Enabling Network Function Virtualization over Heterogeneous Resources

Thomas Lin, Naif Tarafdar, Byungchul Park, Paul Chow, and Alberto Leon-Garcia

The Edward S. Rogers Sr. Department of Electrical and Computer Engineering
University of Toronto, ON, Canada

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Overview

Motivation: IoT data analytics in clouds

NFV & SDN enable flexibility

- *What about performance?*

Growing heterogeneity of clouds

- *FPGAs and GPUs!*

Software-Defined Infrastructure

Abstract creation of VNFs and service chains

Experiments w/ heterogeneous VNFs

Conclusion

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Motivation

IoT era creating a surge of new cloud-based applications and services

- *A LOT of network traffic and data*

IoT analytics challenge:

- *Processing at network edge, closer to end-users*
- *Real-time processing for fast control loops*

By 2020, an estimated 92% data workloads will be cloud-based *

* Cisco Global Cloud Index: Forecast and Methodology, 2015–2020

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Motivation cont.

Streaming processing needs are all different!

- *Depends on characteristics of traffic and data*

Data processing can be:

- *Compute Intensive (a lot of CPU time)*
- *Network Intensive (large bandwidth)*

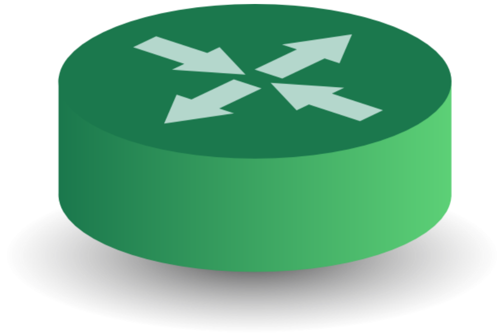
Is there one-size-fits-all approach?

- *Cloud should support flexible processing functions*

Network Function Virtualization

Network Function Virtualization (NFV)

- *Virtualize traditional network "boxes" in cloud*
- *E.g. Firewalls, Routers, Intrusion Detection Systems, etc.*



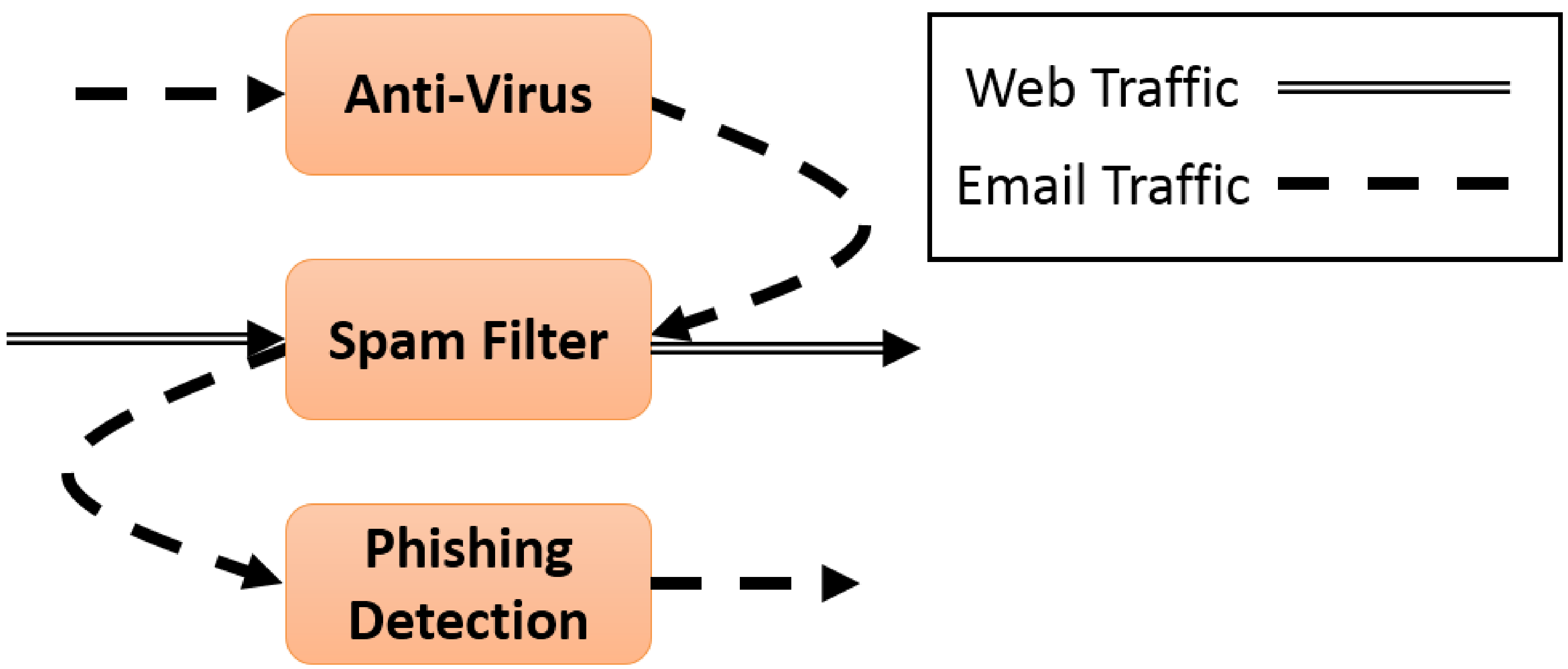
Create *Virtualized Network Functions (VNFs)*

