A new solution to support connection tracking

Yuan Linsi

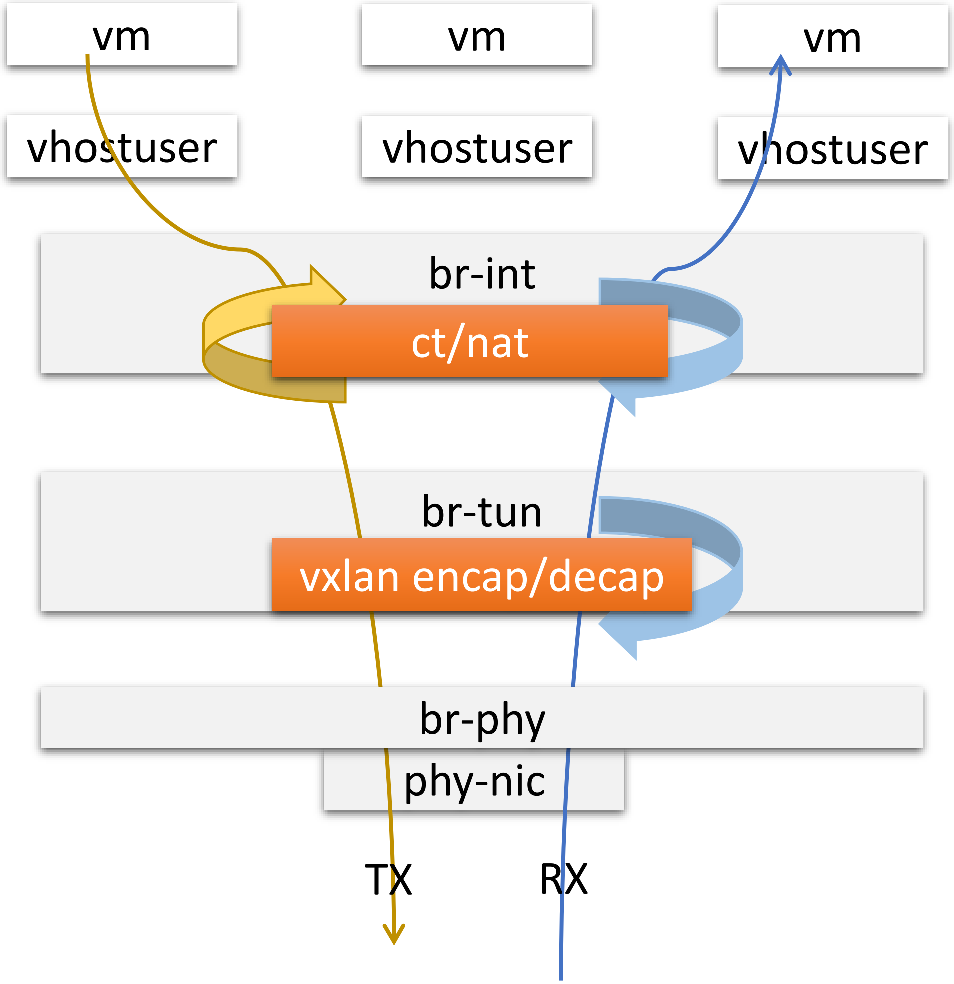
Jaguar Microsystems

Agenda

Open vSwitch

- Community Solution: ct action
- Challenge
- Requirement and Design Goals
- Our Proposed Solution
    - Design
    - Performance
    - Impact on hwol support
- Further work

