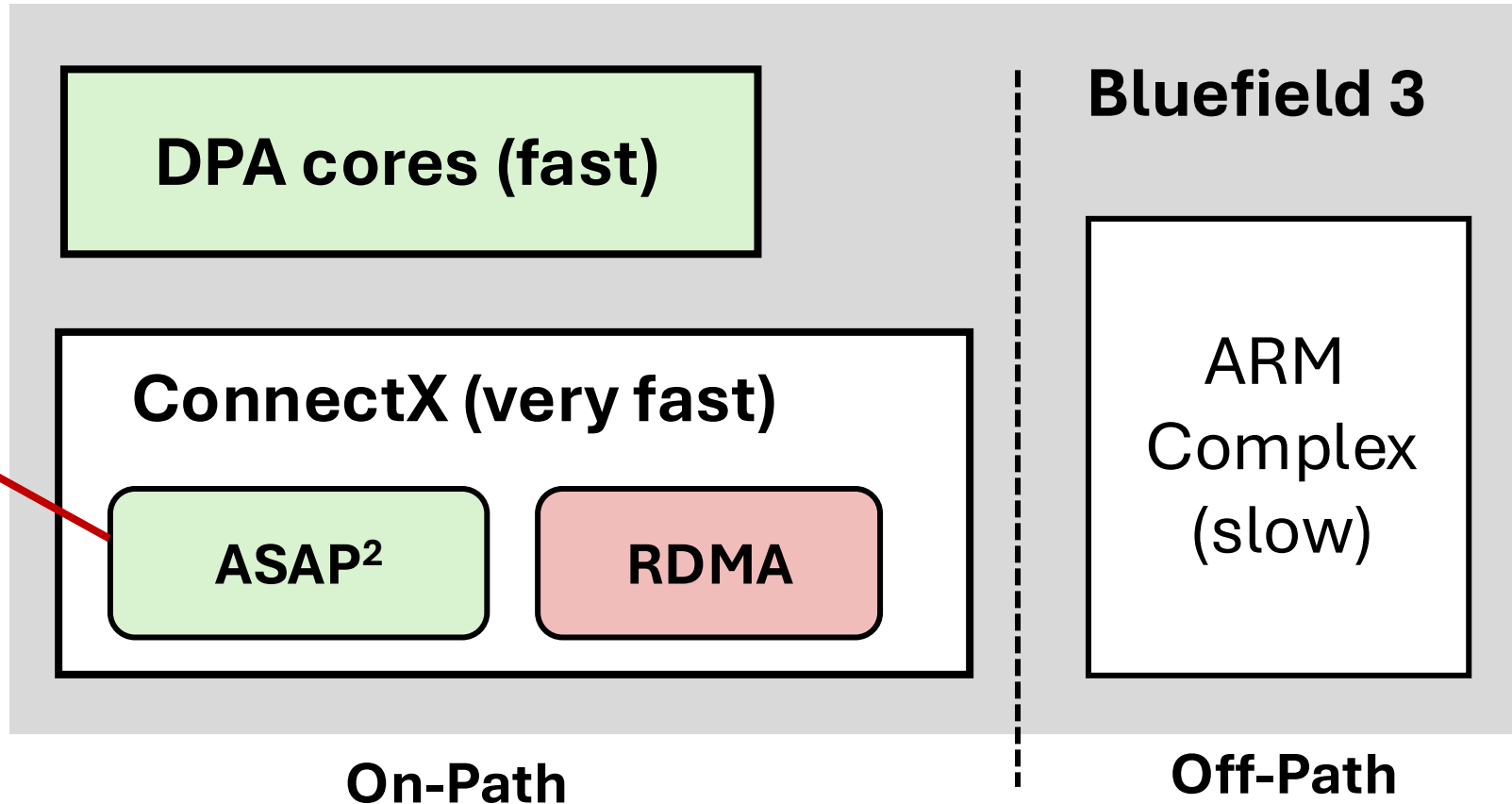
Programmable path selection on BlueField3