- *Flexibly deploy VNFs closer to where they are needed*
- *Rapid development of new network functions & services*

Software-Defined Networking

Software-Defined Networking (SDN) complements NFV

- *Dynamically place VNFs where they're needed*
- *Use SDN to steer traffic to VNFs' location*
- *Supports mobility and re-use of VNFs*



Performance?

Traditional VNFs use SW in virtual machines (VMs)

- *Software is very flexible => Rapid development cycles*
- *Low performance compared to dedicated boxes* ☹

NFV concept is agnostic to computing resource

- *Variety of data and high volume in IoT era will likely require specialized resources*
- *Can use FPGAs, GPUs, Network Processors, etc.*
    - *Much higher performance*

Growing Heterogeneity of Clouds

Heterogeneous resources more common now

- *AWS or Azure (Industry); SAVI (Academic)*

Few users have required skills to program and debug FPGAs and GPUs

Provisioning VNFs should be *simple*

Most users don't care how VNFs are implemented, as long as it *works and performs well*

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Enabling Heterogeneous NFV Service Chains

Can we abstract VNF creation for users?

We need a converged control & management system to abstract creation of:

- *VNFs (traditional SW, or specialized HW)*
- *Service chains (steer traffic through 1 or more VNFs)*

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Required Functionalities

Abstraction over multiple resource types

- *X-as-a-Service*
- *Enable users to acquire virtual resources of any type*

Dynamic service chaining

- *Need ability to steer traffic to VNFs*
- *Enables scaling performance by dynamically switching from SW to HW and vice-versa*

Objectives

Objectives that can simplify system for users

Unified APIs

- *Avoid creating new set of APIs per new resource type*

Orchestration

- *Automate deployment and retirement of applications and services (accelerates IT service delivery)*

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Objectives cont.

Monitoring

- *Monitor health of applications and services*
- *Crucial for auto-scaling and collecting statistics*
- *Calculate KPIs and other metrics*

VNF repository

- *Existing clouds provide bare resources to users*
- *Users must know how to create their own VNFs*
  - *Major barrier to NFV adoption*
- *Allow users to deploy pre-build VNFs*

Software-Defined *Infrastructure* (SDI)

SDI aims to integrate the management and control of ***all infrastructure components***…

- Compute + Networking + Storage + Access + etc.

