DPDK SUMMIT
Powering the Future of Networking Software

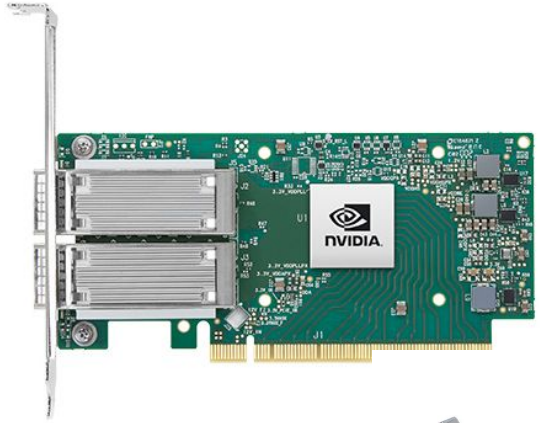
12–13 May 2026

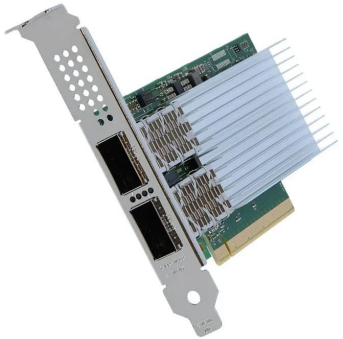
Exploring the Ambiguities and Limits of `rte_flow` Offloading

Pavlína Patová

Motivation & Methodology

- Why?
  - Many customers want RTE Flow
  - How does it work elsewhere?
  - User-friendliness
- What?
  - Identical rules
  - Action behavior
  - Performance & Capabilities
- Where?
  - ConnectX-5 (mlx5)
  - Intel E810-CQDA2 (ice)
  - Silicom N6010 with DynaNIC firmware (nfb)





Available Information Comparison

Mellanox (MLX)

- Two table types: root & non-root
- Rules deleted on device stop
- Capacity: Up to 21,844 priorities in non-root
- Wide patterns & actions variety
- Sequential VLAN items not supported

DynaNIC (NFB)

DYNANIC

Trial Version:

`https://dyna-nic.com/try-now/`

- Four tables: TCAM, EM, LPM, TCAM
- EM/LPM use specific keys
- Small overall capacity
- Design changes (number/size) require support but are easy

Intel (ICE)

- No groups only engines
- SWITCH: EM, large capacity
- ACL: Wild card, small capacity
- FDIR: EM, large capacity
- HASH: RSS
- Refer to `ice_<engine>_supported_pattern`

Order of TCP/IP Layers in the Rule

```
flow create 0 pattern ipv4 / udp …
```


```
flow create 0 pattern udp …
```


```
flow create 0 pattern eth / ipv4 / udp …
```

Identical Rules - MLX

```
flow create 0 group 0 pattern ipv4 dst is 192.168.1.1 / end actions queue
index 1

flow create 0 group 0 pattern ipv4 dst is 192.168.1.1 / end actions queue
index 2
```

```
flow create 0 group 1 pattern ipv4 dst is 192.168.1.1 / end actions queue
index 1

flow create 0 group 1 pattern ipv4 dst is 192.168.1.1 / end actions queue
index 2
```

Not so identical Rules - MLX

```
Rule 1:
flow create 0 pattern ipv4 src is 1.1.1.1 /
end actions queue index 1

Rule 2:
flow create 0 pattern ipv4 dst is 2.2.2.2 /
end actions queue index 2
```

Incoming Packet

IPv4 Header:

src: 1.1.1.1
dst: 2.2.2.2

Packet will arrive at the QUEUE #1

Not so identical Rules - MLX

```
Rule 1:
flow create 0 pattern ipv4 dst is 2.2.2.2 /
end actions queue index 2

Rule 2:
flow create 0 pattern ipv4 src is 1.1.1.1 /
end actions queue index 1
```

Incoming Packet

IPv4 Header:

src: 1.1.1.1
dst: 2.2.2.2

Packet will arrive at the QUEUE #2

Not so identical Rules - MLX

```
Rule 1:
flow create 0 pattern ipv4 / udp dst is 1024
end actions queue index 1

Rule 2:
flow create 0 pattern ipv4 dst is 1.1.1.1 /
end actions queue index 2
```

Incoming Packet

IPv4 Header:	UDP Header:
src: 1.1.1.1	dst: 1024
dst: 2.2.2.2	

- **Packet will arrive at the QUEUE #1**

Not so identical Rules - MLX

```
Rule 1:
flow create 0 pattern ipv4 dst is 1.1.1.1 /
end actions queue index 2

Rule 2:
flow create 0 pattern ipv4 / udp dst is 1024
end actions queue index 1
```

Incoming Packet

IPv4 Header:

src: 1.1.1.1
dst: 2.2.2.2

UDP Header:

dst: 1024

Which rule matches?