This tutorial – (Bonus) Exercise 3

Programmable path selection on BlueField3

- Tags pkts with path IDs
- Split ratio depends on path scores

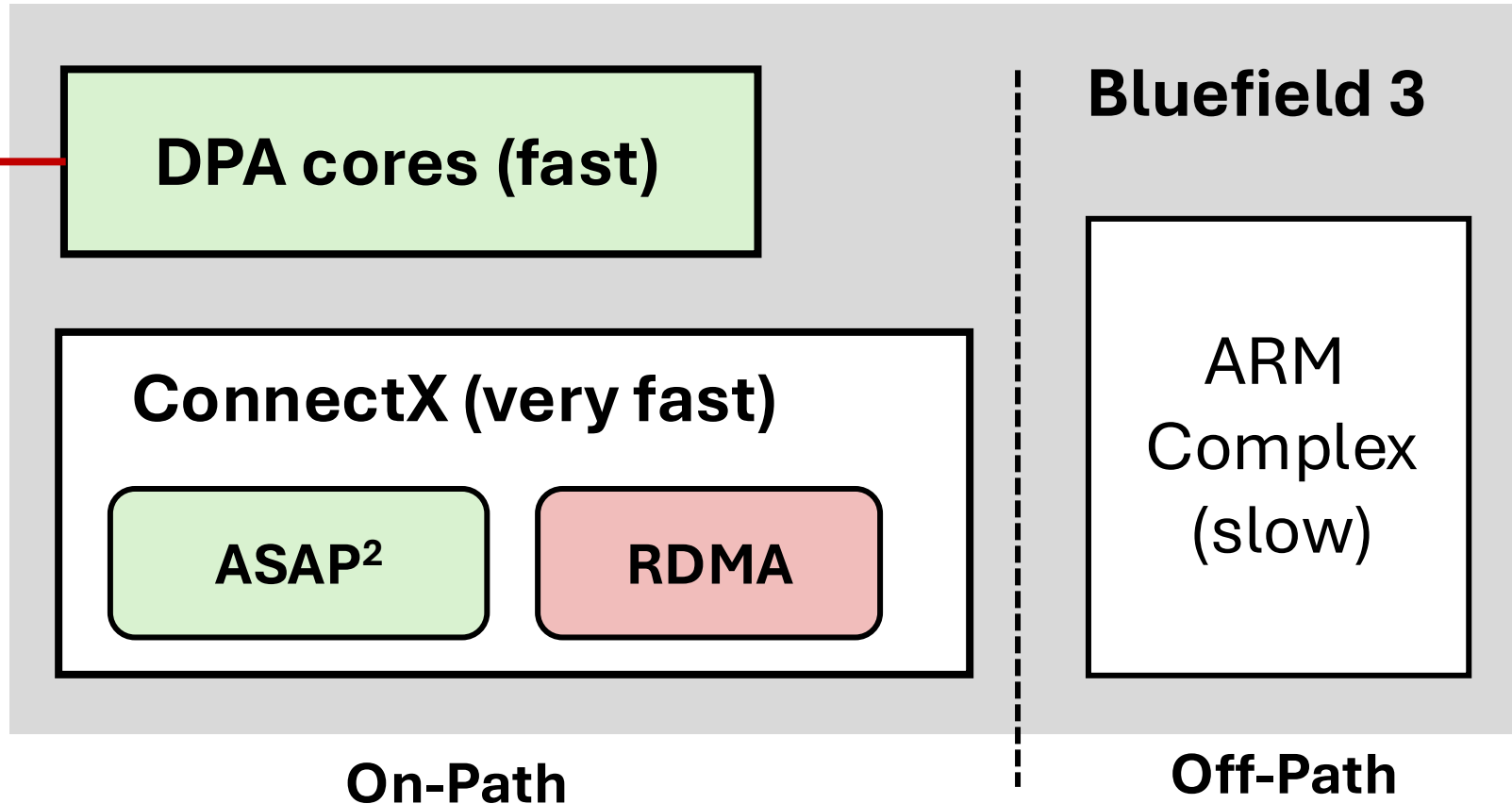


This tutorial – (Bonus) Exercise 3

Programmable path selection on BlueField3

Programmable congestion control (PCC)