While providing open interfaces for users…

- e.g. RESTful APIs for enabling self-service

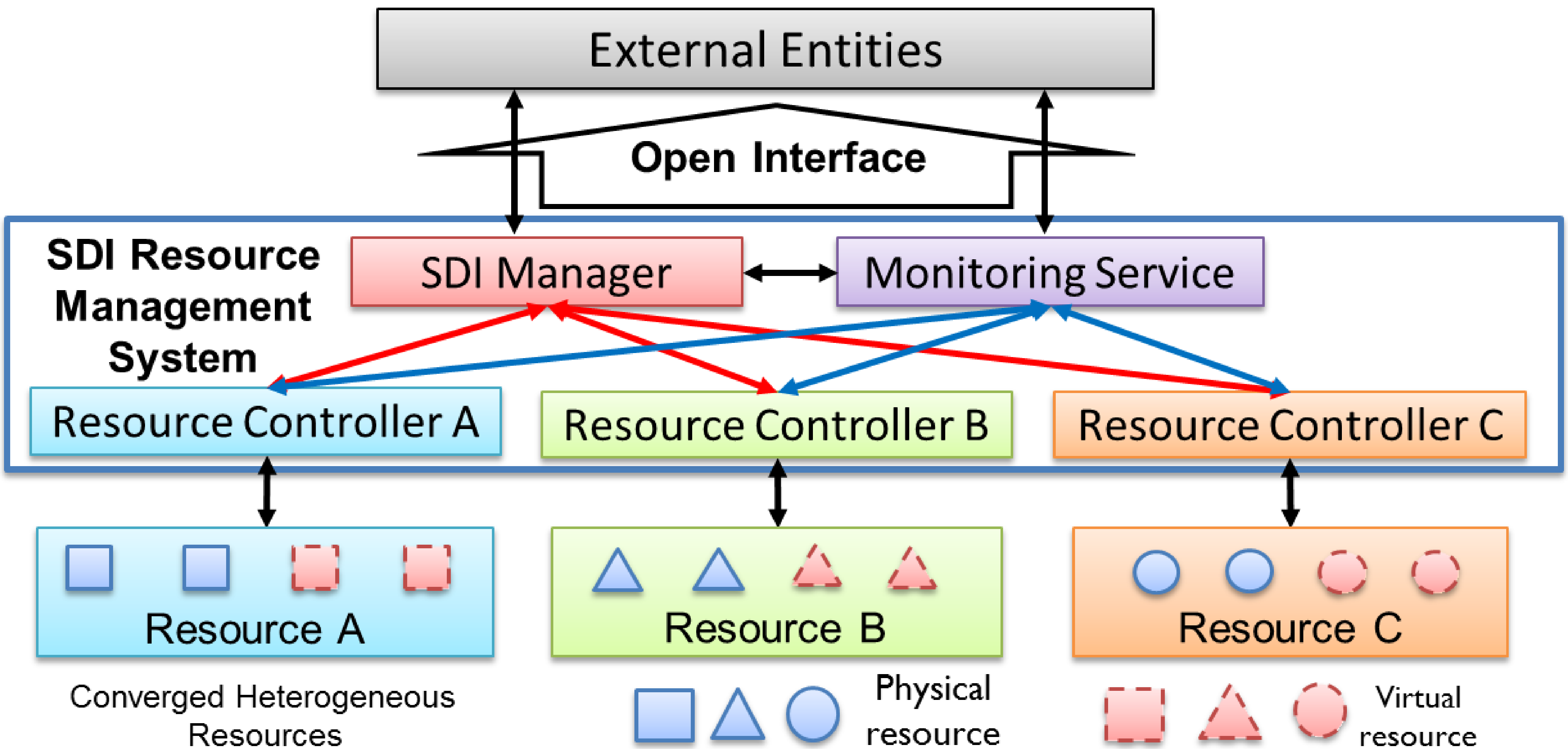
All within a centralized control plane

- Control logic implemented in software

Software-Defined *Infrastructure* (SDI)

Infrastructure control can be centralized

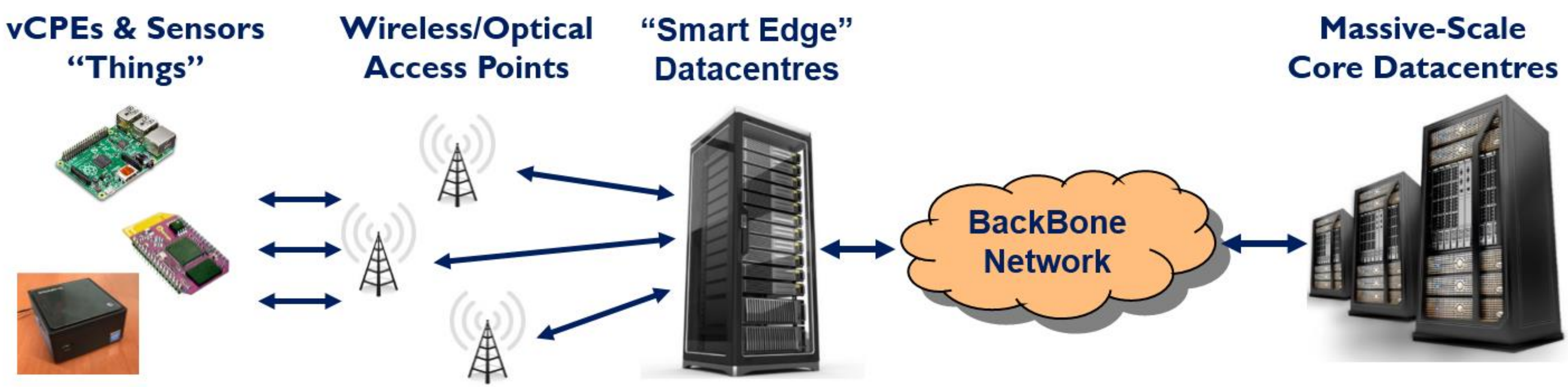
- Grants controller the benefit of a *global view* of the infrastructure, its topology, configuration, and state