Multi-match Logic

1. **Last layer:** The most specific layer takes precedence.
2. **Order of the rule:** If specificity is equal, rule order matters.

Identical Rules - DynaNIC

Rule Execution

Only one rule is executed - the last one - if on the same priority

Action Support

Single fate action within a single rule.
Except for packet duplication.

```
Rule 1:
flow create 0 pattern  ipv4 dst is 1.1.1.1 /
end actions queue index 2

Rule 2:
flow create 0 pattern ipv4 /  udp dst is 1024
end actions queue index 1
```

Engines in ICE

No groups just engines
HASH, **ACL**, **SWITCH** and **FDIR**

HASH Engine
```
flow create 0 ingress pattern eth / ipv4 / udp / end actions rss types ipv4-udp end / end
```

ACL Engine
```
flow create 0 ingress pattern eth / ipv4 src spec 10.0.0.0 src mask 255.255.255.0 / end actions
drop / end
```

SWITCH Engine
```
flow create 0 ingress pattern eth dst is 00:11:22:33:44:55 / end actions drop / end
```

FDIR Engine
```
flow create 0 ingress pattern eth / ipv4 dst is 159.58.1.0 src is 195.168.1.0 / end actions count
identifier 128 / drop / end
```

Identical Rules - ICE

HASH
Last rule is used

ACL
Last rule is used

SWITCH
Try next engine (fdir)

FDIR
Unable to insert identical rules

Engine dependent behavior

COUNT action

+1

Multiple Counters

Support for multiple counters within a single flow rule.

Count ID Behavior

Identifier management and shared counter behavior across rules.

Max Capacity

Hardware limits and maximum available counters.

COUNT Action: Rule Constraints & Capacity

Mellanox (MLX)

- Only one COUNT action per rule.
- No quantity restrictions found.

DynaNIC (NFB)

- No restriction in rule; only last ID is used.
- No quantity restrictions found.

Intel (ICE)

- Only one COUNT action per rule.
- Maximum total COUNT actions available 256

Count ID behavior: Cross-rule sharing

```
flow create 0 pattern ipv4 dst is 1.1.1.1 / end actionscount identifier 1 ...
flow create 0 pattern ipv4 dst is 2.2.2.2 / end actionscount identifier 1 ...
```

IPv4 Packet

src: 1.1.1.1
dst: 2.2.2.2

```
testpmd> flow query 0 0 count
hits_set: 1
bytes_set: 1
hits: 1
bytes: 60
```

```
testpmd> flow query 0 1 count
hits_set: 1
bytes_set: 1
hits: 1
bytes: 60
```

JUMP action

Available groups

Overview of supported flow groups across different hardware.

Jump restrictions

Specific limitations on where packets can be steered, such as forward-only jumps.

Behaviour quirks

Non-standard behaviors or unique hardware-specific implementation details.

JUMP action - overview

Mellanox (MLX)

- Many tables
- No jumps to root
- No other restrictions found

DynaNIC (NFB)

DYNANIC

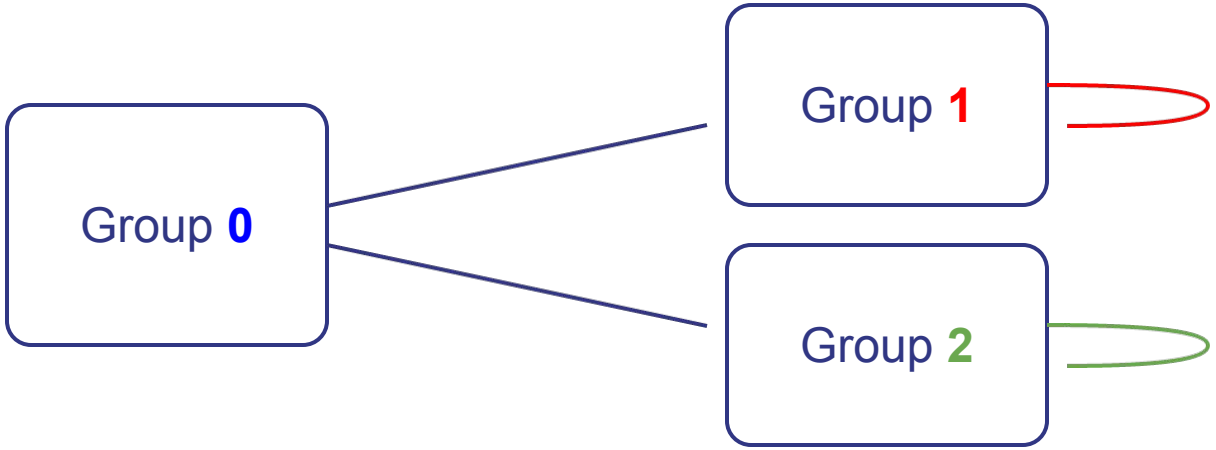
- Table count depends on design
- Trial version 4
- Only forward jumps
- Design restrictions found
- Can be overrun, no reason for now