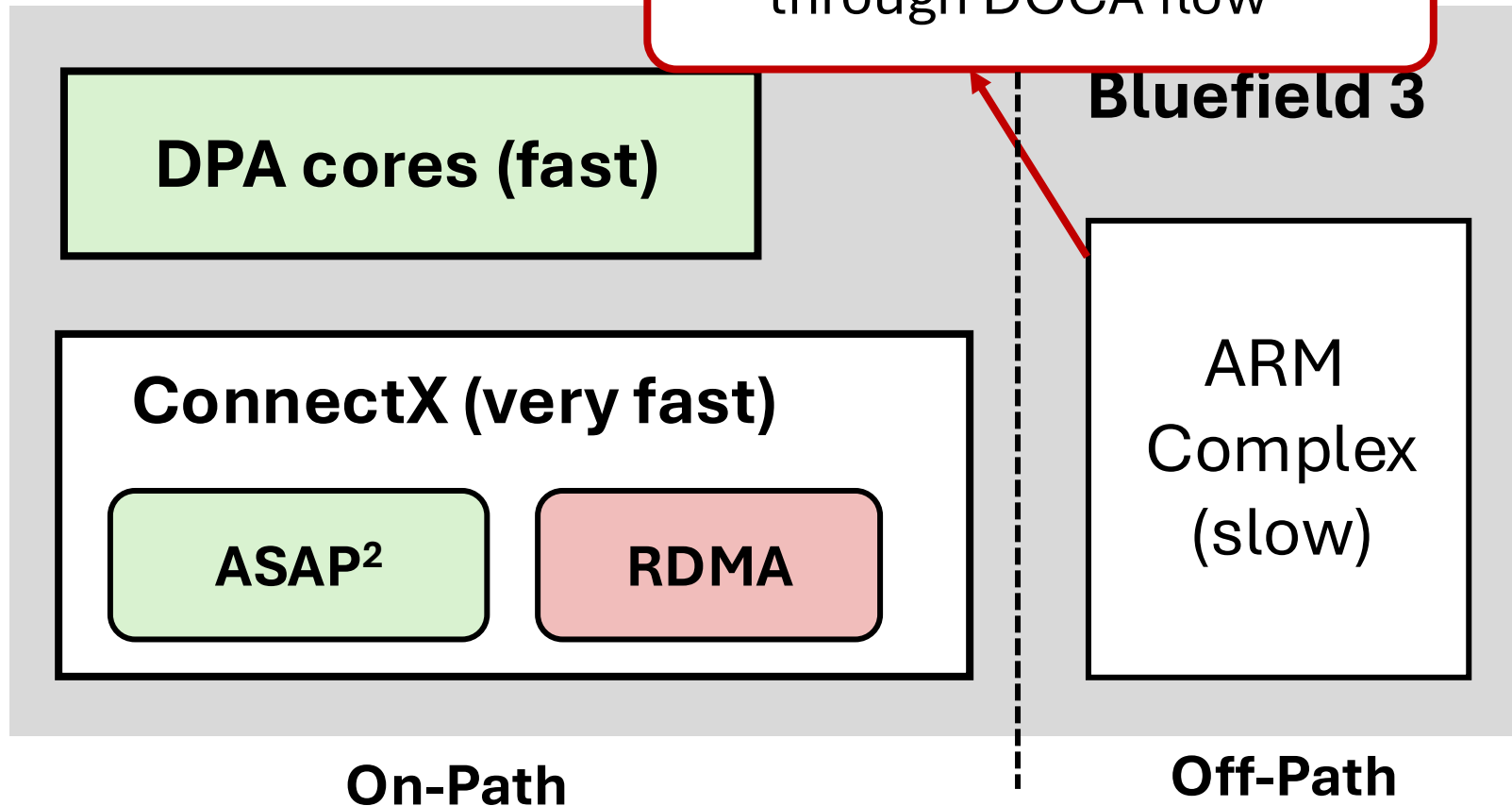
- Get “path” CNPs
- Send stats to ARM cores



