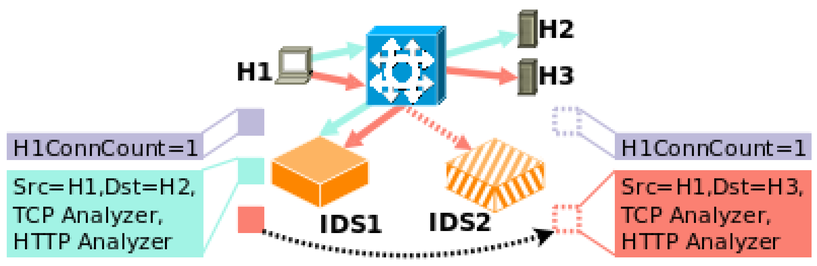
OpenNF
Enabling Innovation in Network Function Control

Aaron Gember, Raajay Viswanathan, Chaithan Prakash, Robert Grandl, Junaid Khalid, Sourav Das, Aditya Akella
{agember,raajay,cprakash,rgrandl,junaid,souravd,akella}@cs.wisc.edu

Motivation

Network functions virtualization (NFV) and Software Defined Networking (SDN) together have the potential to help operators achieve three important goals: (1) offer and satisfy tight SLAs; (2) accurately monitor and manipulate network traffic; (3) minimize operating expenses. However, operators need additional control mechanisms to satisfy these goals when packet processing must be redistributed across a collection of NF instances: e.g., to realize elastic NF scaling, rapid NF upgrades, and selective invocation of advanced remote processing. If any flow can quickly and safely be reallocated to any NF instance at any time, then operators can optimally satisfy all three goals.



An example scenario requiring scale out and load re-balancing to ensure all traffic is analyzed and operating expenses are minimized.

Download the source code or read the paper *http://opennf.cs.wisc.edu*

Potential Solutions

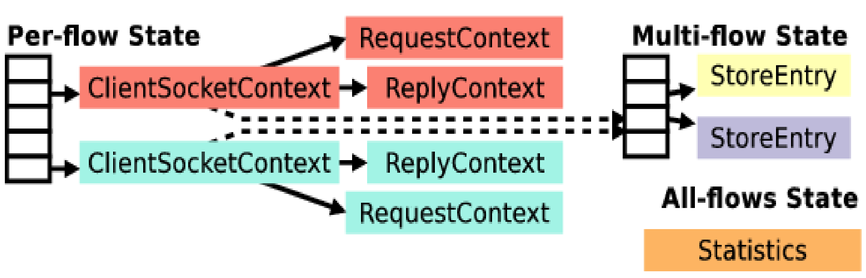
Virtual machine replication – clones more state than necessary, possibly leading to incorrect NF behavior; does not support merging

Vendor-provided controller – vendors can transfer state between NFs based on detailed knowledge of NF internals, but the controller's state decisions may conflict with network-wide objectives

Application-level library [Split/Merge] – NFs call the library to allocate, free, and access state, and a controller calls the library to import/export state; limited scope, does not deal with race conditions or provide guarantees.

NF State Taxonomy

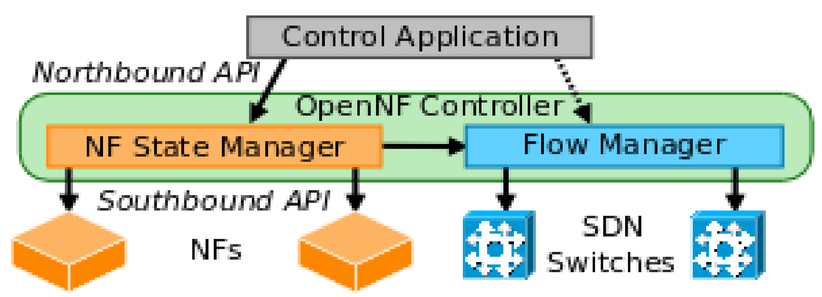
We classify NF state based on scope, or how many flows an NF-created piece of state applies to—one flow (per-flow), multiple flows (multi-flow), or all flows (all-flow).



