DESIGN AND MANAGEMENT OF DOT : A DISTRIBUTED OPENFLOW TESTBED

Presented By

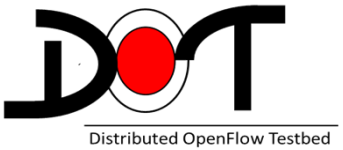
Arup Raton Roy

MMath Student

David R. Cheriton School of Computer Science

University of Waterloo, Canada

Joint work with Md. Faizul Bari, Mohamed Faten Zhani, Reaz Ahmed, and Raouf Boutaba



Outline

- Introduction
- Motivation
- DOT Architecture
- Management framework
- Evaluation
- Conclusion

DOT
Distributed OpenFlow Testbed

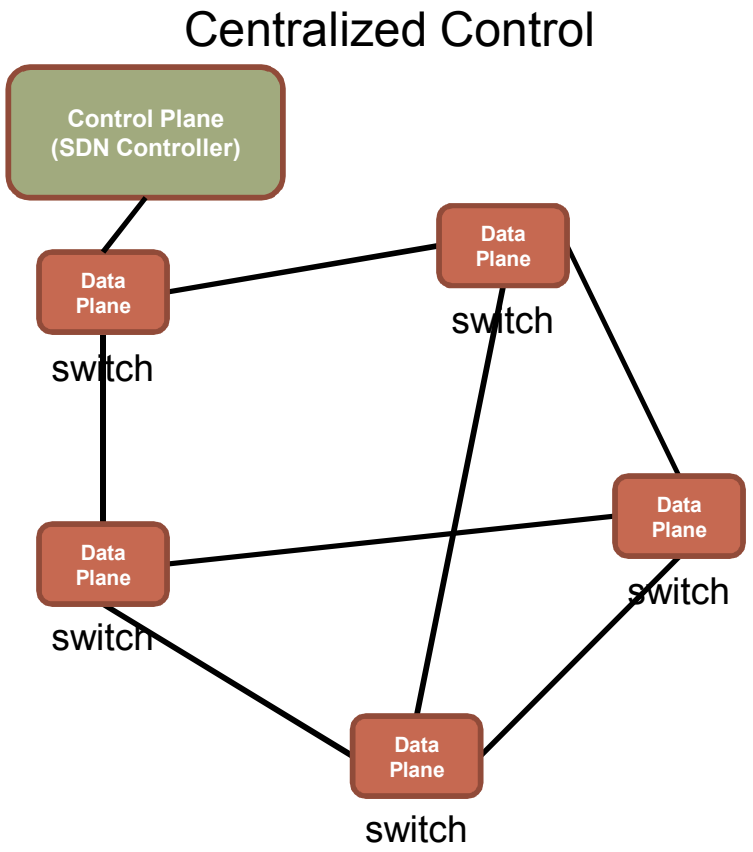
Outline

- Introduction
- Motivation
- DOT Architecture
- Management framework
- Evaluation
- Conclusion

DOT
Distributed OpenFlow Testbed

SDN and DOT

- Software defined networking (SDN)
  - Separates the control plane from the forwarding device
  - Uses logically centralized control plane
- Distributed OpenFlow Testbed (DOT)
  - A distributed emulator for SDN
  - Emulates OpenFlow network
  - Simulates link bandwidth and delay



Outline

- Introduction
- Motivation
- DOT Architecture
- Management framework
- Evaluation
- Conclusion

Distributed OpenFlow Testbed

State of the art

- EstiNet
  - Commercial product
  - Its an emulator with simulated clock
  - If there is insufficient computational resource on the machine simulated time can be slowed down
- OFELIA (**O**pen**F**low in **E**urope: **L**inking **I**nfrastructure and **A**pplications)
  - A flowspace is assigned to a user
  - VMs as end hosts
- Mininet
  - De facto standard SDN emulator
  - Emulates an SDN network in a *single* machine
  - Uses Linux *container* to emulate hosts
  - Supports different types of virtual switches

DOT
Distributed OpenFlow Testbed

Mininet – A good start! But….

In Mininet Support

….Scalability on a single system is something we can work on improving, but for now I'd recommend trying a smaller configuration on your hardware setup….

https://mailman.stanford.edu/pipermail/mininet-discuss/2012-June/000931.html

Mininet – A good start! But….

In Research Paper

..found that Mininet is inadequate for our purpose as it cannot handle the amount of traffic that we wanted to simulate….

