The State of Stateful Services

Joe Stringer, Jarno Rajahalme
{joe,jarno}@ovn.org

Agenda

- Connection Tracking
- Firewalling
- NAT
- Other stateful services
- Summary

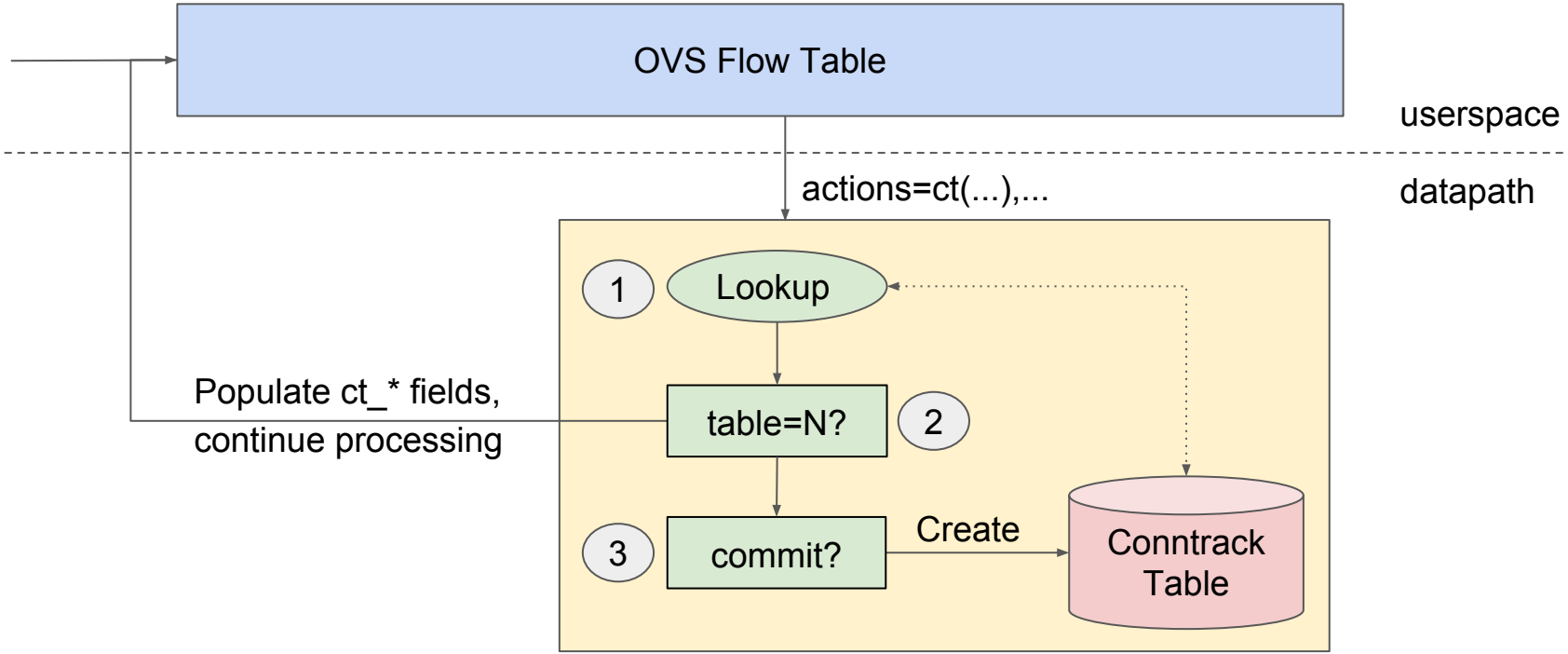
Motivation

- OVN heating up
  - OpenStack
  - Kubernetes
- Expanding feature set
  - Firewalls
  - NAT
  - Load Balancing

Connection Tracking

- Track connections
  - Per-connection state stored in datapath
  - Expose concepts like “new connection”
- Microflow steering without matching every microflow
  - Avoid upcall when possible
- Leverage existing work
- Foundation for a variety of stateful services

Connection Tracking



Example firewall

Table	Match	Action
0	priority=100,in_port=1,ip	ct(commit),2
0	priority=100,in_port=2,ip,**ct_state=-trk**	ct(table=1)
0	priority=10,arp	normal
0	priority=1	drop
1	priority=100,in_port=2,ip,**ct_state=+est**	1
1	priority=1	drop

Packet & connection states

- Packets are untracked initially*, become tracked via ct()
- Tracked (**trk**) packets may be..
  - Part of a **new** or **est**ablished connection
  - Reply (**rpl**): Connection must be established
  - **Rel**ated: Related to an established connection
  - **Inv**alid

* Exception: internal ports in current namespace may inherit state from local network stack