SAVI: Smart Applications on Virtual Infrastructure

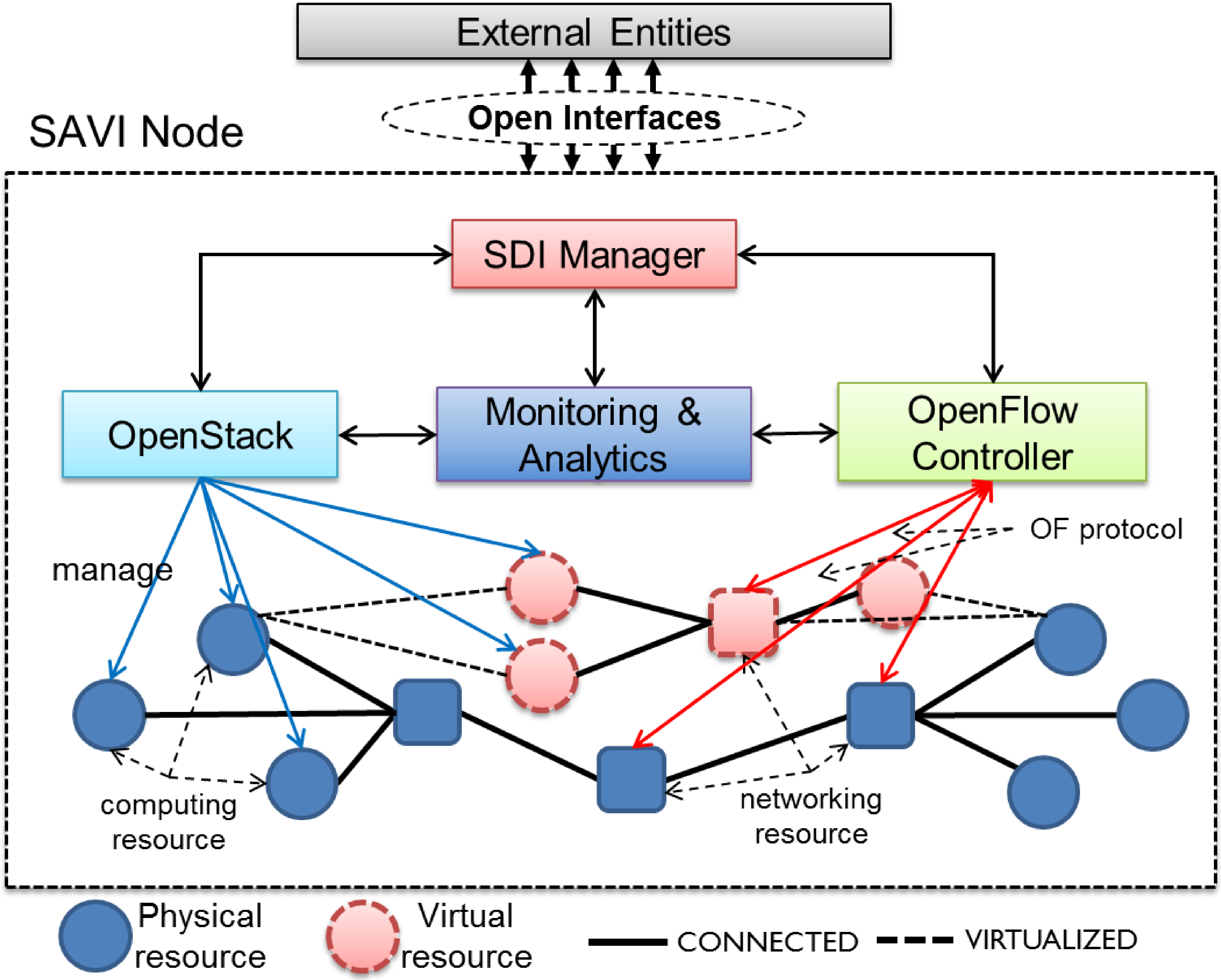
Multi-Tiered Cloud Testbed

- Core nodes
  - *Massive-scale, many resources*
- Smart Edge nodes
  - *Smaller scale, may be connected to access points*