Intel (ICE)

- No ordinary groups
- No jumps

JUMP action - MLX

```
flow create 0 group 0 pattern void / end actions jump group 1 / end
flow create 0 group 0 pattern void / end actions jump group 2 / end
flow create 0 group 1 pattern void / end actions jump group 1 / end
flow create 0 group 2 pattern void / end actions jump group 2 / end
```



JUMP action - MLX

```
flow create 0 group 1 pattern void / end actions jump group 1 / count identifier 1
flow create 0 group 2 pattern void / end actions jump group 2 / count identifier 1
```

```
testpmd> flow query 0 2 count

COUNT:

hits_set: 1
bytes_set: 1

hits: 0
bytes: 0
```

```
testpmd> flow query 0 3 count

COUNT:

hits_set: 1
bytes_set: 1

hits: 987283996
bytes: 126372351488
```

“Jumps” in ICE

FDIR Engine

```
flow create 0 ingress pattern eth / ipv4 dst is 198.168.1.0 / end actions queue index 1 / count identifier 1 / end
```

SWITCH Engine

```
flow create 0 ingress pattern eth / ipv4 dst is 198.168.1.0 / end actions queue index 2 / end
```

ACL Engine

```
flow create 0 ingress pattern eth / ipv4 dst spec 198.168.1.0 dst mask 255.255.255.0 / end actions queue index 3 / end
```

IPv4 Packet

dst: 198.168.1.0

Rules	Result	
	QUEUE	COUNT hit
FDIR + SWITCH + ACL	2	1
FDIR + ACL	1	1

VLANS

Parser & Patterns

- VLAN in the pattern
- Robustness of parser

Example Command:

```
flow create 0 ingress pattern ipv4 src is 192.168.1.1 / end actions queue index 5 / end
```

VLAN conclusion

Mellanox (MLX)

- One VLAN ID match
- All packets matched

DynaNIC (NFB)

DYNANIC

- Firmware dependent
- No VLAN ID match
- Five packets matched

Intel (ICE)

- Specific pattern needed to match VLAN ID
- One packet matched

Performance

Rule Insertion

- Speed
- Capacity

Throughput

With inserted rules

- 1024 packet size on 100G link speed

[1] https://dejankosticgithub.github.io/documents/publications/nicbench-pam21.pdf

Rule insertion - MLX

Capacity

- Root - 65k rules
- Asynchronous rules
- Non-root - 8M rules

Insertions speed

- Slowdown after 56k rules
  - hundreds instead of thousands
- Root - 90s
- Non-root - 30s

Rule insertion - DynaNIC

Capacity

- TCAM - 512 rules
- EM - 130k rules
- LPM - 130k rules

Insertions speed

- TCAM - 2ms but very small capacity
- EM - 700ms
- LPM - 117s

Rule insertion - ICE

Capacity

- HASH - One overwriting rule
- ACL - 64 rules
- SWITCH - 32k rules
- FDIR - 15k rules (5-tuples)
- SWITCH + FDIR - 48k rules

Insertions speed

- SWITCH - 17s
- FDIR - 116ms
- SWITCH + FDIR - 30s

Rule insertion - ICE

```
flow create 0 ingress group 0 pattern eth / ipv4 src is 0.0.41.235 dst is
0.0.41.235 proto is 17 / end actions queue index 11 / end
```

SWITCH

```
flow create 0 ingress group 0 pattern eth / ipv4 src is 0.0.41.236 dst is
0.0.41.236 proto is 17 / end actions queue index 12 / end
```

SWITCH

Rule insertion - ICE

```
flow create 0 ingress group 0 pattern eth / ipv4 src is 0.0.41.235 dst is
0.0.41.235 proto is 17 / end actions queue index 11 / end
```

SWITCH

```
flow create 0 ingress group 0 pattern eth / ipv4 src is 0.0.41.236 dst is
0.0.41.236 proto is 17 / end actions queue index 12 / end
```

FDIR

Throughput

Mellanox (MLX)

- Root table 57k rules 3Gbps
- Non root millions rules
- 50 jumps 88G

DynaNIC (NFB)

DYNANIC

- No drop detected
- Special design with more tables for jump test

Intel (ICE)

- No drop detected

[1] https://dejankosticgithub.github.io/documents/publications/nicbench-pam21.pdf

Conclusion

MLX

+ Many patterns and actions
+ Predictable rule behavior
− Performance influenced by number of rules

DYNANIC + NFB

+ Easily customizable
+ Performance invariant to rule count
− Low number of patterns
(User specific)

ICE

+ Performance invariant to rule count
− Engines readability
− Available patterns from code


DPDK
SUMMIT
Powering the Future of Networking Software
Try DynaNIC now:
https://dyna-nic.com/try-now/
DYNANIC
Findings in more detail:
https://dyna-nic.com/rte-flow-nic-platform-comparison-dpdk-2026/
DYNANIC