Conntrack match fields

- ct_state
- ct_zone
  - Logically separate connection tracking table
  - Multi-tenancy
- ct_mark
  - Attach 32 bits of metadata to particular connections
- ct_label
  - Similar to mark, 128 bits

Conntrack action

- Transparently reassemble IP fragments (re-fragment on output)
- No args: Let the connection tracker know, ignore its results.
- zone=N: Track in logical zone N
- alg=ftp: Apply protocol-specific tracking, eg FTP detect data connections
- exec(..): Additional actions in connection tracking context
  - set_field(...->ct_mark); set_field(...->ct_label)
  - Changes matchable only on recirculated packets.
- table=N: Clone packet to send to connection tracker. When the connection tracker is finished, resume processing in table N for that packet. The original packet continues right after the ct(...) action.
- commit: Persist state about this connection

NAT & Load Balancing

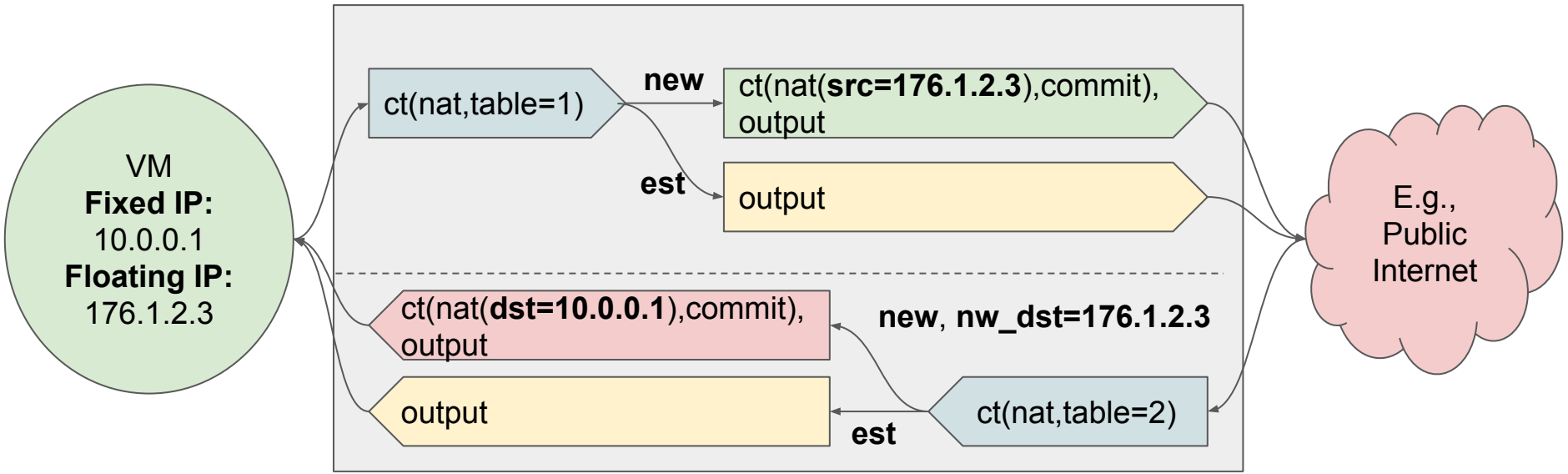
Network Address Translation Use Cases

- OpenStack allows a persistent *Floating IP* to be assigned for a VM in addition to dynamically allocated *Fixed IP*
  - Both Source NAT (**SNAT**) and Destination NAT (**DNAT**) needed to map between these
- Kubernetes Services hide servers behind a *Virtual IP addresses*
  - *Load balancer* chooses the server for each connection
  - DNAT to map the virtual IP to the chosen server's IP address
- The corresponding transport port can also be mapped
  - Without an explicit port (range) the port is mapped only in case of a collision