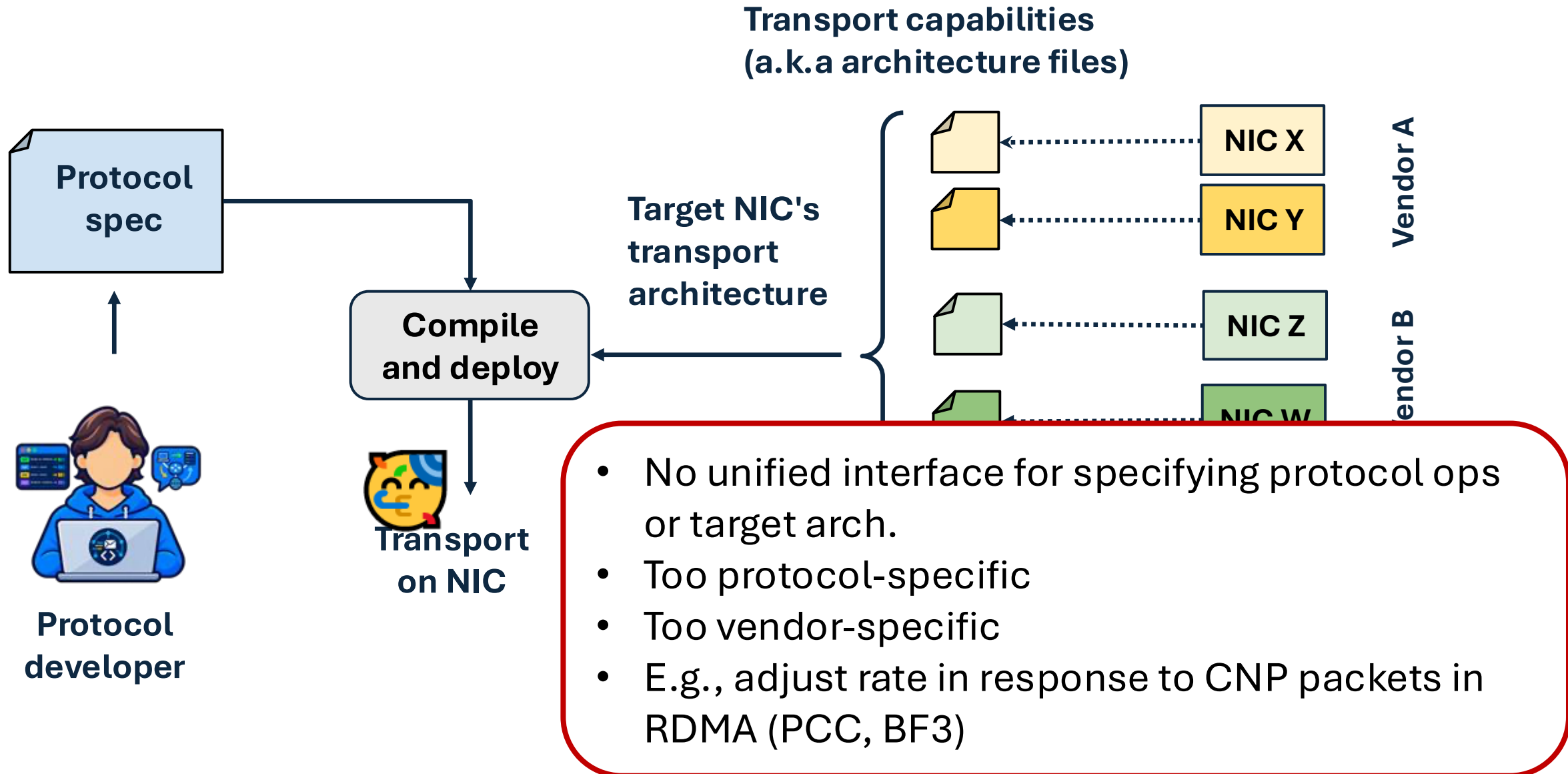
This tutorial – (Bonus) Exercise 3

Programmable path selection on Blue

- Receive stats from DPA
- Recompute path scores
- Configure ASAP² through DOCA flow



We are not quite at this state yet



MTP: A new DSL for programmable transport

MTP Program

```
// decl.           // main
event {...}        main(){
context {...}      register_ep_chains(...);
ev_proc. {...}     register_ctx(...);
dispatch {...}     ...
seg_rule {...}     }
...
```