"Dynamic controller provisioning in software defined networks" – Bari et al. (CNSM 2013)

Distributed OpenFlow Testbed

Mininet – A good start! But….

In Mininet Wiki

Mininet's original goal was "1000 nodes on your laptop" but such networks aren't really practical. ....

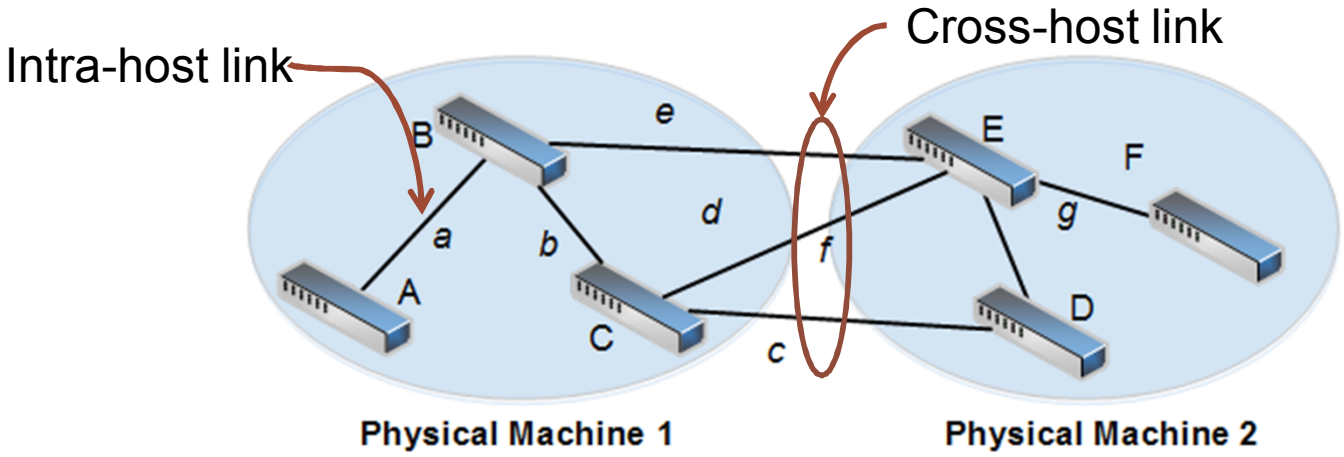
https://github.com/mininet/mininet/wiki/Ideas

Outline

- Introduction
- Motivation
- DOT Architecture
- Management framework
- Evaluation
- Conclusion

Achieving Scalability

- Distributing the emulation across multiple physical machines
- Embedding algorithm partitions the *logical network* into multiple physical hosts
  - Formulated as an ILP
  - Proposed a greedy heuristic



Embedding: Formulation

- DOT embedding is formulated as an ILP
- Objective function

  Minimize $\alpha C^T + \beta C^E$

- Where
  - C^T →Represents the number of cross-host links and their bandwidths
  - C^E →Number of active physical hosts
- Constraints
  - Physical resource constraints
  - Cross-host link delay constraint

  ⟶ DOT embedding is NP-hard

Distributed OpenFlow Testbed

Embedding: Heuristic

- **Switch selection**
  - Select a switch i using

$$R_i = \gamma_D R_i^D + \gamma_B R_i^B + \gamma_N R_i^N$$

    - Where
      R_i^D →Degree ratio
      R_i^B →Resource ratio
      R_i^N →Neighbor ratio
- **Host selection**
  - Select an active physical host p for switch i