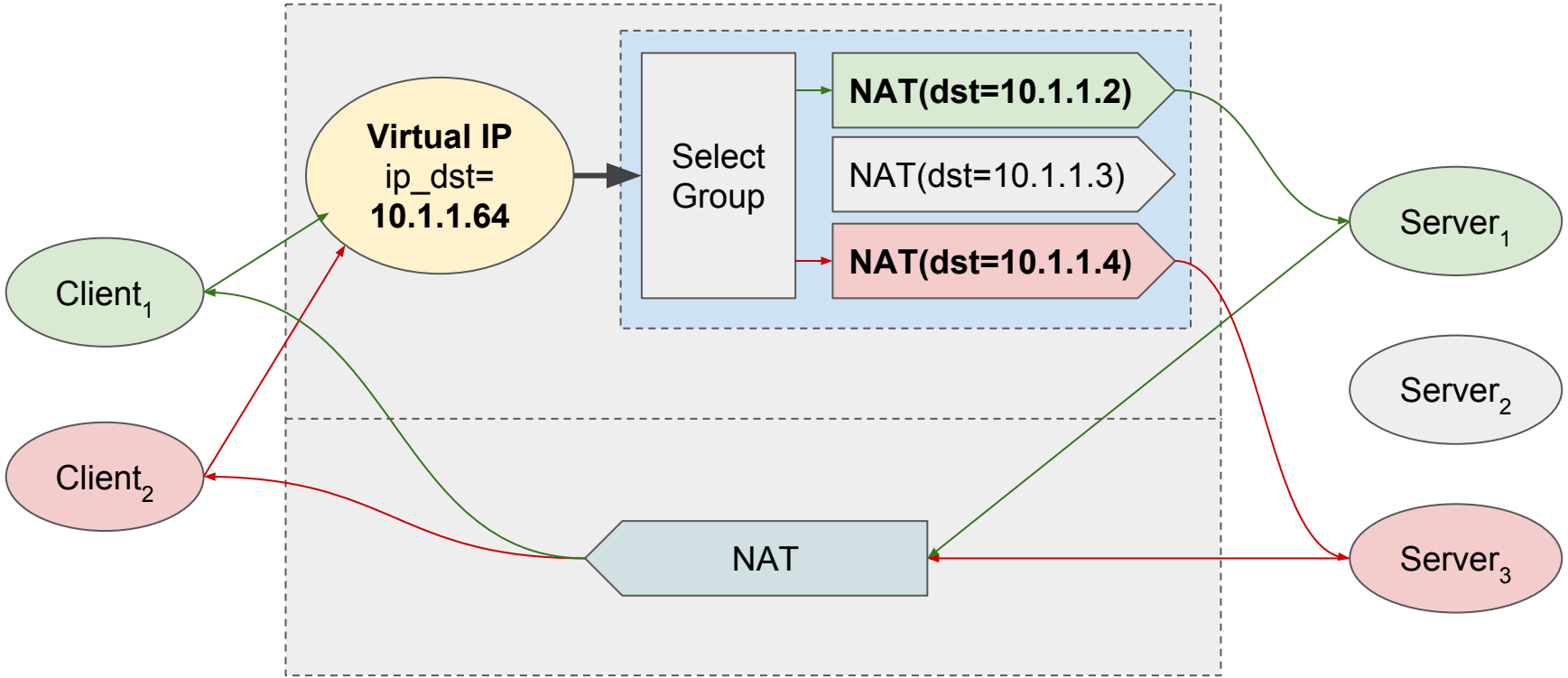
NAT Action Extends The CT Action

- Always executes in the context of the current connection
  - `CT(..., NAT(...), ...)`
  - Typically NAT can be added to CT actions already used for ACLs.
- **New connections** need a source or destination address (range) and optionally a port (range) + a CT `commit` and possibly the `zone` argument
  - `ct(`**`commit`**`,nat(`**`src=10.0.0.240`**`),`**`alg=ftp`**`)`
  - `ct(commit,`**`zone=1`**`,nat(`**`src=10.0.0.240:32768-65535`**`,random))`
  - `ct(commit,nat(`**`dst=10.0.0.128-10.0.0.254`**`,hash))`
  - `ct(commit,nat(`**`dst=10.0.0.240-10.0.0.254:32768-65535`**`,persistent))`
- NAT **without arguments** only NATs committed, established, or related uncommitted connections

NAT for OpenStack Floating IPs



DNAT Load Balancing



DNAT Load Balancing (cont.)

- Controller needs to balance traffic by (re-)specifying group weights
  - Based on server feedback or group stats
- Bucket selection currently happens on Ethernet + 5-tuple hash
  - recirc_id(0),in_port(2),eth(**src=80:88:88:88:88:11**,dst=80:88:88:88:88:88),eth_type(0x0800), ipv4(**src=10.1.1.1**,dst=10.1.1.64,proto=6,frag=no),tcp(**src=60754**,dst=80), … , actions:ct (commit,nat(dst=10.1.1.4)),recirc(0x1)
- Every connection goes to userspace as a miss upcall
- More work needed to avoid unnecessary upcalls

Connection Tracking Status

- Conntrack kernel patches merged and part of Linux-4.3
- Open vSwitch conntrack patches:
  - Userspace (ofproto) support in master
  - System-traffic testsuite in master
  - Kernel datapath backport under review
  - DPDK/Userspace datapath series posted
- NAT: RFC series posted on net-next and ovs-dev
  - Non-RFC when net-next window opens
  - DPDK/Userspace datapath future work
- Load-balancing: Investigation phase
  - Plausible with NAT functionality
  - May need further extension for a full implementation

Q&A