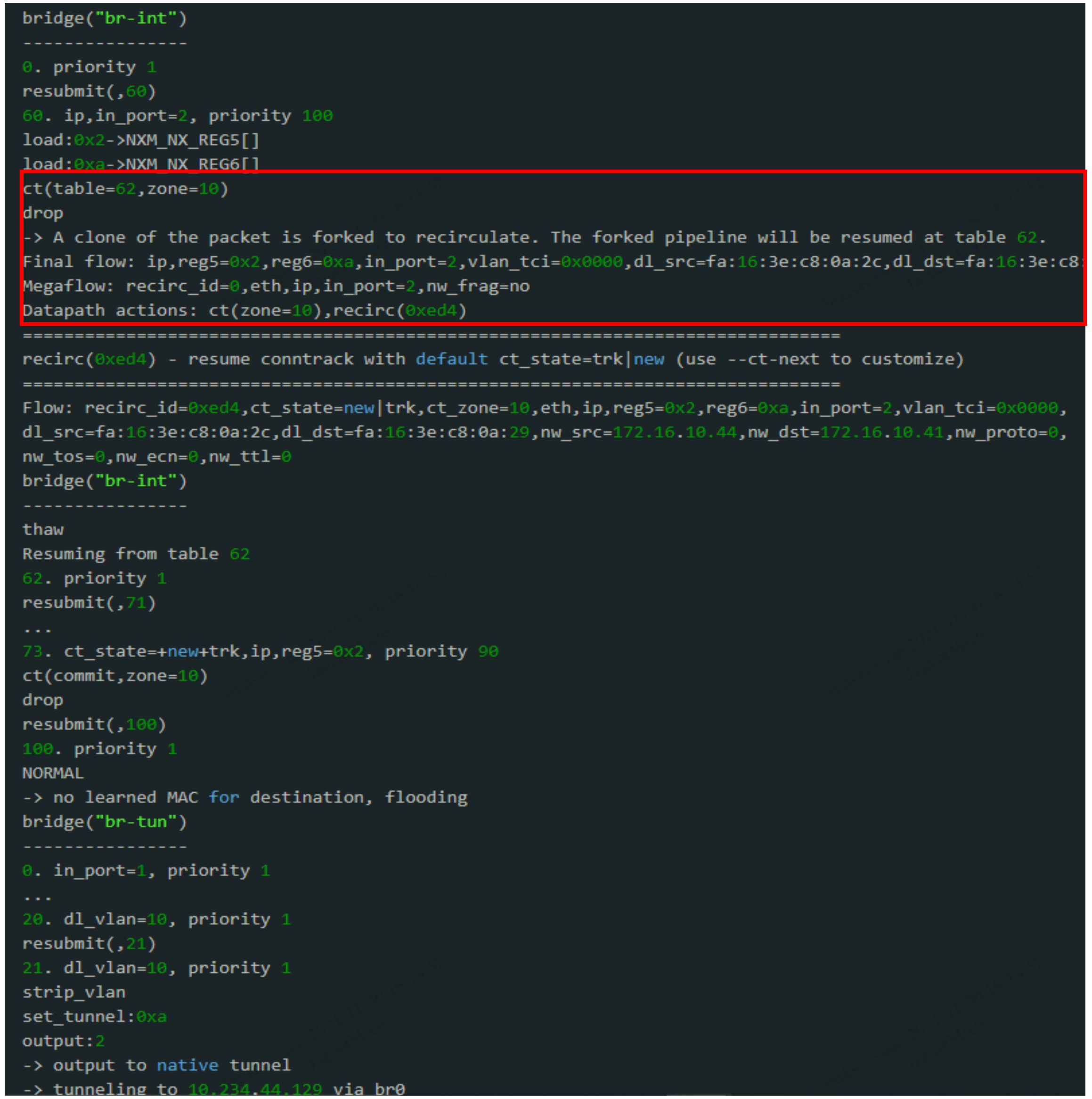
Community Solution: ct action



TX Recirc once: ct update conn state

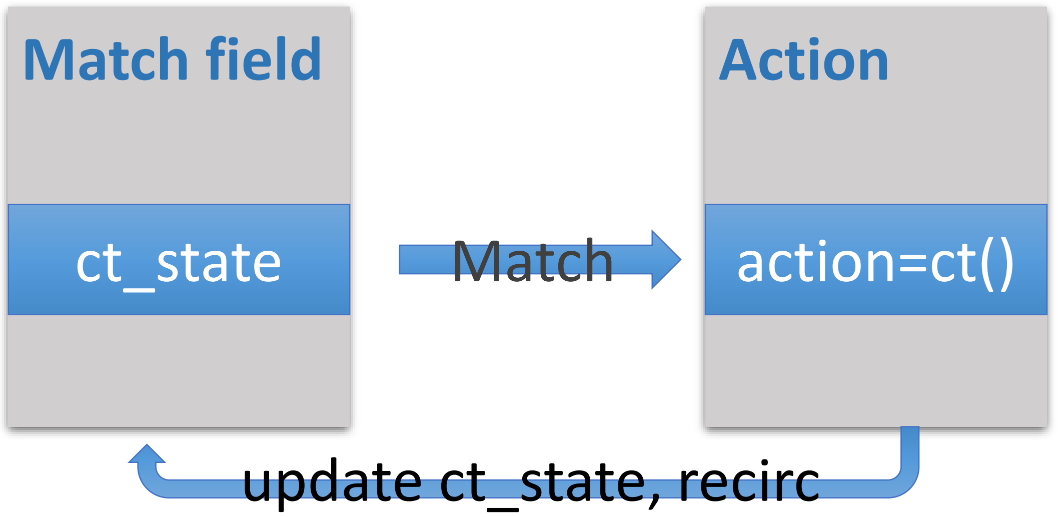
RX Recirc twice: ct update conn state; vxlan tunnel decap

ofproto pkt trace result for tx:



```
bridge("br-int")
----------------
0. priority 1
resubmit(,60)
60. ip,in_port=2, priority 100
load:0x2->NXM_NX_REG5[]
load:0xa->NXM_NX_REG6[]
ct(table=62,zone=10)
drop
-> A clone of the packet is forked to recirculate. The forked pipeline will be resumed at table 62.
Final flow: ip,reg5=0x2,reg6=0xa,in_port=2,vlan_tci=0x0000,dl_src=fa:16:3e:c8:0a:2c,dl_dst=fa:16:3e:c8
Megaflow: recirc_id=0,eth,ip,in_port=2,nw_frag=no
Datapath actions: ct(zone=10),recirc(0xed4)
===============================================================================
recirc(0xed4) - resume conntrack with default ct_state=trk|new (use --ct-next to customize)
===============================================================================
Flow: recirc_id=0xed4,ct_state=new|trk,ct_zone=10,eth,ip,reg5=0x2,reg6=0xa,in_port=2,vlan_tci=0x0000,
dl_src=fa:16:3e:c8:0a:2c,dl_dst=fa:16:3e:c8:0a:29,nw_src=172.16.10.44,nw_dst=172.16.10.41,nw_proto=0,
nw_tos=0,nw_ecn=0,nw_ttl=0
bridge("br-int")
----------------
thaw
Resuming from table 62
62. priority 1
resubmit(,71)
...
73. ct_state=+new+trk,ip,reg5=0x2, priority 90
ct(commit,zone=10)
drop
resubmit(,100)
100. priority 1
NORMAL
-> no learned MAC for destination, flooding
bridge("br-tun")
----------------
0. in_port=1, priority 1
...
20. dl_vlan=10, priority 1
resubmit(,21)
21. dl_vlan=10, priority 1
strip_vlan
set_tunnel:0xa
output:2
-> output to native tunnel
-> tunneling to 10.234.44.129 via br0
```

Community Solution: ct action



Match field

ct_state

Match

Action

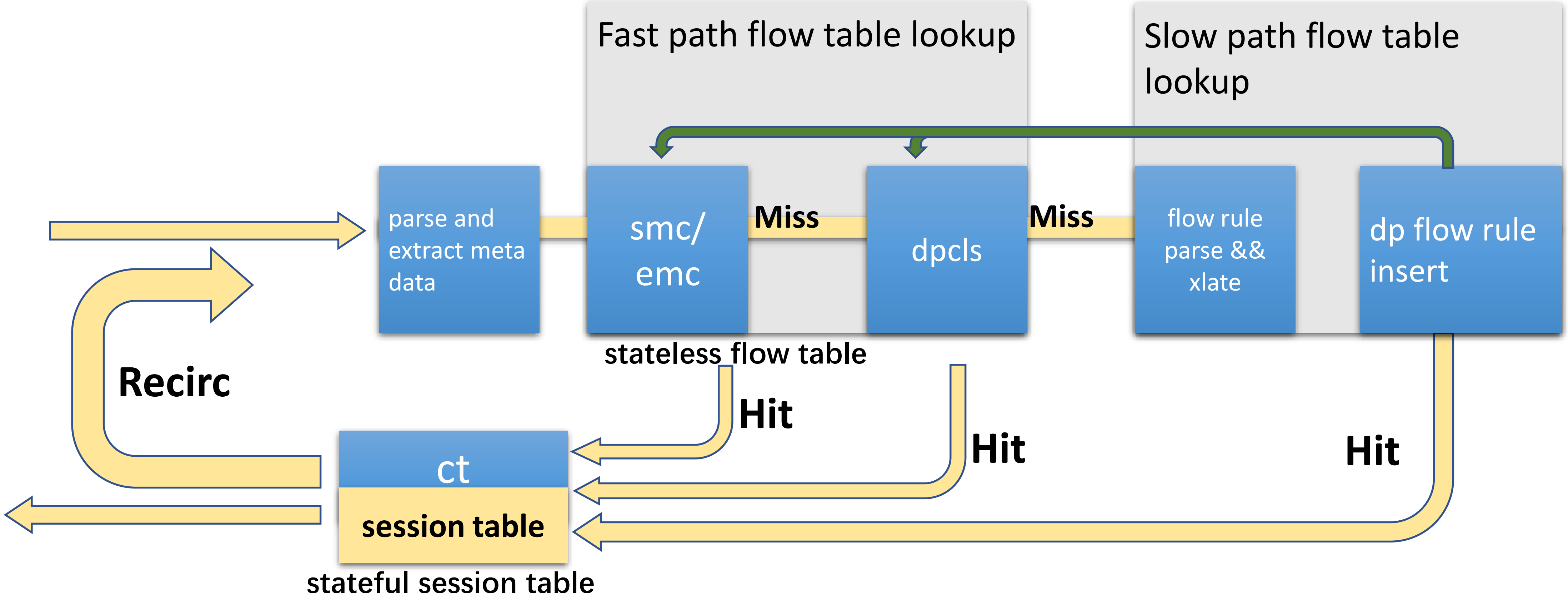
action=ct()

update ct_state, recirc

1. ct_state: -trk	action:ct	ct_state:+new+trk
2. ct_state: +new+trk	action:output	ct_state:+new+trk