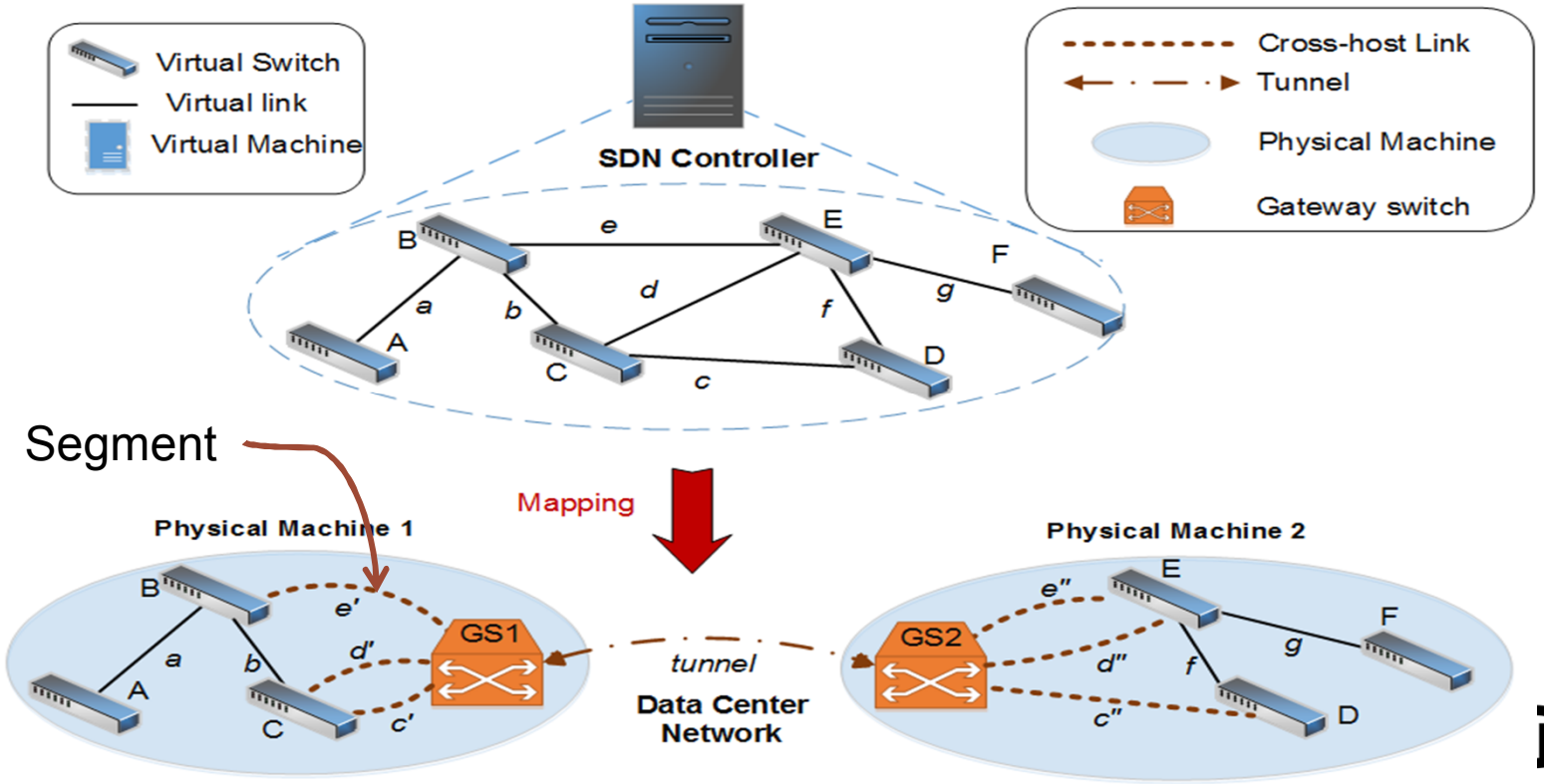
$$F_{ip} = \lambda_R F_{ip}^R + \lambda_N F_{ip}^N$$

    - Where
      F_{ip}^R →Residual capacity ratio
      F_{ip}^N →Locality ratio
  - Otherwise, activate another feasible host
- Repeat until all switches are assigned or no embedding is possible with the policy