The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

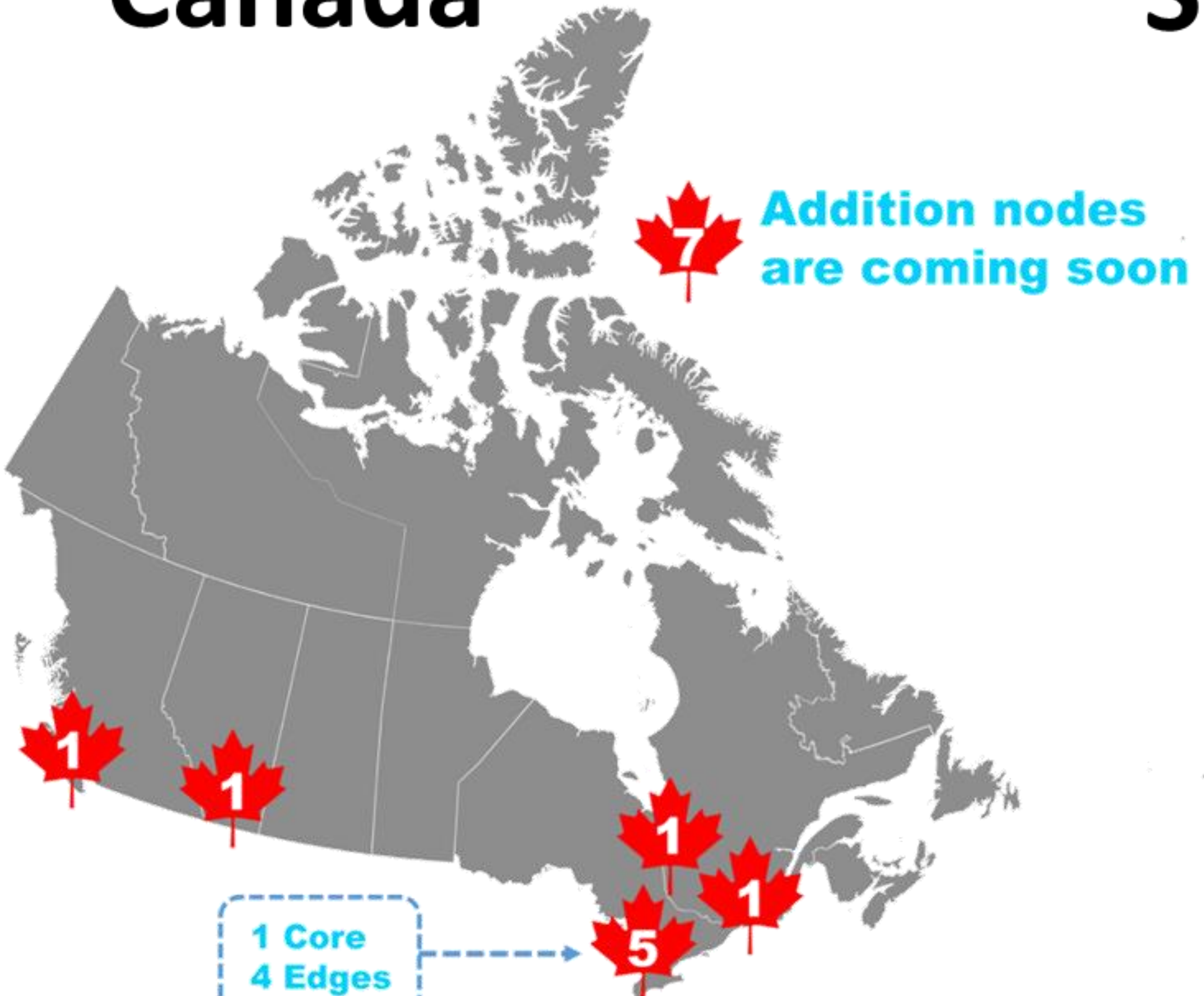
SAVI Node Based on SDI



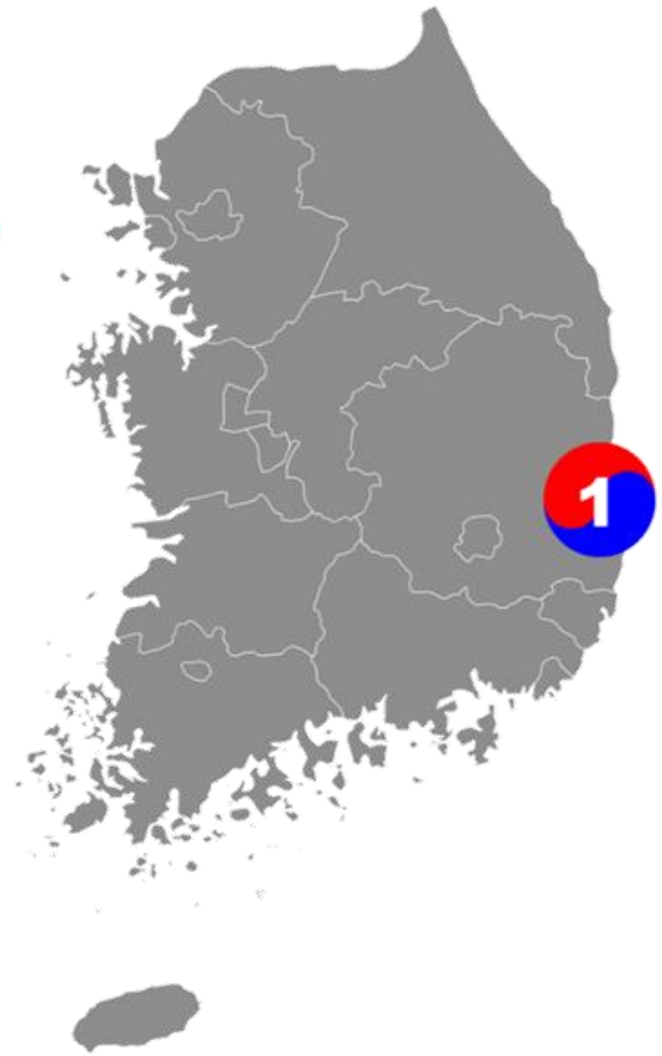
The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Current Status of SAVI

Canada



South Korea



Federation with geni

Exploring Networks
of the Future

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Current Status of SAVI cont

As of year-end 2016

Cores	RAM (GB)	Storage (TB)	GPUs
3700+	8700+	440+	7

FPGAs	SDRs	Wireless	IoT
60+	15	23	60+

More to come

SAVI "Smart Edge" Based on SDI

Smart Edge" is a heterogeneous cloud

- Users can create intelligent applications and services

OpenStack Nova to virtualize compute resources

- Nova w/ modifications to support FPGAs & GPUs
- PCIe passthrough of custom hardware to VMs

OpenFlow + OpenStack Neutron to virtualize network

- Explicit access control of network flows; Control at L2 Ethernet

SDI exposes APIs for installing L2 flows *directly into network*

- Users can configure custom paths for their resources' traffic

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Meeting Other Objectives

Modify Nova to support FPGAs and GPUs

- Abstracts multiple resource types using unified API
  - *New resource types exposed as new "flavors"*
- Means we can integrate with existing OpenStack services

OpenStack Glance for VNF repository

- Store FPGA bitstreams; or
- Pre-packaged VM images can re-program FPGA upon boot-up

OpenStack Heat for orchestration

MonArch (built in-house) for monitoring

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Experiments

Conducted 2 experiments to demonstrate system

1. Hardware-accelerated signature matching

- Demonstrate need for custom accelerated VNFs

2. Hot-swapping SW to HW VNF

- Study interruption to network traffic

FPGA VNFs created using High-Level Synthesis (HLS)

- C++ code compiled to FPGA bitstreams

Experiment 1: Signature-Based Filter

Scan entire packet payload for set of signatures

Experiment setup:

- 10 GE network, UDP with 1470 Byte payloads
- Signatures are all 10 Bytes
- We vary # of signatures to look for from 10 to 140