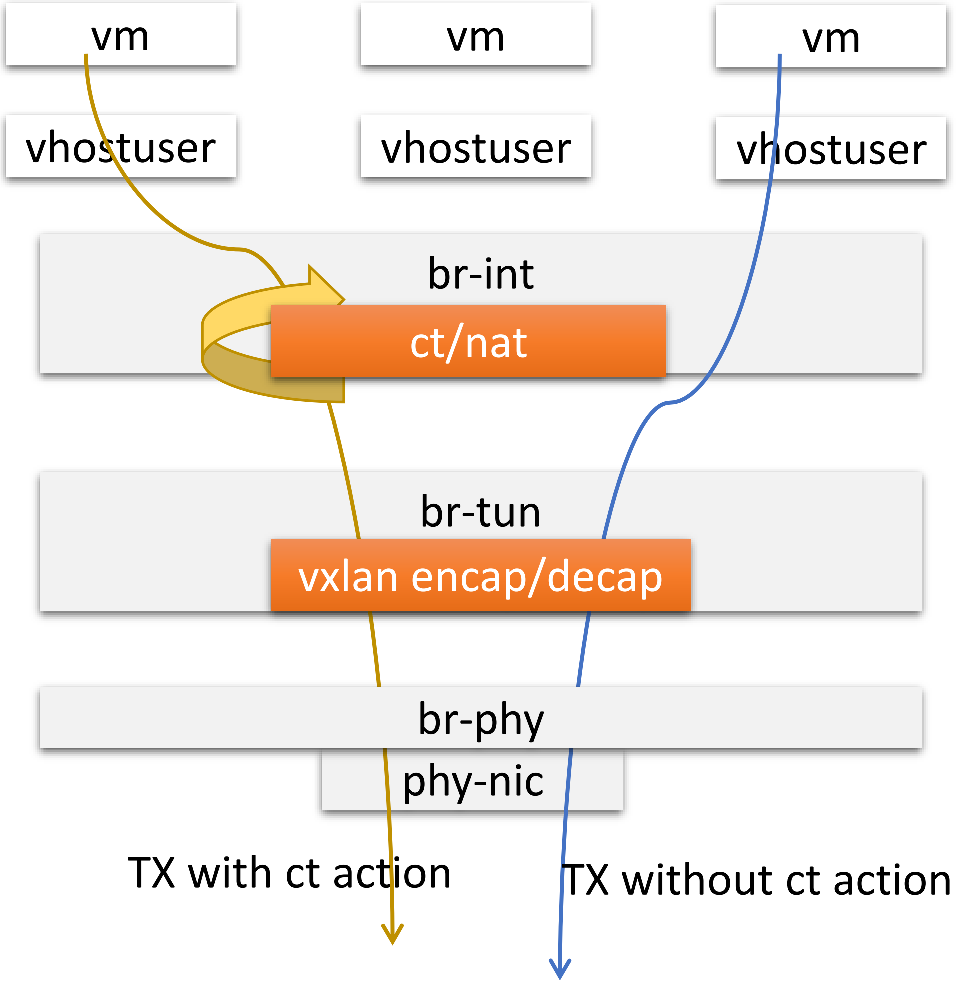
- update the conn state in ct action and recirc back to match based on the new conn state

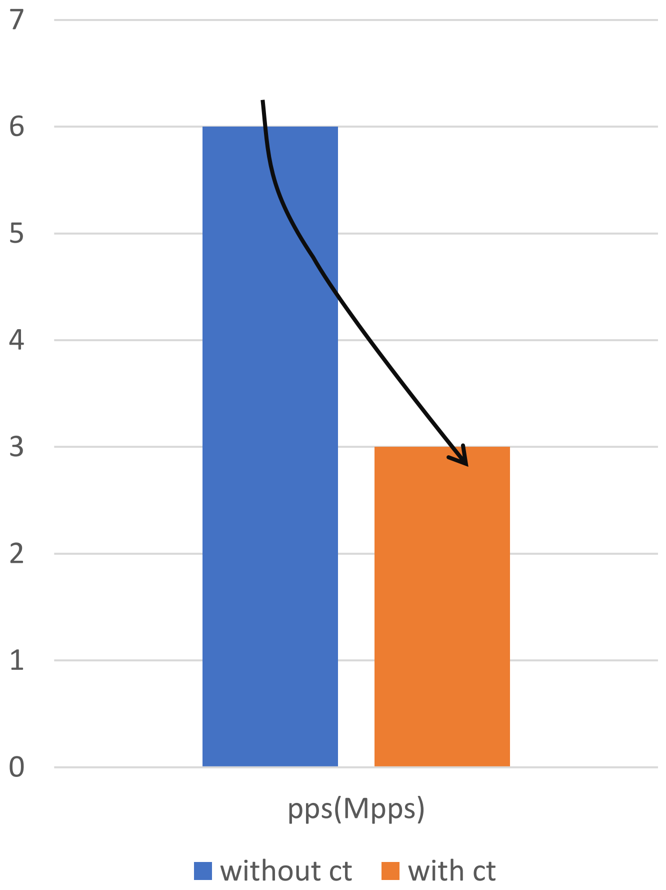
Community Solution: ct action



- The whole design is Flow-oriented, stateless
- rely on recirc to support conn track

Challenge1: performance degradation





~50% performance degradation

cpu: skylake 6148, turbo on, 3.1Ghz

ovs: 4pmd (2core,4ht), base ovs 2.9

numa： keep the vm running in the same numa node with pmd thread

test case: vm to vm, running with netperf to generate the traffic

Challenge2: hard to support hardware offload

- hard to support hardware offlaod base ct action:

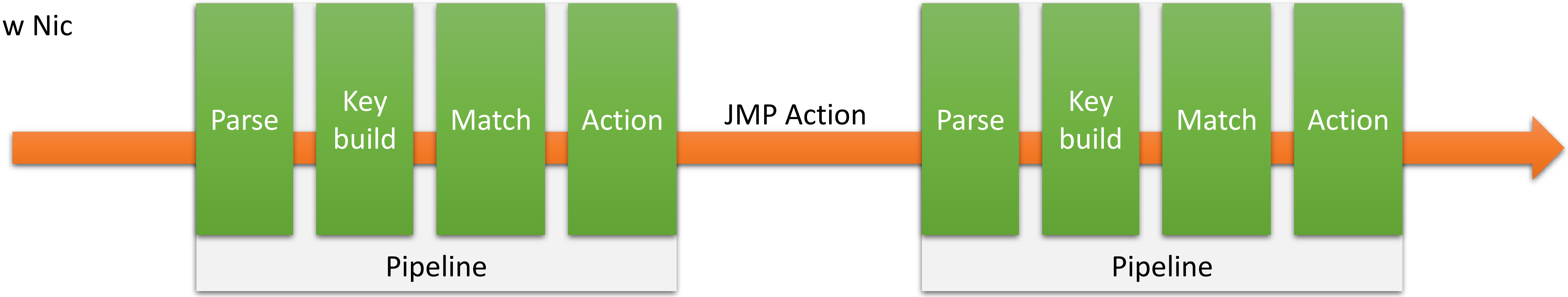
  Require extra hw resource to support multi table to guarantee the hwol performance will not degrade

ovs dp

Match	Action
recirc_id(0),in_port(6),eth(),ipv4()...	actions:ct(zone=10),recirc(0xf0a)

Match	Action
ct_state(new+trk),recirc_id(0xf0a),eth(),ipv4()	actions:clone(tnl_push(...)),out_port(3)),4)

hw Nic

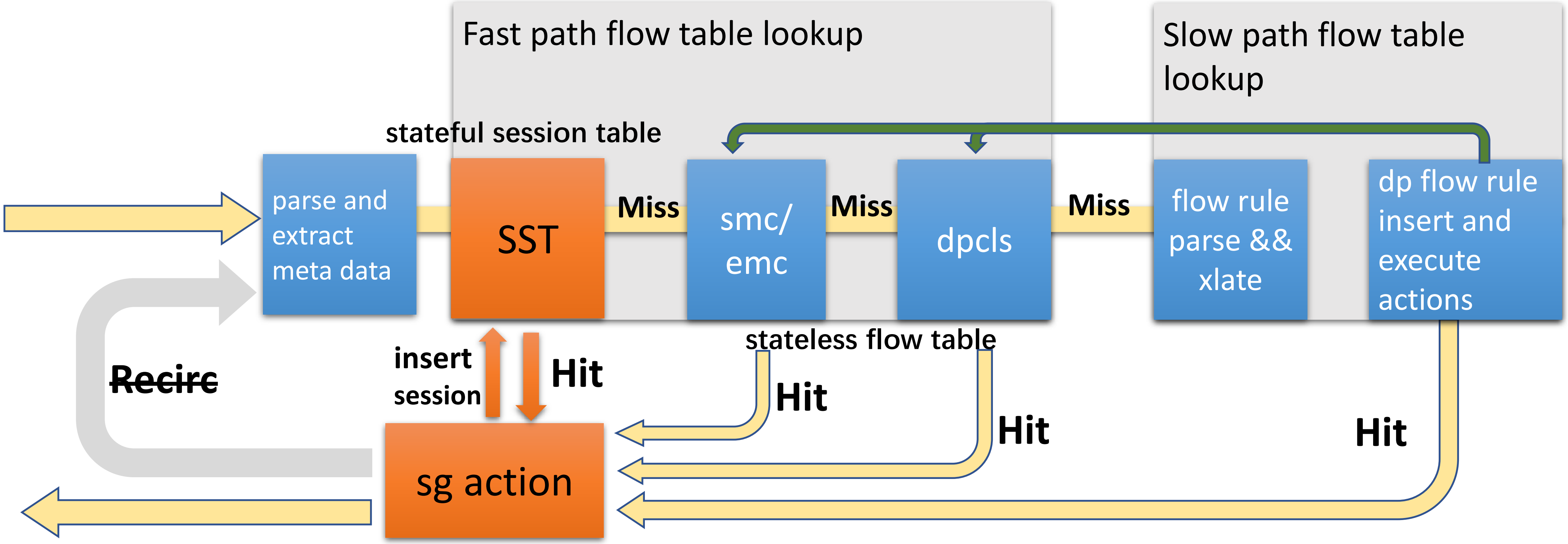


Requirement and Design Goals

- Goals:
    - To get Higher performance in software datapath
    - Simplify the hardware offload support
- Requirement:
    - keep control plane programmable
        - up to the control plane to decide whether or not to enable the new logic
    - modular design
        - Support fallback to the old one

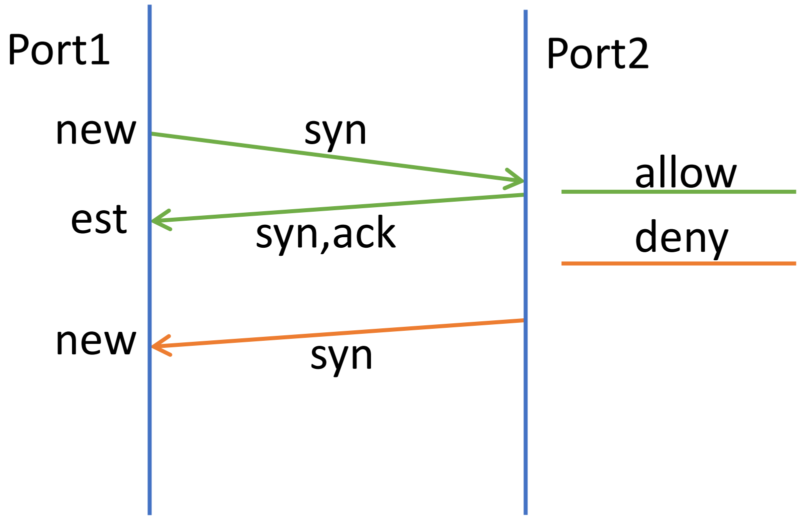
New design overview

- Introduce a new action： **sg**(allow|deny,[nat()],[alg()])
- Introduce a new fastpath cache： **SST** (Stateful Session Table Cache)
- The whole design is Connection oriented



Design of sg action

sg police：



base ct action:

```
table=0,priority=100,ip,ct_state=-trk,action=ct(table=1)
table=1,in_port=1,ip,ct_state=+trk+new,action=ct(commit),2
table=1,in_port=1,ip,ct_state=+trk+est,action=2
table=1,in_port=2,ip,ct_state=+trk+new,action=drop
table=1,in_port=2,ip,ct_state=+trk+est,action=1
```