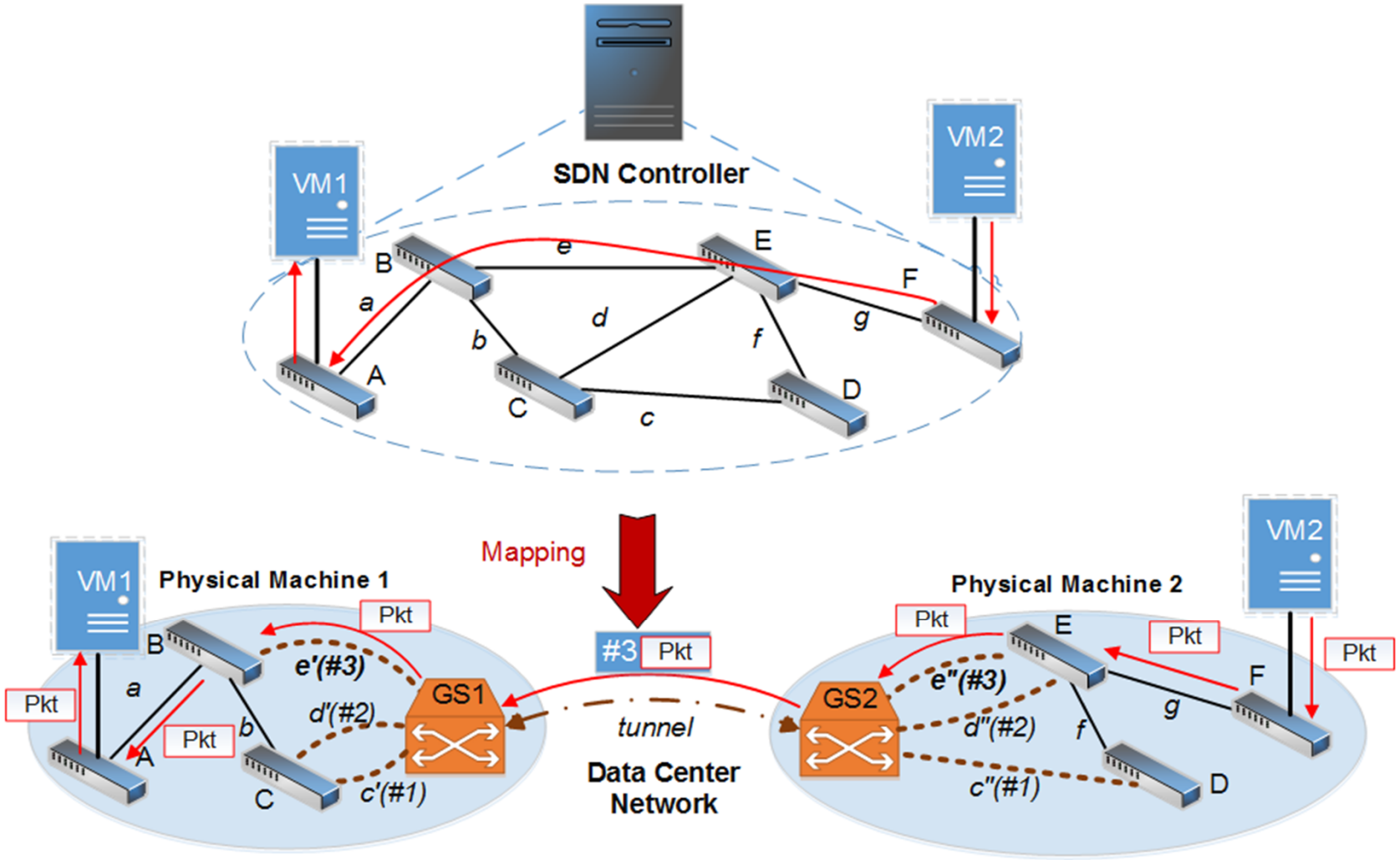
Distributed OpenFlow Testbed

Achieving Transparency

- Gateway Switch (GS) is added to *each active physical* host
  - It unicasts packets passing through the cross-host links
  - It hides the partitioning from the SDN controller