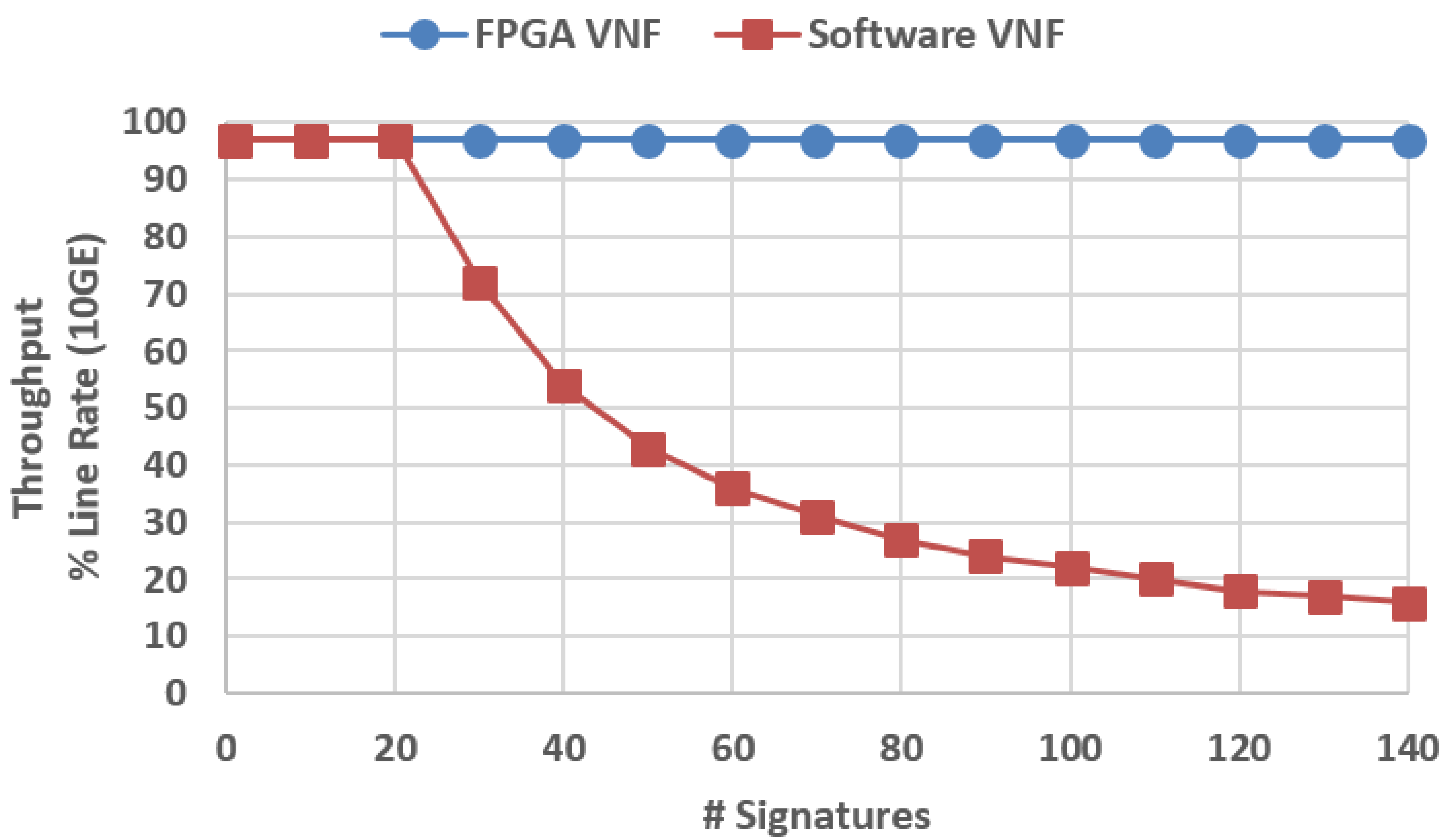
Software VNF uses many optimizations:

- DPDK to bypass network stack, direct access to NIC
- Run on server with high clock-rate
  - *Cloud servers usually have lower clock rate*
- Many other micro-optimizations