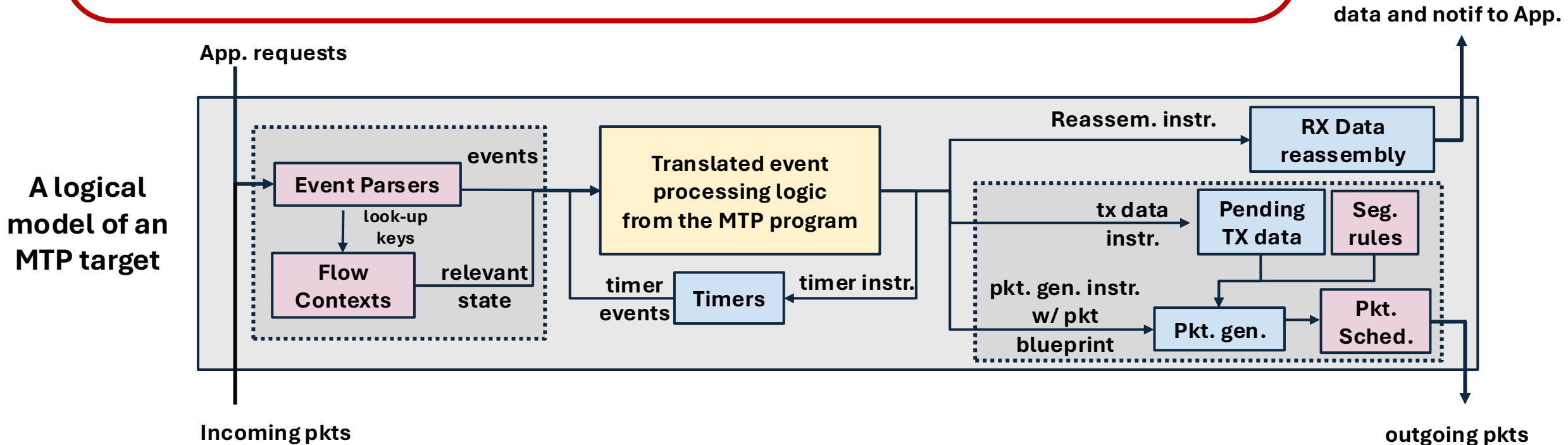
MTP *protocol* programs specify protocol operations

- Transport Event → Process → Transport Instruction
- *All three transport paths* under the same unifying abstraction

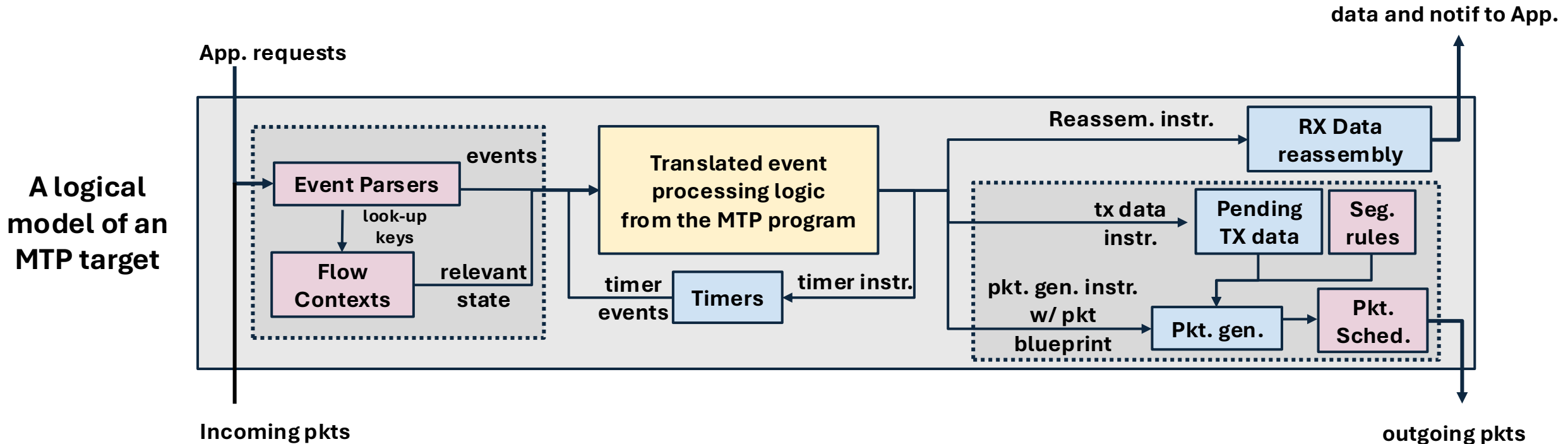
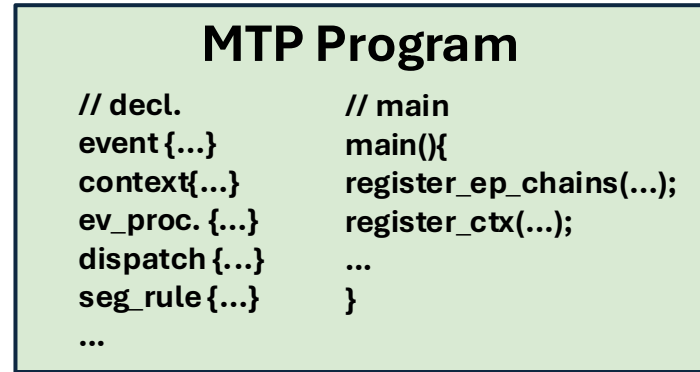
MTP: A new DSL for programmable transport