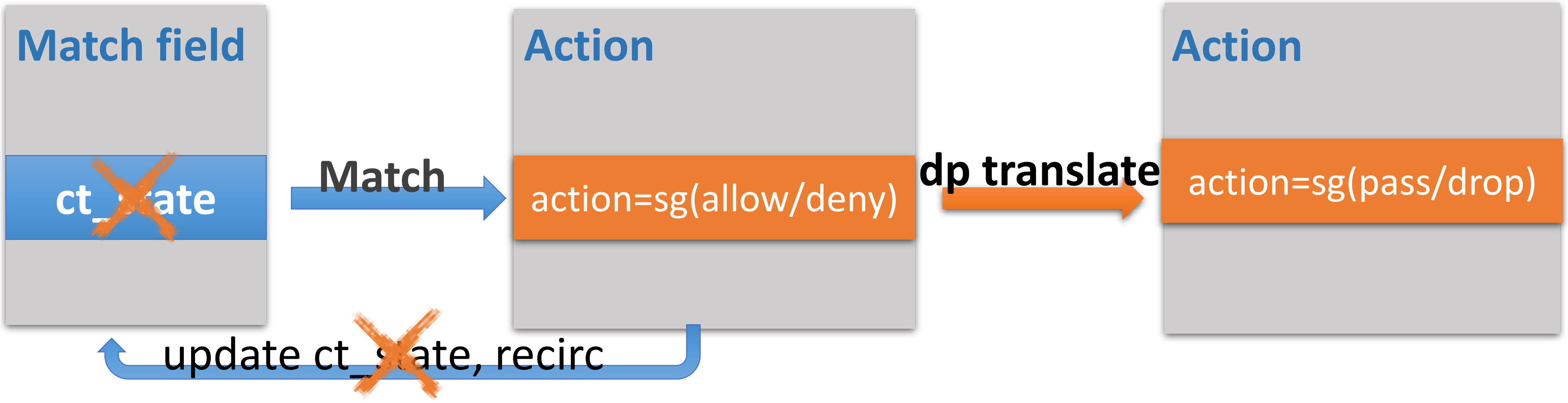
base sg action:

```
table=0,in_port=1,ip, action=sg(allow),2
table=0,in_port=2,ip, action=sg(deny),1
```

- Solution：
  - Control plane no more need to specify the conn state in the match field
  - Data plane will do a secondary translate base on the conn state and get final action

Design of sg action

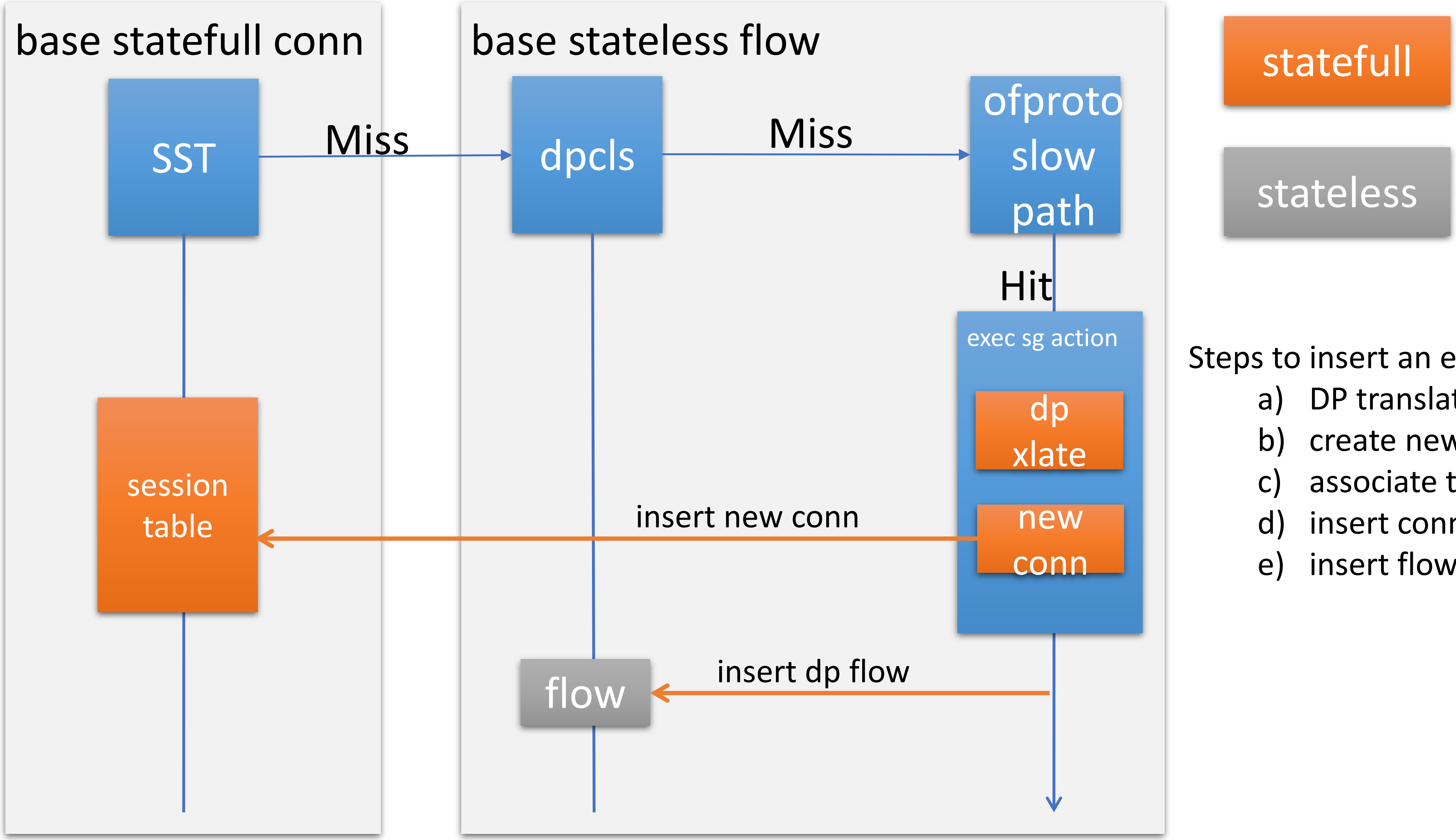
Introduce a new action： sg(allow|deny,[nat()])



1. new conn	action:sg(allow)	pass
2. new conn	action:sg(deny)	drop
3. conn exist	action:sg(deny)	pass

- Solution：
  - Control plane no more need to specify the conn state in the match field
  - Data plane will do a secondary translate based on the conn state and get final action