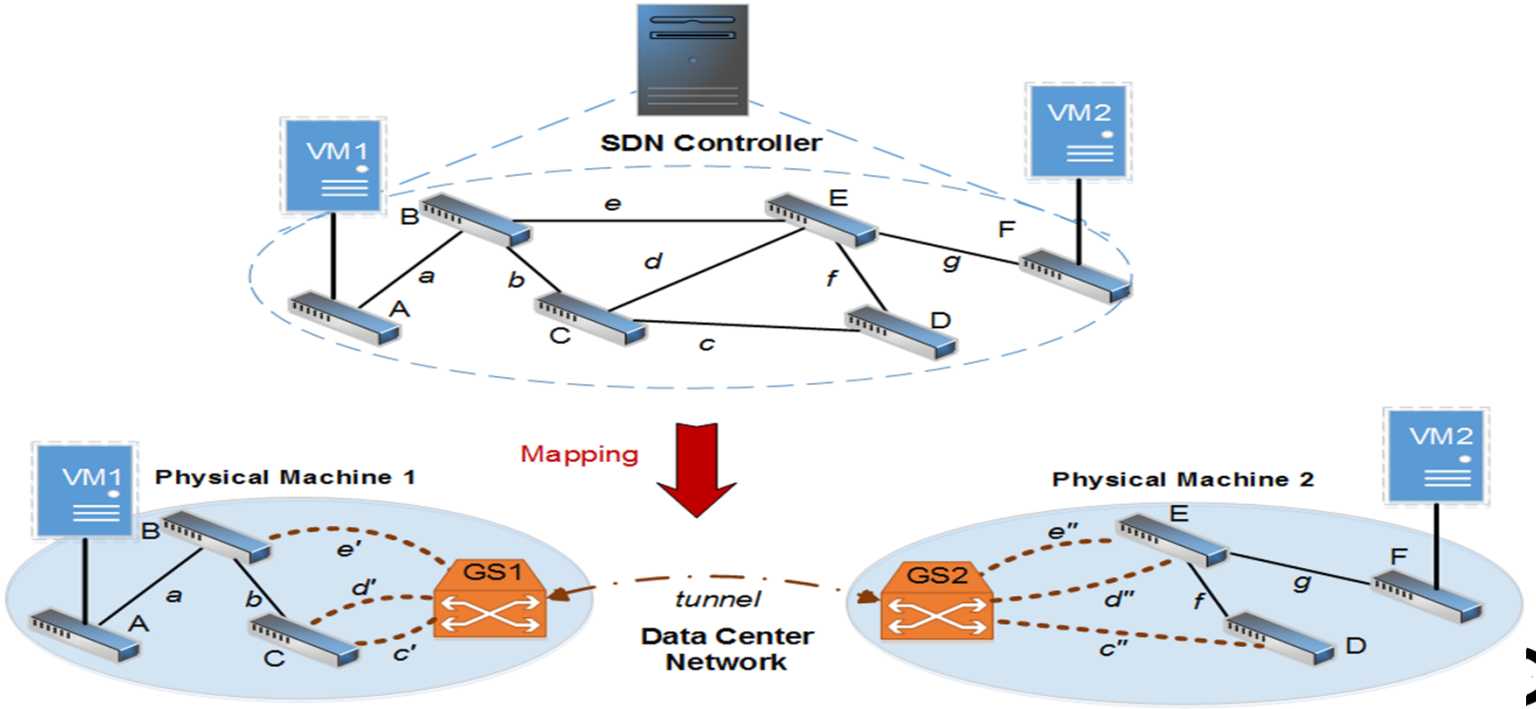


Inter-host Traffic Forwarding



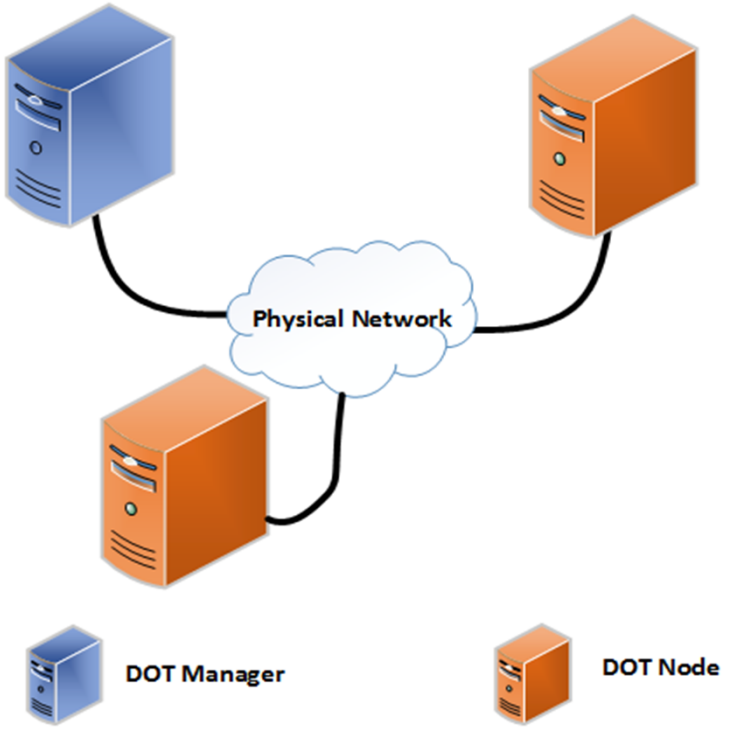
Achieving Flexibility

- DOT supports
  - Container based virtualization
  - Full virtualization (End-hosts → full fledged VM)
- VMs can be used for
  - Generating traffic
  - Running SDN controller
  - Providing network services (e.g, firewall, IDS)