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Experiment 1 Results

SW VNF becomes throughput bottleneck as # signatures increase



Experiment 2: Hot-Swapping VNFs

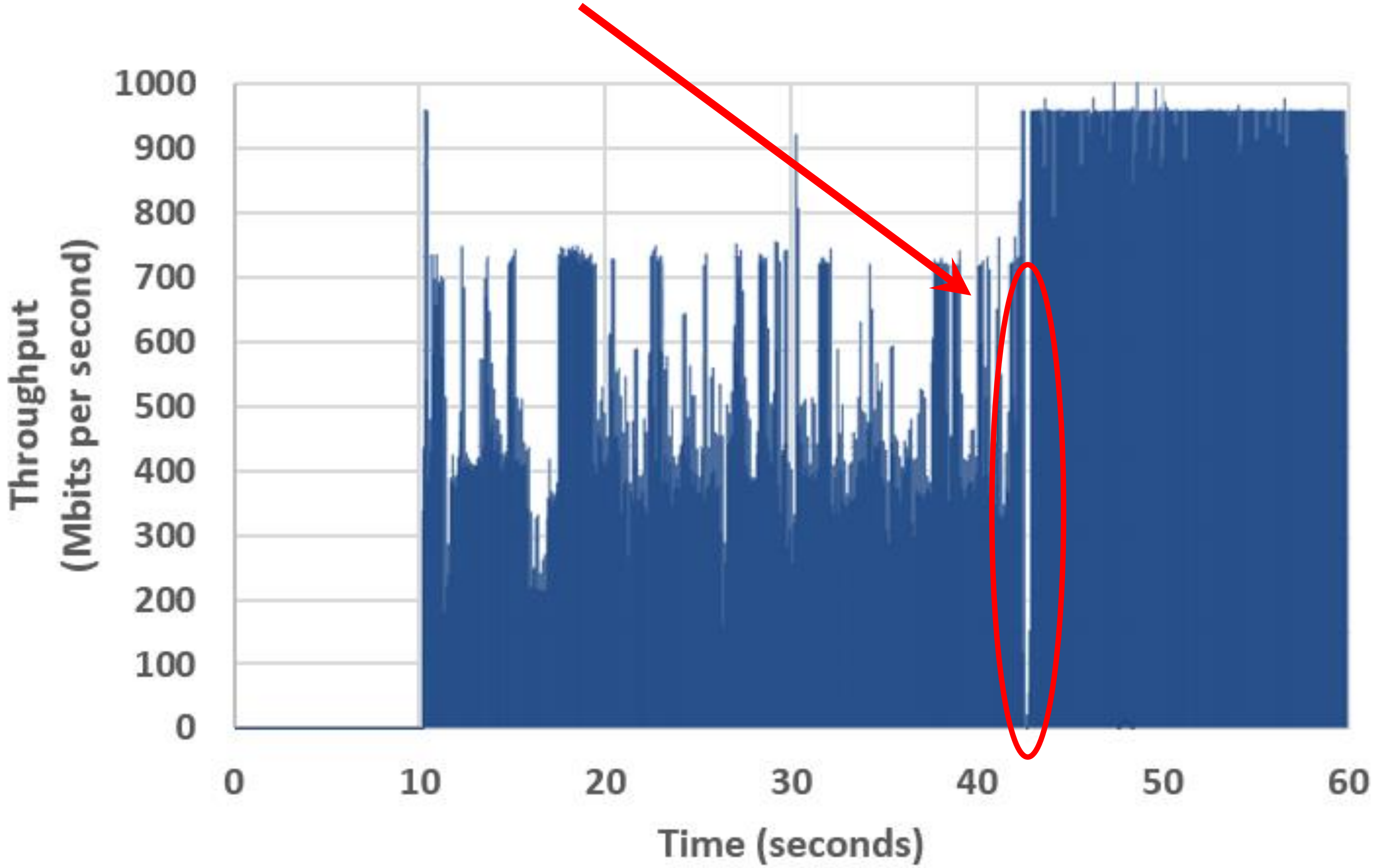
Hot-swap SW VNF for HW VNF

Experiment setup:

- Simple packet forwarding
- UDP flooding in 1 GE network
- No SW optimizations done (lazy implementation)

Experiment 2: Results

Application downtime ~ 0.1 seconds @ 43s



Incremental Design Flow

Our system enables piece-wise transition of existing VM-based software VNFs to FPGAs or GPUs

1. Implement existing service chain in VMs using SW
2. Implement each VNF in FPGA or GPU
3. Hot-swap SW VNF for FPGA or GPU-based VNF
4. Verify service chain functions identically. If yes, repeat steps (2) and (3) for each VNF in chain; else debug the VNF

Summary

IoT era will produce a lot of traffic

- Real-time data analytics at edge for fast response
- VM-based SW VNFs may not meet performance needs

Heterogeneous compute resources in cloud

- Few users have required skills; VNFs should be easier

Designed & implemented SDI-based architecture for creating heterogeneous NFV service chains

- Unified APIs to create different types of VNFs
- APIs for dynamic service chaining
- Create VNF repository with FPGA and GPU VNFs

Thanks!

Questions?

Email: t.lin@mail.utoronto.ca

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO