Building a Fast, Virtualized Data Plane with Programmable Hardware

Bilal Anwer
Nick Feamster
Network Virtualization

• **Network virtualization** enables many virtual networks to share the same physical network resources.

• Many possible applications:
  – Hosting of multiple service provider networks
  – Experimentation
  – Running new protocols side-by-side with old ones
Shared Infrastructure

Networks have illusion of dedicated hardware.
Network Virtualization: Requirements

• **Scalability**
  – Support large number of networks (implies sharing)

• **Performance**
  – Support real traffic at line rate

• **Flexibility**
  – Support custom network services

• **Isolation**
  – Protection of networks from each other
Goal: Fast, Virtualized Data Plane

- **Strawman approach:** Software
  - Provides flexibility
  - ...but poor performance and often inadequate isolation

- **Our approach**
  - Control plane in software
  - Data plane in hardware
  - Share hardware elements among virtual networks where possible
Virtualized Data Plane

Source -> Router-4 -> Sink

- Router-1
- Router-2
- Router-3
- Router-4 (virtual router)
- Router-5
- Router-6
- Router-7
- Router-8

2 Ethernet links
16 Ethernet links
Hardware-Based Virtualization

- **Forwarding in hardware**
  - faster than software
  - provides better isolation

- **Sharing physical substrate amortizes cost**
  - Unused hardware resources are already paid for

- **Key challenge**: Design must take advantage of both hardware and software
  - Requires interface between hardware and software
  - Requires identifying elements that can be shared among many virtual networks
Design Overview

- Control plane
  - two contexts
  - virtual environments in OpenVZ

- Interface to NetFPGA based on NetFPGA reference router
Talk Outline

- Implementation
  - Virtualization at Layer 2
  - Fast forwarding
  - Resource guarantees per virtual network
- Preliminary Results
  - Performance & Efficiency
- Conclusion and Future Work
Virtualization at Layer 2

<table>
<thead>
<tr>
<th>Virtual MAC</th>
<th>VM-ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:11:22:33:44:55</td>
<td>0x1</td>
</tr>
</tbody>
</table>

Source: 00:11:22:33:44:55
Sink: 00:11:22:33:44:55
Layer-2 Virtualization: VMAC-VE Table

- **VMAC-VE Table**
  - provides virtualization at Layer 2
  - maintains states for virtual Ethernet interfaces of each virtual environment

- **Current implementation**
  - Max. of four Ethernet interfaces per virtual router (currently limited by on-chip memory)
  - Max. of eight virtual routers working in parallel
    - Hence, 32 Table Entries
VMAC in packet determines the virtual network (and, hence, which CAMs to use)
Resource Guarantees

• CPU Isolation
  – Provided by using PCI-based NetFPGA card

• Bandwidth Isolation
  – Virtual networks are not affected by each other if they abide by their allocated bandwidth
  – What if user steps beyond allocated limited?
    • Currently, no enforcement (limitation)
    • Limit could be enforced at either ingress or egress
Evaluation

• What forwarding rates does the architecture achieve?

• How do these rates compare to the forwarding rate of the base hardware?

• How will the architecture scale with future hardware trends?
Experimental Setup

NetFPGA Based Virtual Router

Hardware
PktGen

VMAC
VE

VIRTUAL ROUTER 1
VIRTUAL ROUTER 2
VIRTUAL ROUTER 3
VIRTUAL ROUTER 4
VIRTUAL ROUTER 5
VIRTUAL ROUTER 6
VIRTUAL ROUTER 7
VIRTUAL ROUTER 8

Hardware
PktSink
Packet forwarding rates are at least as good as Linux kernel. (~2.5x for small packets)
Forwarding Performance: Overhead

Performance of up to eight virtual routers is equivalent to base router.
Efficiency

Cards will support more virtual routers as Xilinx technology improves.

- **Base router:**
  - 45% of logic,
  - 53% of BRAM,
  - 8.6M gates

- **8 Virtual Routers:**
  - 69% of logic,
  - 87% of BRAM,
  - 14.1M gates
Future Work

• Adding support for forwarding tables on SRAM.

• Providing bandwidth isolation when users exceed allocated bandwidth.

• Providing an interface to each user for performance statistics, etc.
Summary: Fast, Virtualized Data Plane

- **Scalable**
  - Design is scalable (Off-chip FIB will allow more virtual data planes.)

- **Fast**
  - Current implementation has the same performance as base hardware

- **Flexible**
  - Support for custom control and data planes

- **Provides Isolation**
  - Virtual networks don’t interfere with each other if traffic within limits
Conclusion

- Resource sharing in routers using programmable hardware is possible.

- Hardware resource sharing provides improved isolation and packet forwarding rates than software based solution.

- Current implementation achieves isolation and forwarding performance of native hardware without any overhead.
Extra
Extra
Performance Overhead

- Tested with 1,2,3,4,5,6,7,8 virtualized data-planes working in parallel and for 64-byte sized packets
- The forwarding rate was same for all eight virtualized data configuration
- All eight configuration showed forwarding rate equal to base router forwarding rate for 64-byte sized packets