Network virtualization: role of OpenFlow & acid test for network virtualisation

Didier Colle
didier.colle@intec.ugent.be
www.ibcn.intec.ugent.be
INTEC Broadband Communication Networks (IBCN)
Department of Information Technology (INTEC)
Ghent University – IBBT

EC SPARC project: www.fp7-sparc.eu
EC OFELIA project: www.fp7-ofelia.eu
OpenFlow

OpenFlow = protocol
to make TCAMs available to ctrl
to hand-off exceptional packets to ctrl
<> Virtualization

• RULE =
  filter on sw_port, s_mac, d_mac, eth_type, vid,
  s_ip, d_ip, ip_prot, s_port, d_port
• ACTION =
  to port(x) (&s_mac=y&d_mac=z) || drop ||
  to Ofctrl || to lecgacy pipeline || ....
• STATE =
  packet&byte counters
VN ACID TEST 1: CTRL ISOLATION.

FlowVisor

Slice1 policy:
- Deny: s_ip=ip1
- Deny: d_port=80

Slice2 policy:
- Allow: s_ip=ip1
- Allow: s_ip=ip2
- Allow: d_port=80

Slice3 policy:
- Allow: d_port=80

Virtualization: slice isolation possible, but not enforced.
VN ACID TEST 2: access control & VN labeling

FlowVisor

Slice1 policy:
- Allow: $s\_ip=ip1 \& d\_port=80$
- Allow: $s\_ip=ip2 \& d\_port=80$
- AllowedPorts: 0, 2, 3 $s_1$
- AllowedPorts: 0, 2, 3 $s_2$
- AllowedPorts:: all $s_3$

Guarantee they are not high jacking IP1 or IP2 addr => drop rules on these ports for $s\_ip=ip1 \| ip2$ (FlowVisor support???)
VN ACID TEST 3: virtualization of address/port ranges

**FlowVisor**

Slice1 policy:
- Allow: s_ip=ip1
- Allow: s_ip=ip2
- AllowedPorts: 0, 2, 3 s1
- AllowedPorts: 0, 2, 3 s2
- AllowedPorts:: all s3

Slice2 policy:
- * Allow: s_ip=ip1
- * Allow: s_ip=ip2
- * AllowedPorts: 1, 2, 3 s1
- * AllowedPorts: 1, 2, 3 s2
- * AllowedPorts:: all s3

Slice3 policy:
- * AllowedPorts:: all s3

Overlapping address ranges should be possible

1) translate virt. addr. Into (internal) phys. addr. at ingress and vice versa at egress

2) similar translation in openflow messages (support in FlowVisor ???)
FlowVisor

• FlowVisor also has built in measures for regulating usage of resources outside the wirespeed forwarding path:
  – Rate limiting unmatched packets
  – Rate limiting OF requests from OF CTRLs
  – Slow path (/ legacy) forwarding
  – ...

VN ACID TEST 4: CPU usage fairness
Address the issue of asking for 100Mbps:
Receiving 50 Mbps --> unhappy
Receiving 500 Mbps --> happy

TCAM
- RULE<f1>+ACT<to_port(p1, q1)>
- RULE<f2>+ACT<to_port(p1, q2)>

Set min rate @10% For q2@p1
Set min rate @15% For q1@p1
Link Aggregation
(specific case of virtual topo)

CONCEPTUAL PROBLEM:
On what basis should FlowVisor decide to do the load balancing???
Conclusions: OpenFlow & FlowVisor

- **OpenFlow & FlowVisor:**
  - IS NOT per def virtualization, but
  - (at least conceptually) ALLOWS pretty rigorous virtualization

  - **EXCEPT** no virtual to (internal) physical address/port range mapping conceived in FlowVisor
    - **RESTRICT** to slice FlowSpace (rather than TRANSLATE) OF messages

- **OTHER features** to make rigorous VIRTUALIZATION instead of SLICING should be possible (let’s not blame existing solutions when not having defined our own acid tests):
  - Through FlowVisor code enhancements
  - Setting proper FlowVisor Slice policies and/or
  - Setting proper OF rules at the VN edge.
Conclusions: OpenFlow & FlowVisor

- OpenFlow & FlowVisor enabled Network Virtualization:
  - ++: Slicing/virtualization ends up as regular entries in TCAM --> no performance degradation
  - ++: high flexibility in defining slices/VNs (e.g., coarse and fine grained coexist), while all layers are covered by the slices.
  - --: main focus of FlowVisor on the OF protocol, less attention paid to edge of slices/VNs.
ACID tests: overview

• Ctrl isolation
• access control & VN labeling
• virtualization of address/port ranges
• CPU usage fairness