NF state taxonomy applied to the Squid caching proxy

Our taxonomy provides a natural way for reasoning about how a control application should move/copy/share state.

OpenNF Architecture



- Control Applications dictate the behavior and performance of NFs using the northbound API
- NF State Manager enforces them by managing the distribution of NF states across instances using the southbound API
- Flow Manager is used to direct traffic in accordance with the distributed state

Northbound API

```
move(srcInst,dstInst,filter,scope,properties)
copy(srcInst,dstInst,filter,scope)
void share(list<inst>,filter,scope,consistency)
void notify(filter,inst,enable,callback)
```

Guarantees Supported:

Move: *No_Guarantee, Loss Free, Loss Free + Order Preserving*

Copy: *Eventual Consistency*

Share: *Strong or Strict Consistency*

Implemented live migration and scaling control applications on top of northbound API

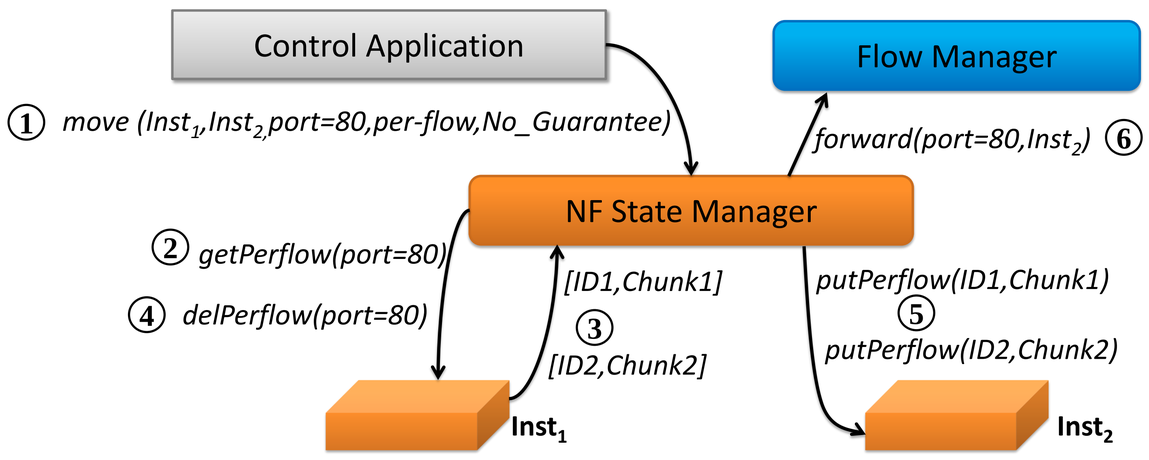
Southbound API

```
multimap<flowid,chunk> getPerflow(filter)
void putPerflow(multimap<flowid,chunk>)
void delPerflow(list<flowid>)
multimap<flowid,chunk> getMultiflow(filter)
void putMultiflow(multimap<flowid,chunk>)
void delMultiflow(list<flowid>)
list<chunk> getAllflows()
void putAllflows(list<chunk>)
void enableEvents(filter,action)
void disableEvents(filter)
```

Event Actions: *Process_packet, Buffer_packet, Drop_packet*

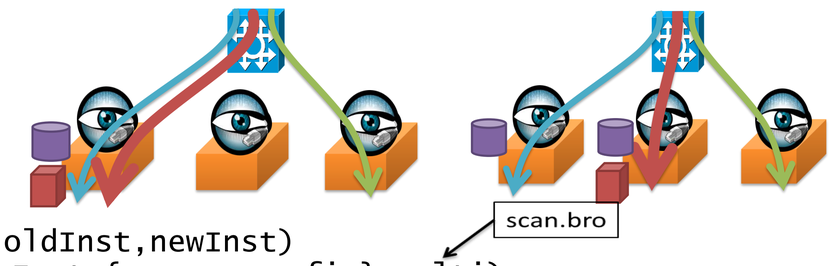
Modified Bro, PRADS, Squid and iptables to support southbound API (3-8% increase in code)