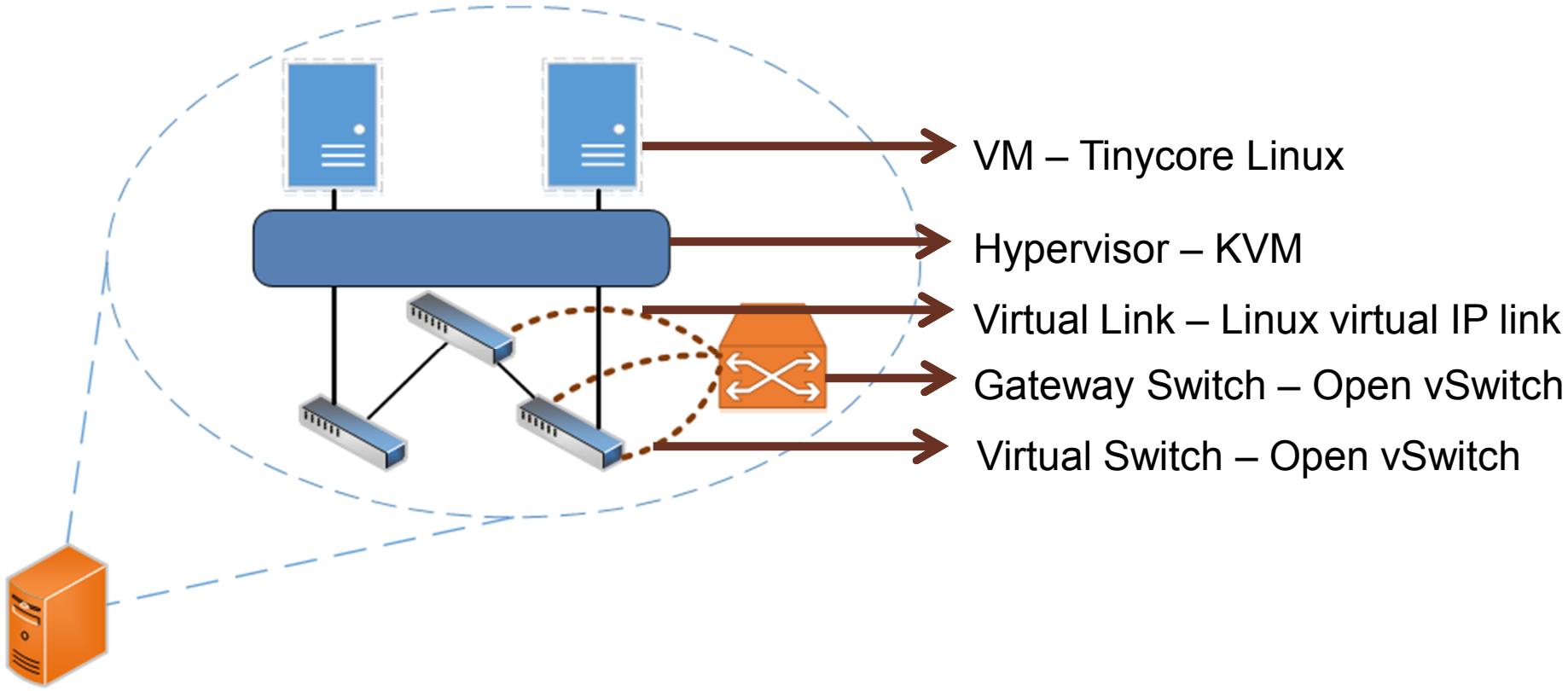


A Typical DOT Deployment

- DOT uses one *DOT Manger* and one or more *DOT Nodes*
- DOT Manager
  - Allocates and provisions the virtual infrastructure
  - Provides centralized access and monitoring facility
- DOT Node
  - Hosts the virtual switches and VMs



DOT Node

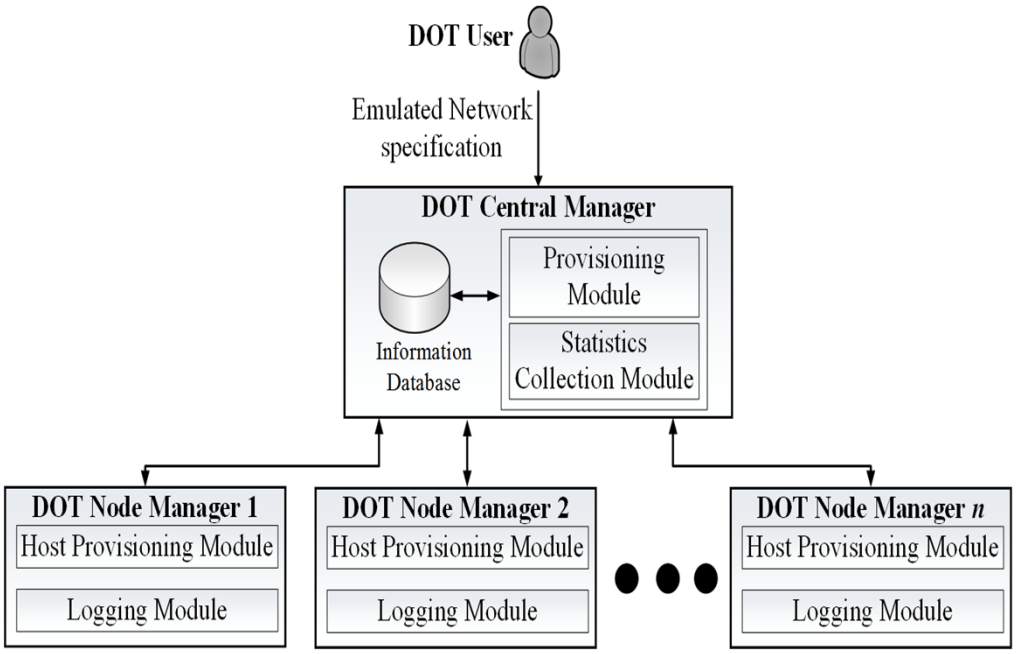

VM – Tinycore Linux
Hypervisor – KVM
Virtual Link – Linux virtual IP link
Gateway Switch – Open vSwitch
Virtual Switch – Open vSwitch

Outline

- Introduction
- Motivation
- DOT Architecture
- Management framework
- Evaluation
- Conclusion

Management Architecture

- DOT Central Manager
  - Provisioning module runs an embedding algorithm to determine the placement and instructs the host provisioning module about it.
  - Statistics collection module gathers information from logging modules of each DOT nodes
- DOT Node Manager
  - Host Provisioning module is responsible for allocating and configuring the virtual instances
  - Logging module collects local statistics

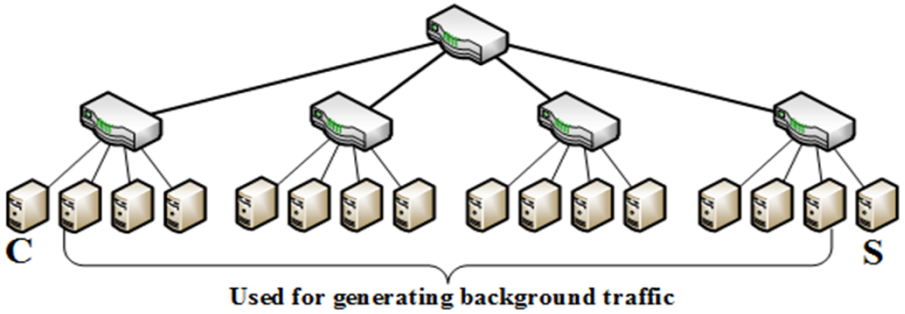


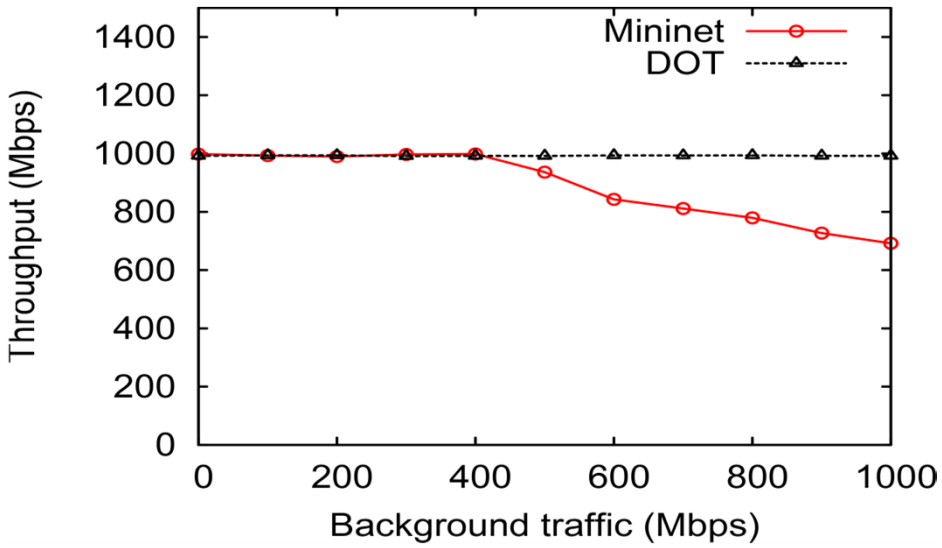
Outline

- Introduction
- Motivation
- DOT Architecture
- Management framework
- Evaluation
- Conclusion