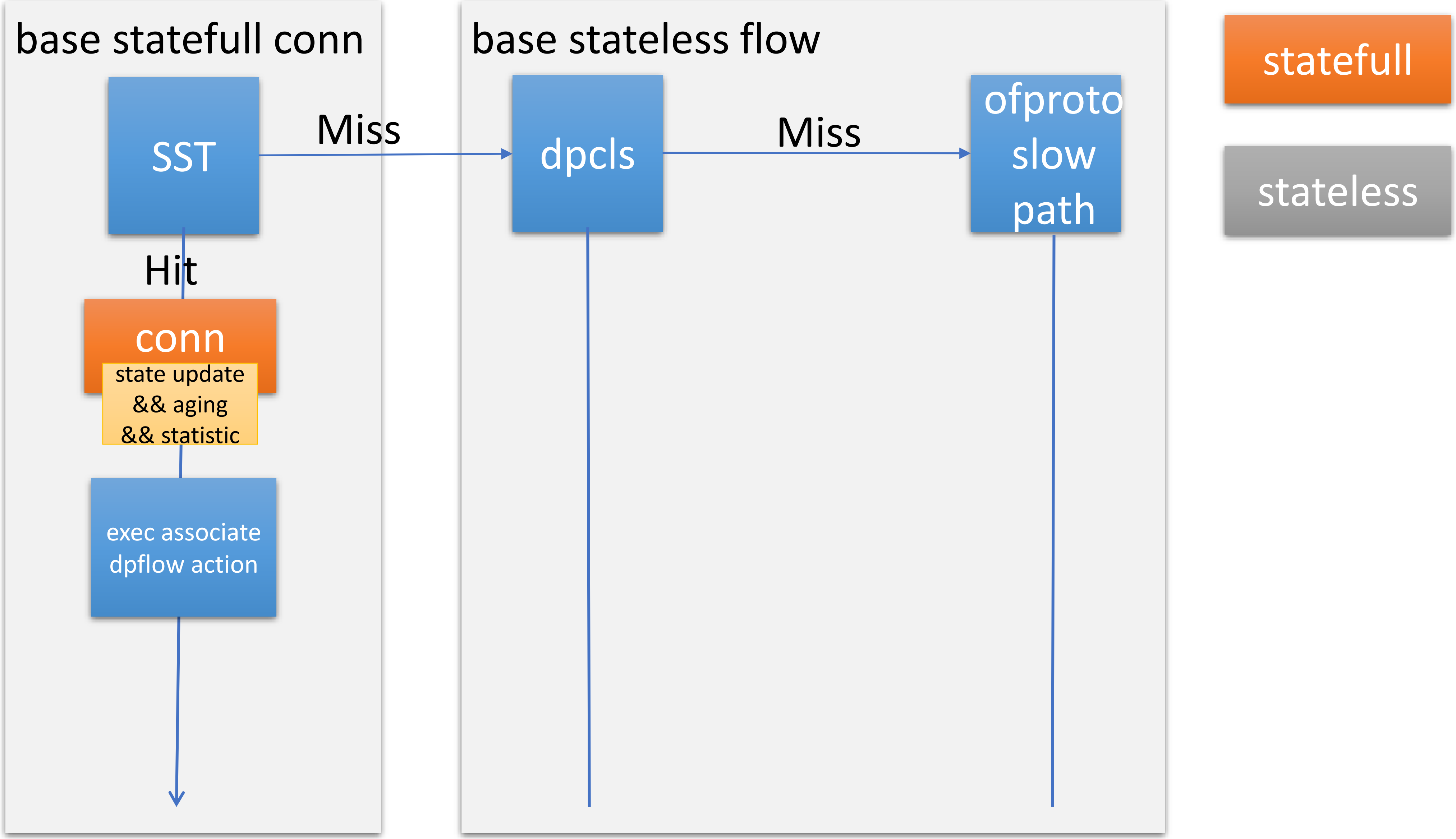
Introduction of SST (Stateful Session Table)



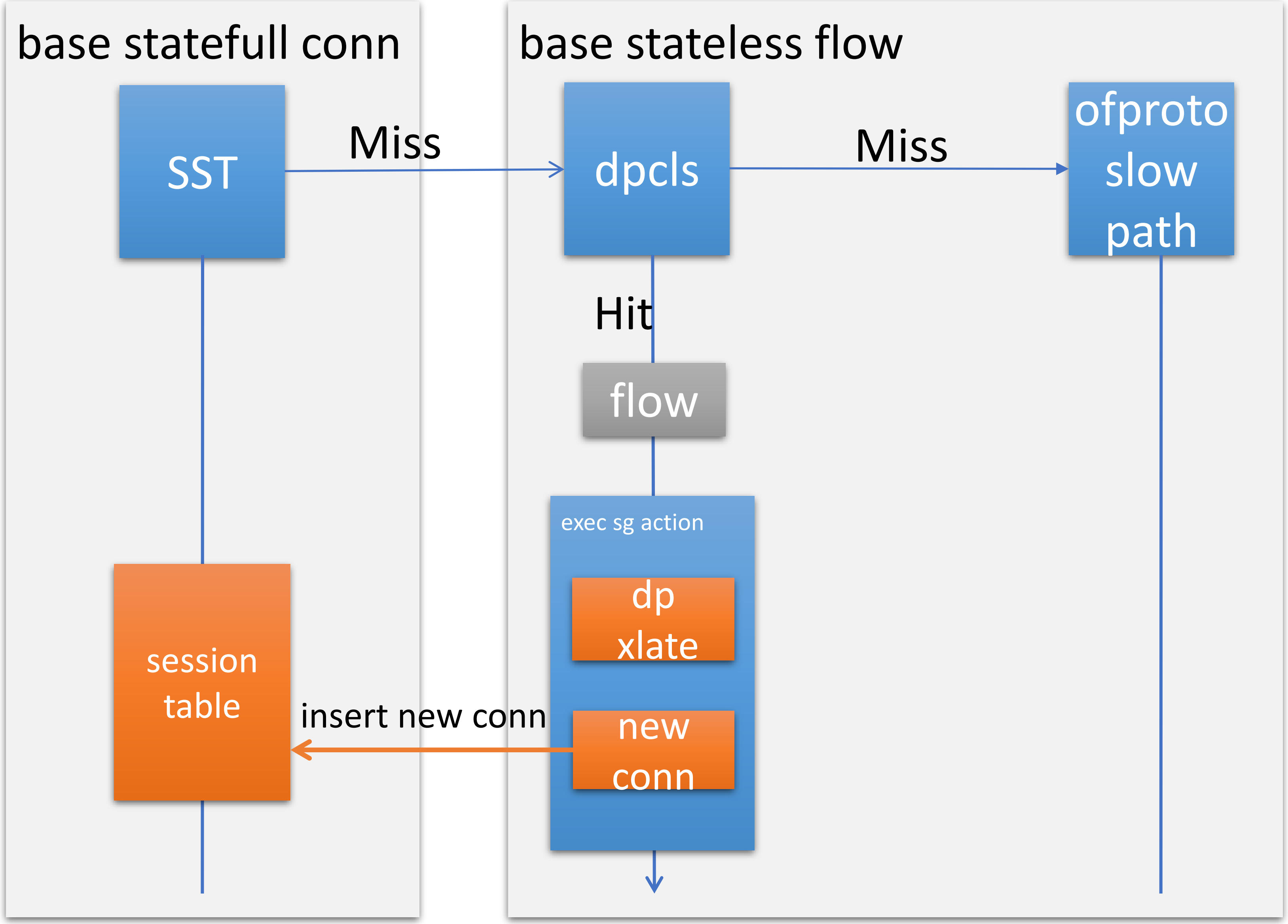
Steps to insert an entry to SST:

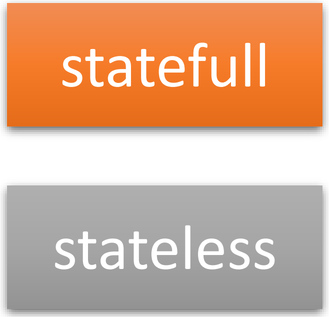
a) DP translate in sg action
b) create new conn
c) associate the conn with the flow
d) insert conn to SST
e) insert flow to the dpcls and EMC

Introduction of SST (Stateful Session Table)



Introduction of SST (Stateful Session Table)





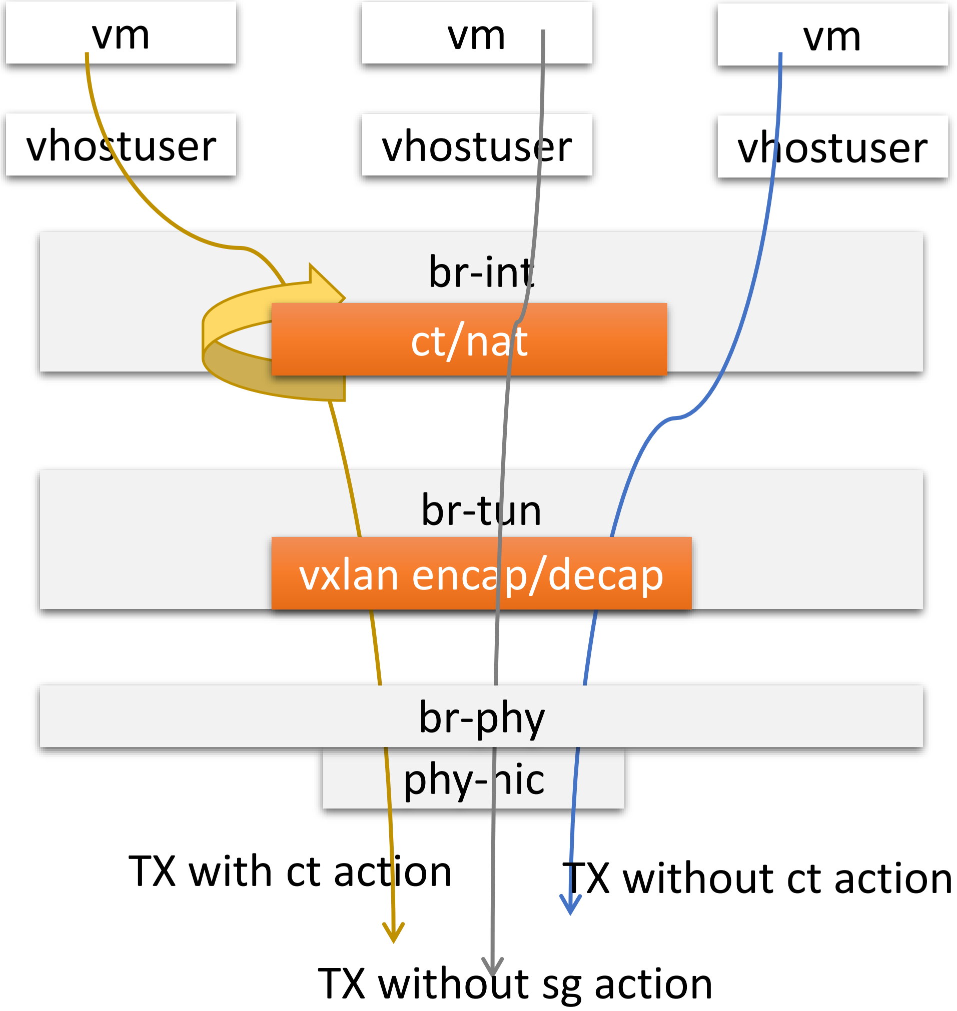
- dpcls works as L2 cache
- avoid too much upcall to the ofproto slow path

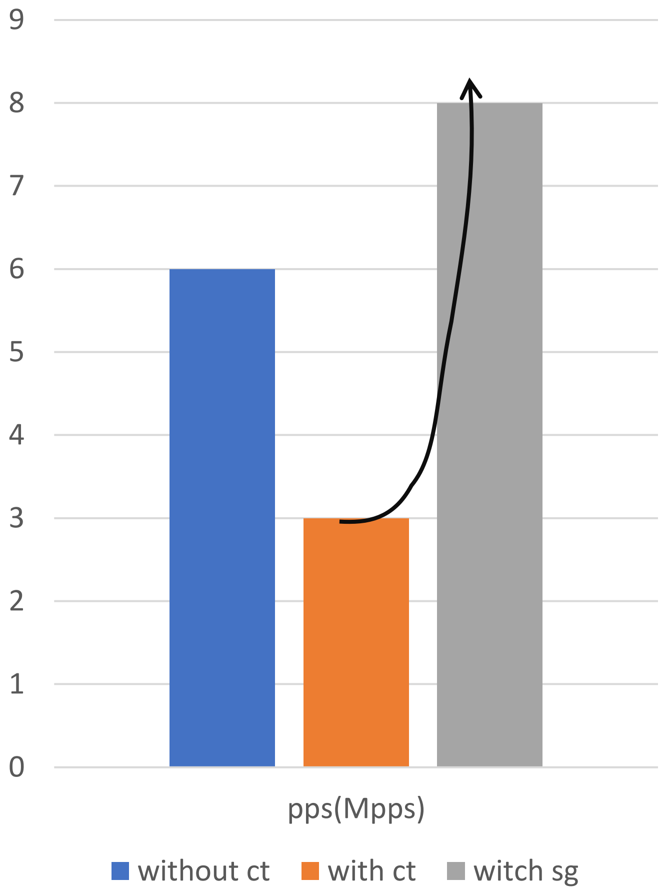
More optimizations

Base on the new design, more optimizations can be done now:

- Unify vxlan decap dp flow
  - No more rely on recirc to finish the vxlan decap
- Merge actions
  - Since the dp flow has been unified, no more rely on recirc to support ct, we can merge the actions to simplify the action execute
- Use spin lock instead of mutex
  - Avoid context switch, and spin lock is much more lightweight
- Use hugepage based session memory pool to support session entry allocation
  - Eliminate the TLB miss events and speed up the allocation and free for each session entry
- batch execute tnl_push and tnl_pop

Performance





~260% performance improvement

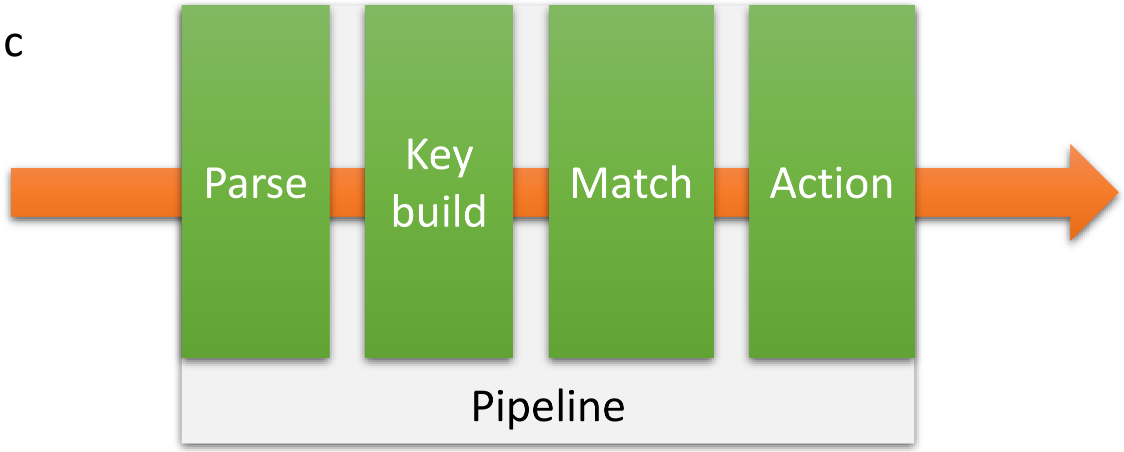
cpu: skylake 6148, turbo on, 3.1Ghz
ovs: 4pmd (2core,4ht), base ovs 2.9
numa: keep the vm running in the same numa node with pmd thread
test case: vm to vm, running with netperf to generate the traffic

Impact on hwol support

hardware offlaod support base on sg action:

	Match	Action
ovs dp	`recirc_id(0),in_port(6),eth(),ipv4()...`	`actions:`**`sg(),tnl_push(), output 3`**

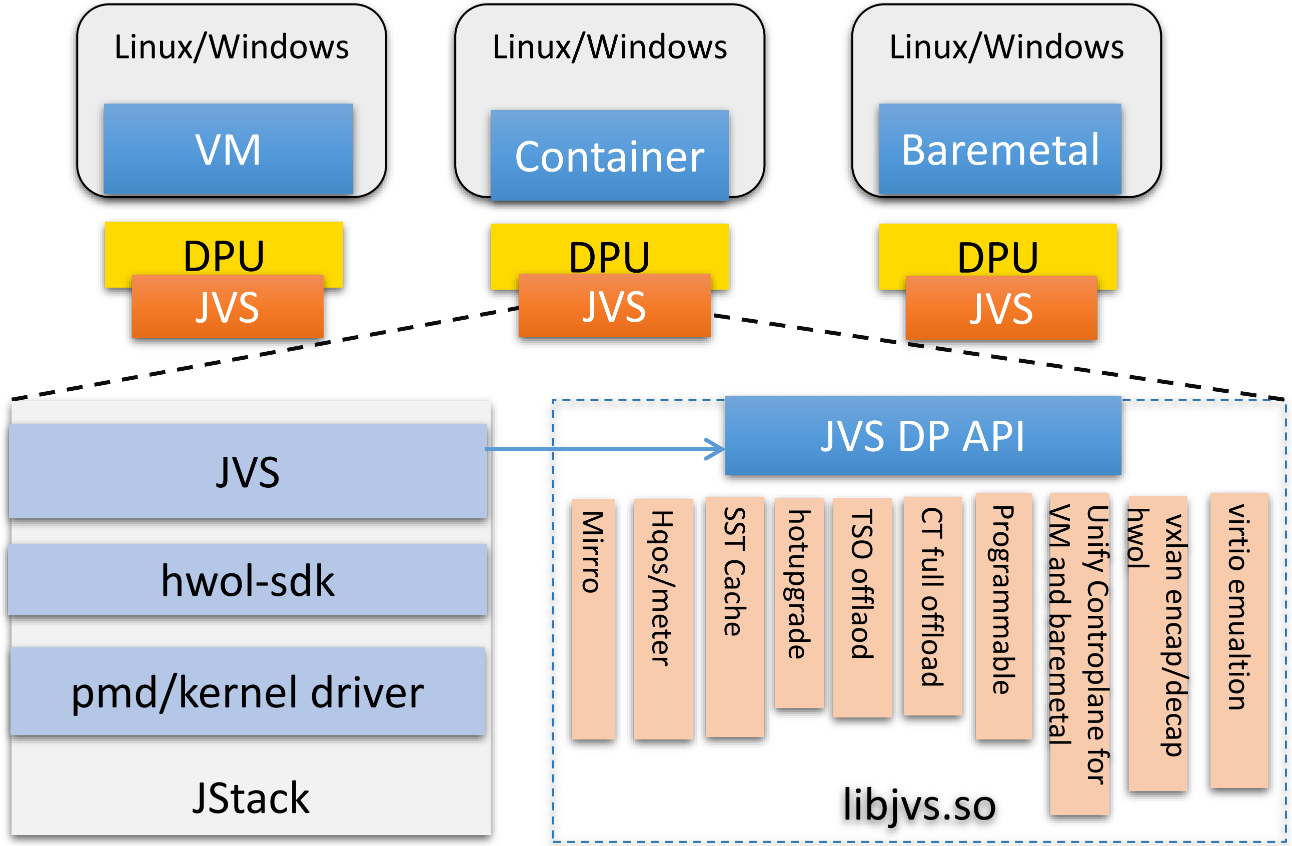
hw Nic



Unify multi flow in ovs layer, no extra work need to do to offload to the HW Nic, much more easy to offload to the hw Nic.

Acknowledgement

JVS: Jaguar Virtual Switch



support running on different archs:

CPU:

x86_64 / AMD / ARM

Linux distribution:

Not limited to a specific one

- Thanks to:
  - Wang Yao, Baidu
  - Mao Yingming, Baidu
  - Liu Feifei, Baidu
- Any comments on this design are welcome. Feel free to contact me via email
- Contact:
  - lindsay.yuan@jaguarmicro.com

THANKS FOR WATCHING

Contact : lindsay.yuan@jaguarmicro.com