MTP *target* files specify target programmability and capabilities

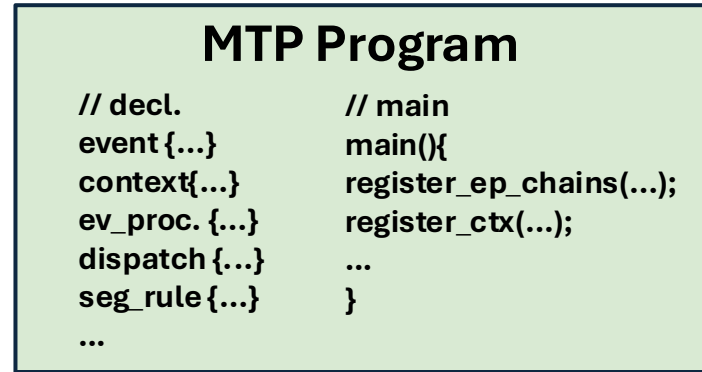
- Limit event processing functions that can be redefined
- Limit instructions that are available for programmers to call
- Limit the available arguments for instructions
- Provide extra instructions (e.g., for encryption, compression, etc.)



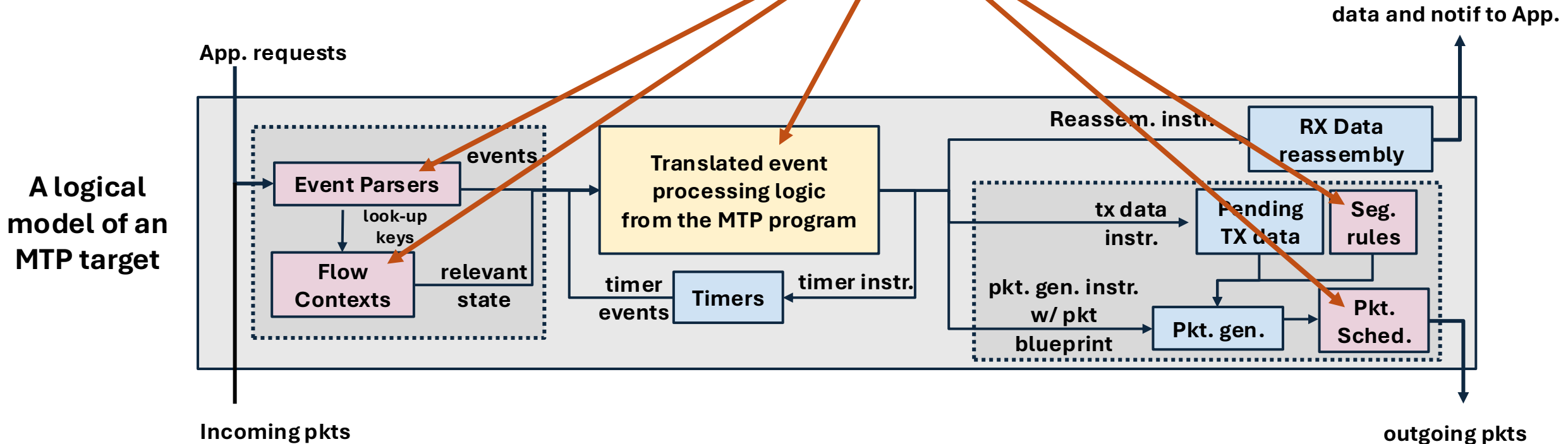
MTP: A new DSL for programmable transport



Modular Transport Programming (MTP)



If you are interested,
please come find us!



Shoutout to the team!

- Fernando Ramos (INESC-ID, IST, University of Lisbon)
 - Mario Baldi (NVIDIA)
 - Muhammad Shahbaz (University of Michigan)
 - Mina Tahmasbi Arashloo (University of Waterloo)
-
- Francisco Pereira (INESC-ID, IST, University of Lisbon)
 - Bilal Saleem (Purdue University / University of Michigan)
 - Tyler Liu (University of Waterloo)
 - Chenxingyu Zhao (University of Washington)