Move Operation Example



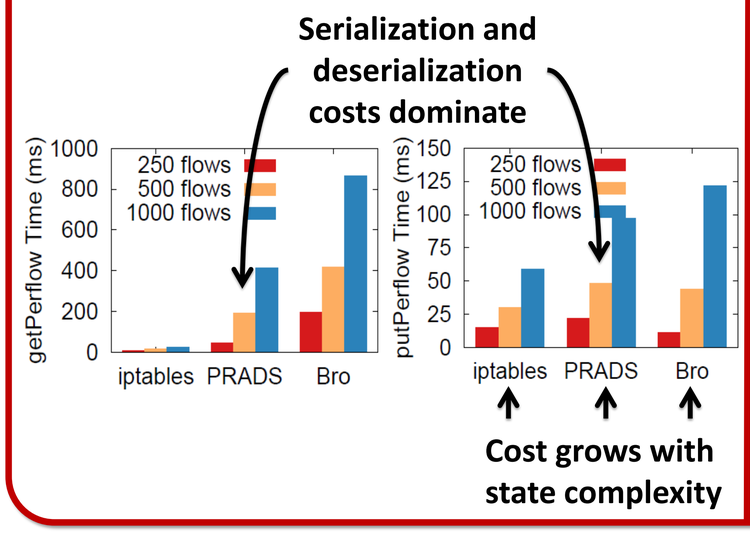
Example App: Load Balanced Network Monitoring

scan.bro
(copy + eventual consistency)
vulnerable.bro
(LF Move)
weird.bro
(LF + OP Move)

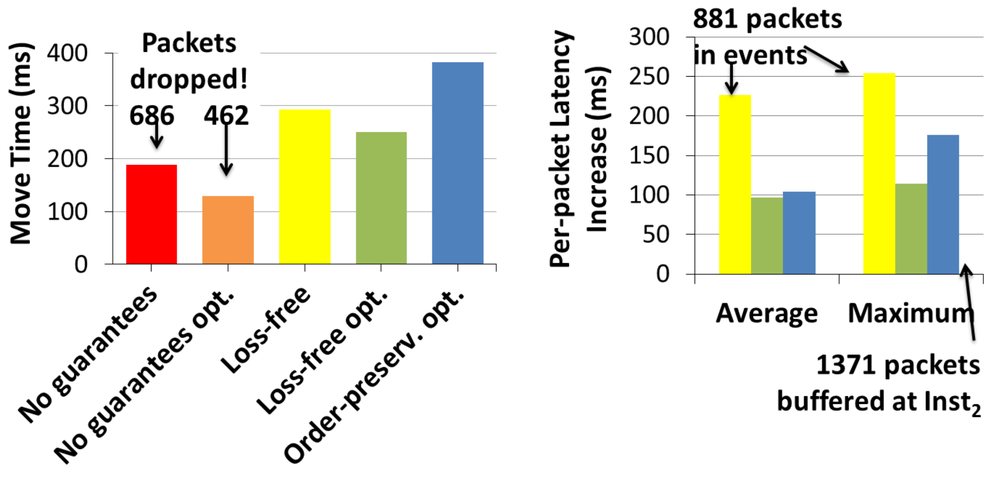



```
movePrefix(prefix,oldInst,newInst)
1. copy(oldInst,newInst,{nw_src:prefix},multi)
2. move(oldInst,newInst,{nw_src:prefix},per,LF+OP)
3. while (true):
       sleep(60)
       copy(oldInst,newInst,{nw_src:prefix},multi)
       copy(newInst,oldInst,{nw_src:prefix},multi)
```

Microbenchmark: NFs



Microbenchmark: Operations



End to End Benefits

Load balanced monitoring with Bro IDS:

- *Load*: replay cloud trace at 10K pkts/sec
- *At 180 sec*: move HTTP flows (489) to new Bro
- *At 360 sec*: move HTTP flows back to old Bro

OpenNF scaleup: 260ms to move (loss-free opt.)

- Log entries equivalent to using a single instance

VM replication: 3889 incorrect log entries

- Cannot support scale-down

Forwarding control only: scale down delayed by more than 1500 seconds