Comparison to Mininet

- We consider a fat-tree topology
- We run *iperf* to generate traffic between two hosts
- Foreground traffic
  - UDP traffic at a constant rate of 1000Mbps between C and S
- Background traffic
  - 7 UDP client-server pairs are chosen randomly



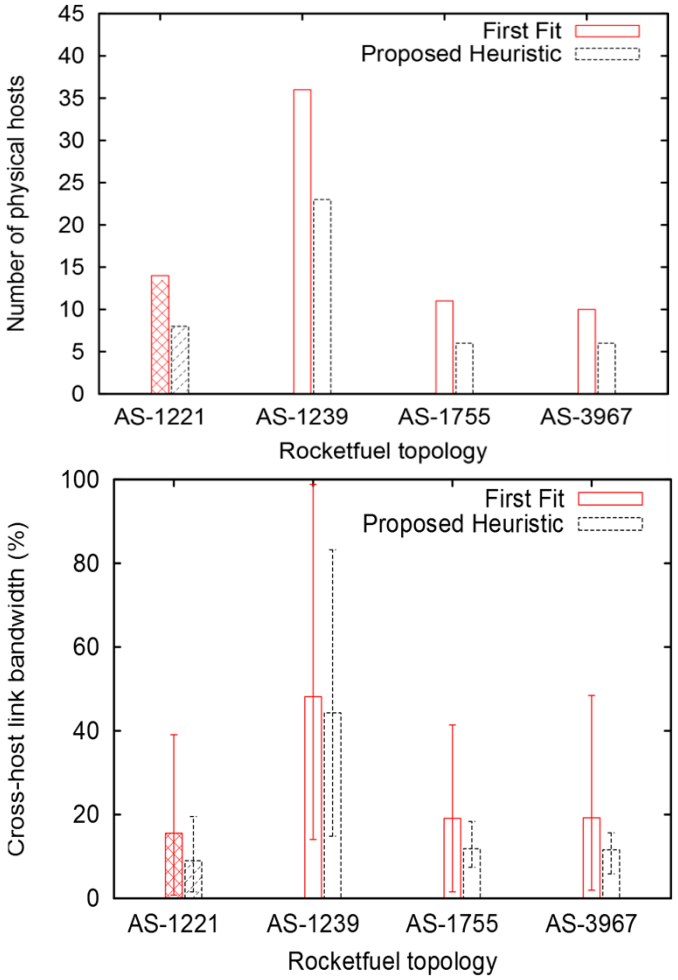


Embedding Algorithm

- We compare four different topologies (from RocketFuel [1])

Topology	#of Switch	#of Link
AS-1221	108	306
AS-1239	315	1944
AS-1755	87	322
AS-3967	79	294

- We compare the proposed heuristic with *First Fit* approach for these topologies.



[1] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson.Measuring ISP topologies with rocketfuel. *IEEE/ACM Trans. Netw.* 12, 1 (February 2004), 2-16.

Outline

- Introduction
- Motivation
- DOT Architecture
- Management framework
- Evaluation
- Conclusion

DOT
Distributed OpenFlow Testbed

~~Conclusion~~ Again its just a start…

- Until today, DOT
  - Solves scalability problem of Mininet
  - Hides distributed deployment of virtual infrastructure from SDN controller
  - Provides opportunities to emulate a wider range of network services
- Future DOT
  - Have auto scaling feature
  - Provide *multi-user* support
  - Support configurable logging facility
  - Have RESTful APIs for remote monitoring and management

DOT
Distributed OpenFlow Testbed

Everything about DOT

dothub.org

Distributed OpenFlow Testbed

An emulator for large scale OpenFlow networks

Home | Installation | Tutorial | Publications | Forum | FAQ

What is DOT?

Distributed OpenFlow Testbed (DOT) is a tool for emulating large scale OpenFlow based Software Defined Networks. DOT distributes the emulated network across several physical machines to provide guaranteed CPU time, bandwidth and network latency for the emulated components (i.e., switches, hosts and links). It scales with the network size and traffic volume. It also has built-in support for configuring and monitoring the emulated components from a central point. DOT is an outcome of an ongoing research project of the WatSDN research group at the University of Waterloo.

Questions?

Distributed OpenFlow Testbed