Distributed Network Experiment Emulation

Giuseppe di Lena, Andrea Tomassilli, Damien Saucez, Frédéric Giroire, Thierry Turletti, Chidung Lac, Walid Dabbous

GEFI 2019,
Coimbra, november 7th 2019

Network Emulation

- Emulation is an important step in the evaluation cycle of network applications and protocols
  - provides a fully controlled environment to test real protocols & applications
  - in a reproducible way, without any modification
  - scalable scenarios
- Mininet is the reference emulator in the network community
  - but it has been designed to run on a single machine only
- Distrinet allows distributing Mininet scenarios on any pool of computing resources including:
  - Linux cluster of machines (e.g., Grid5000)
  - the cloud (e.g., Amazon EC2)
  - or any Linux-based testbed (e.g., R2lab)

Network Emulation on Testbeds

- Interest for network emulation capabilities in testbeds
  - run large scale scenarios with realistic and accurate results
  - attract more users to test-beds
- The Mininet API offers the right level of abstraction
- Distrinet is a good way to implement Mininet in testbeds
- Provides **uniform interface** on federated platforms Fed4Fire for running scenarios
  - including hybrid ones that involve more than one testbed

Mininet

+Widely used

- Mininet works well when the virtual hosts and the virtual switches do not require a lot of resources
  - can return wrong results when the physical host is overloaded
  - resource intensive experiments need to distribute the load
- Mininet is also problematic when you need isolation in the virtual nodes

Related Work

- Maxinet runs multiple instances of Mininet, one per cluster node
  - But is not directly compatible with Mininet scripts
- No notion of performance of worker nodes
  - weak machines become a bottleneck for the emulation
- No notion of physical network properties
  - no influence mapping of partitions to workers
- Maxinet has no automatic deployment
  - needs careful manual configuration
- vNode not completely isolated
  - like Mininet

Related Work

- Containernet an extension of Mininet
  - isolates nodes using docker containers
- Can be combined with Maxinet to distribute network experiments
  - But no performance guarantee, like Maxinet
- DISTEM
  - Used in Grid5000 clusters
  - Uses Multicast Vxlan Tunnels, cannot run on Clouds
  - One should specify manually which physical host to use
  - not compatible with Mininet API

Distrinet

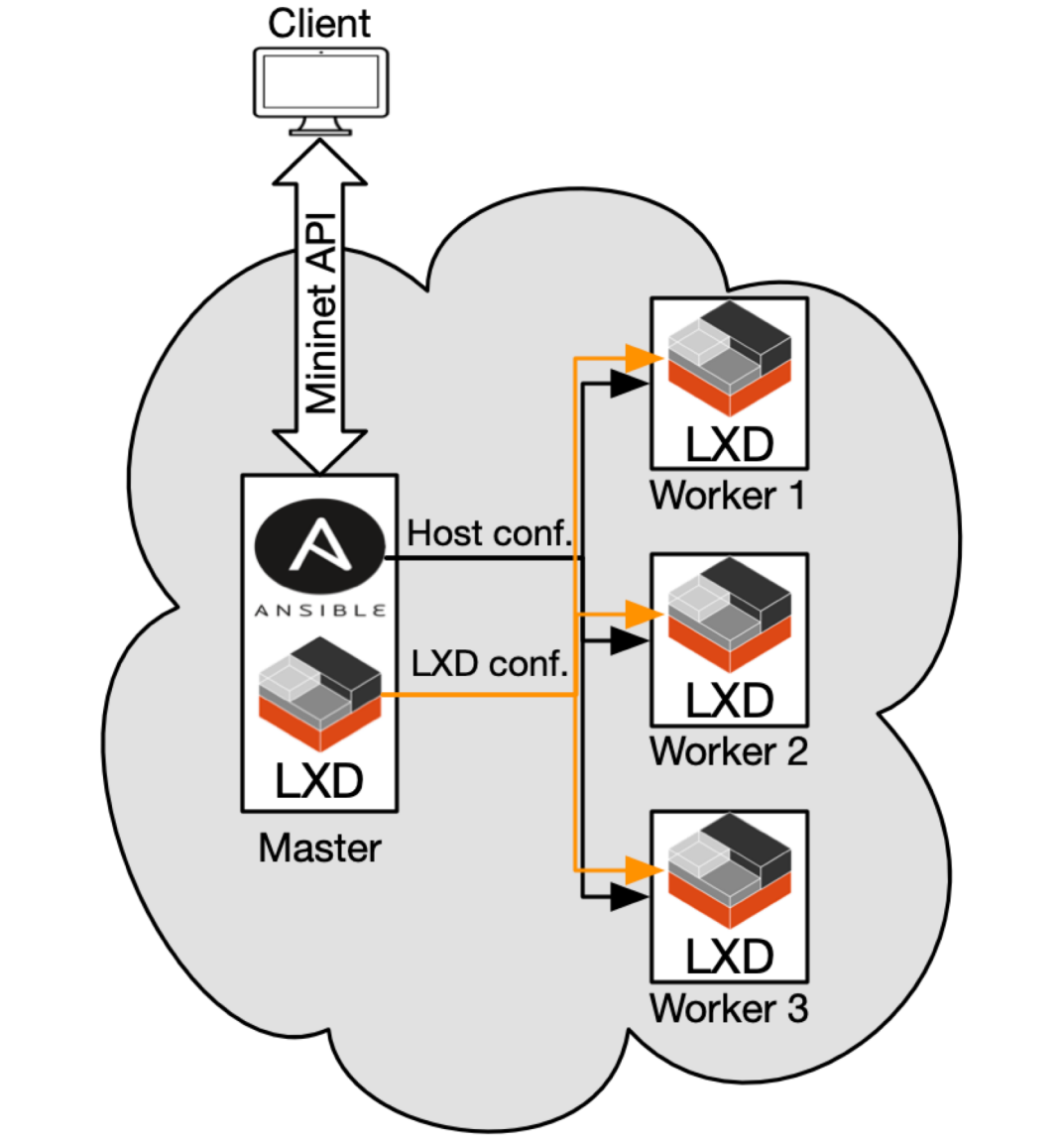
- Available for physical machines clusters (G5k) or cloud (AWS)
  - Using Unicast Vxlan to be compatible with the clouds
- Automatic Deployment of the infrastructure
  - Allocation algorithm for Physical or Cloud infrastructure
  - Taking as input both the physical and virtual network topologies/information
- Compatible with Mininet API
- vNodes Isolated with LXC containers

Cluster vs Cloud

- Cluster:
  - physical infrastructure known
  - complete control of the resources
  - Network embedding problem
    - Number of hosts
- Cloud:
  - partial knowledge and control
  - Vector bin packing problem
  - Cost of the experiment

Architecture

- Ansible: Configure the physical machines or Amazon EC2 instances
- LXD: Run containers in the configured machines
- Mininet Api

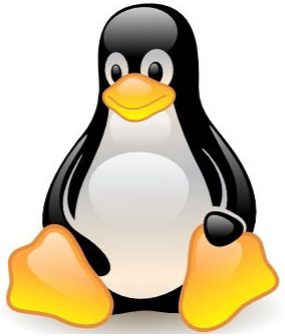


Distrinet General Configuration

- Requirements:
  - The Client must be able connect to the Master host via SSH
  - The Master Host must be able connect to all the Worker Hosts via SSH
- Environment set up:
  1. Distrinet connects to the Master host to install and configure Ansible
  2. With the previous configuration, Ansible installs and configures LXD in the Master and the worker hosts
  3. Distrinet is ready for the emulation

Distrinet

- Available for everyone
- Develop a Provision class for your test-bed



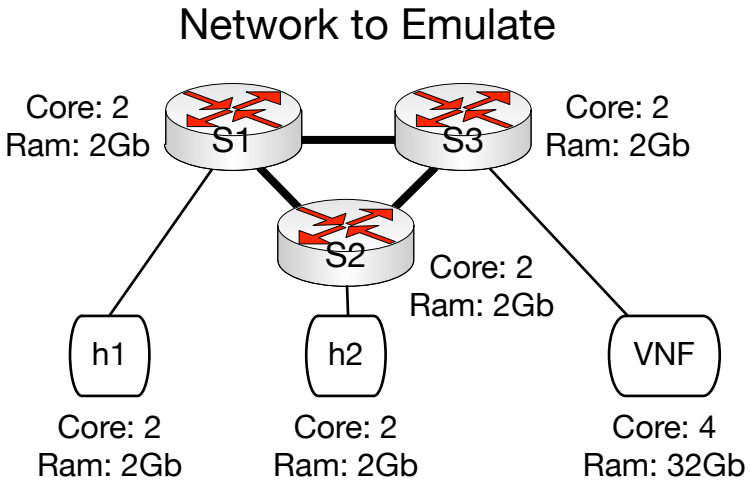
amazon web services™ | EC2

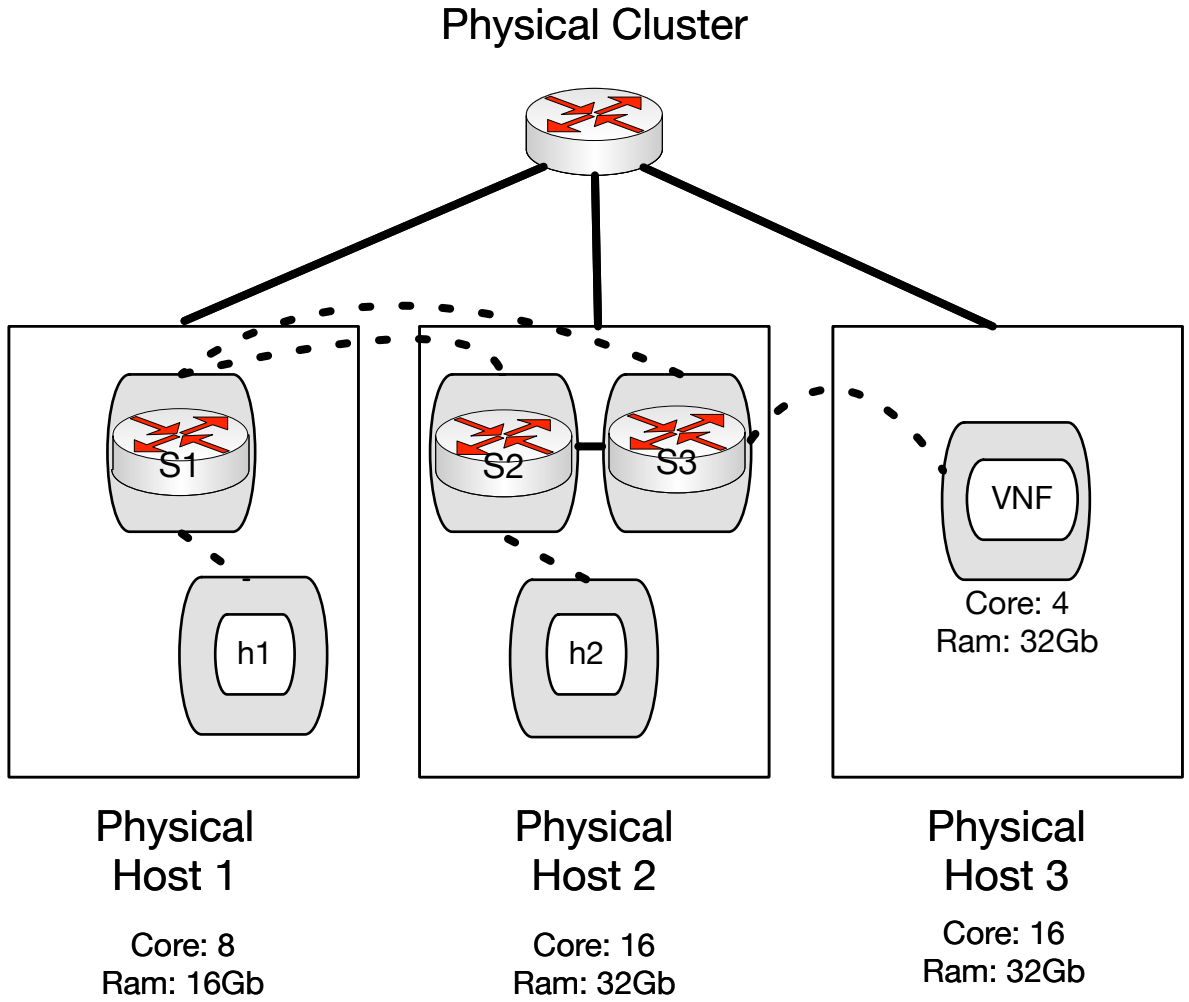
Grid'5000

[1] https://github.com/Giuseppe1992/Distrinet
[2] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html
[3] https://www.grid5000.fr/w/Grid5000:Home
[4] https://www.youtube.com/watch?v=p3-nXUxd-F8&feature=youtu.be
[5] https://www.youtube.com/watch?v=o2bsK_-VPGY&feature=youtu.be

More information

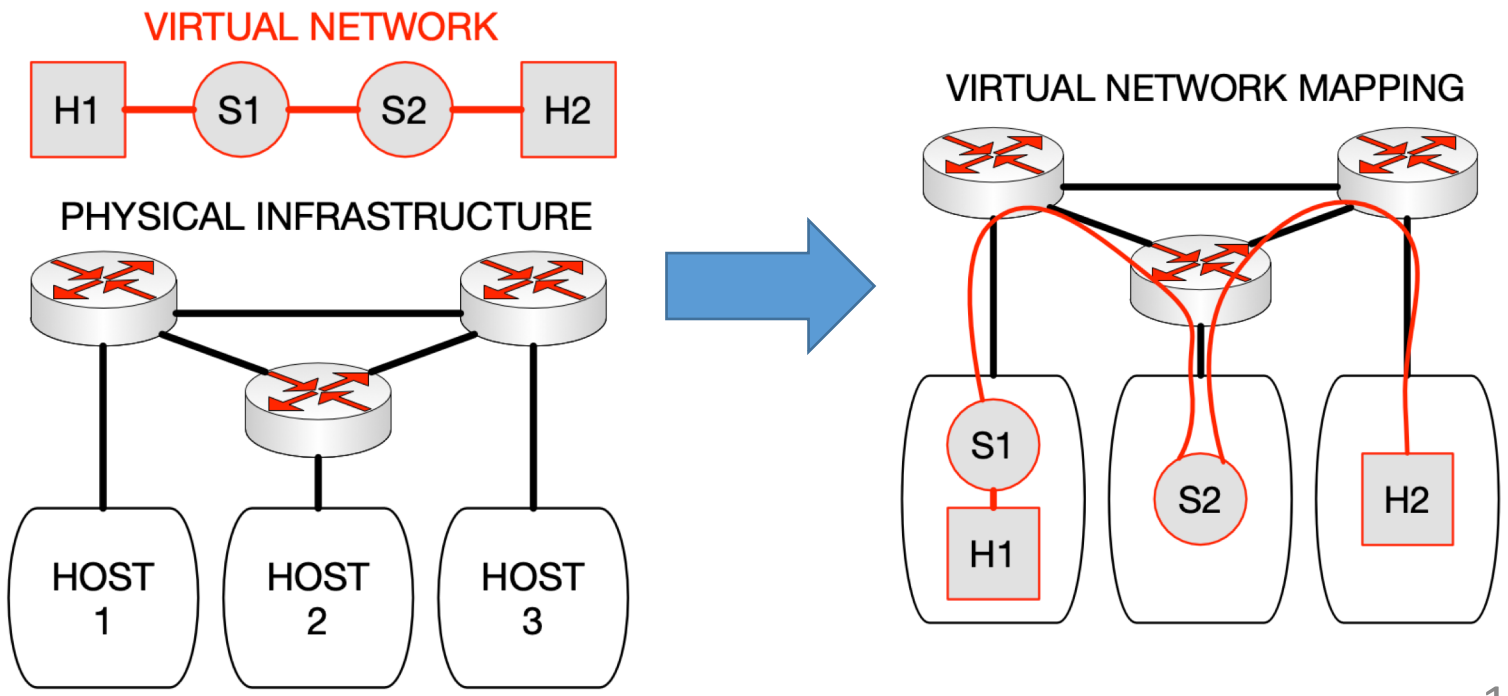
Distributed Network Emulation





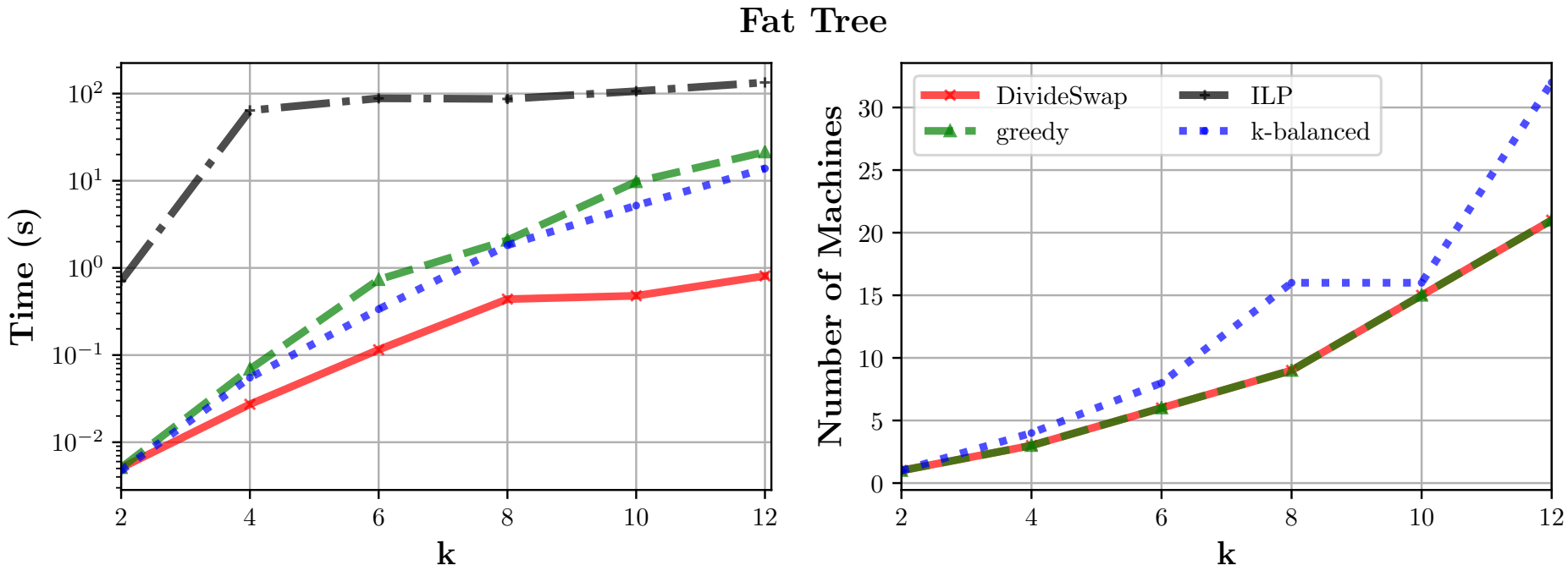
Optimization: Cluster

- **Input**: Virtual Network, Physical Infrastructure
- **Output:** Feasible mapping that minimizes the **# hosts used**



Experiments Fat Tree Topo

- **Physical:** Nancy Cluster[5]
- **Vhost and Vswitch:** 2 Cores and 8Gb Memory, bw=0.2Mbps

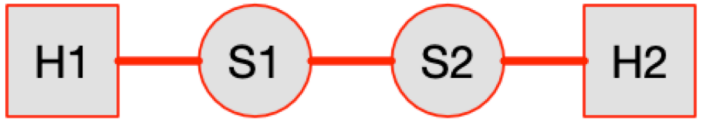


[5] https://www.grid5000.fr/w/Nancy:Hardware
[6] https://github.com/atomassi/mapping_distrinet, A.Tomassilli

Optimization: Cloud

- **Input**: Virtual network, cloud instances details
- **Output:** Feasible mapping that minimizes the **cost** of the experiment.

VIRTUAL NETWORK



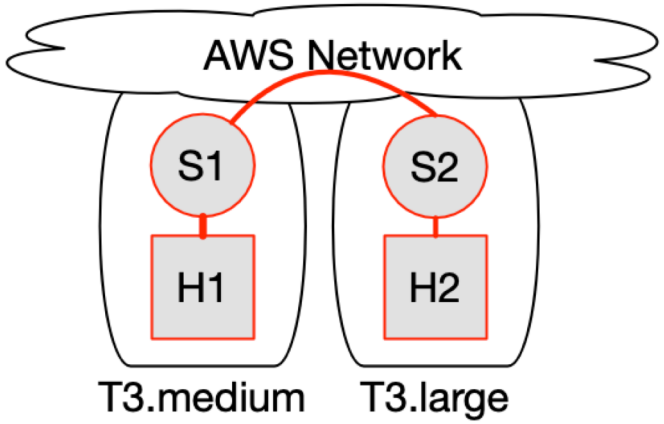
CLOUD INSTANCES DETAILS

NAME	vCpu	Memory	Cost
T3.medium	2	4Gb	0.05 $
T3.large	2	8Gb	0.10 $
T3.xlarge	4	16Gb	0.20 $
...	...	...	...

CLOUD INSTANCES REQUIRED

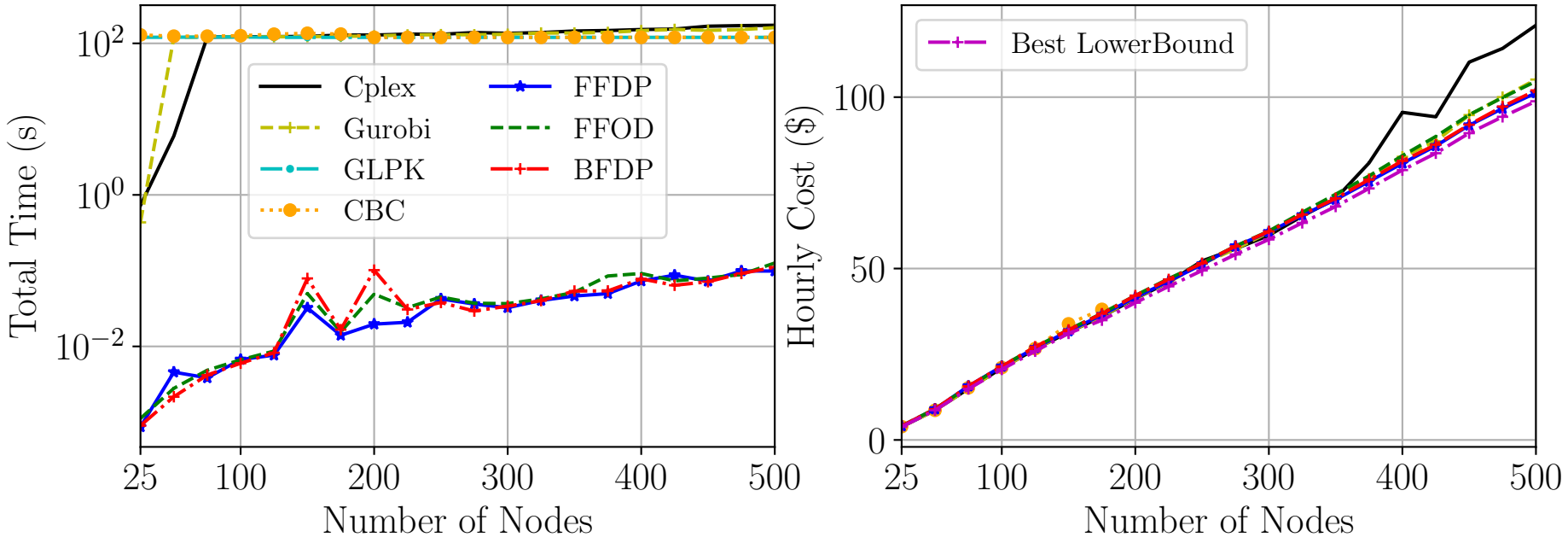
NAME	Quantity
T3.medium	1
T3.large	1

VIRTUAL NETWORK MAPPING



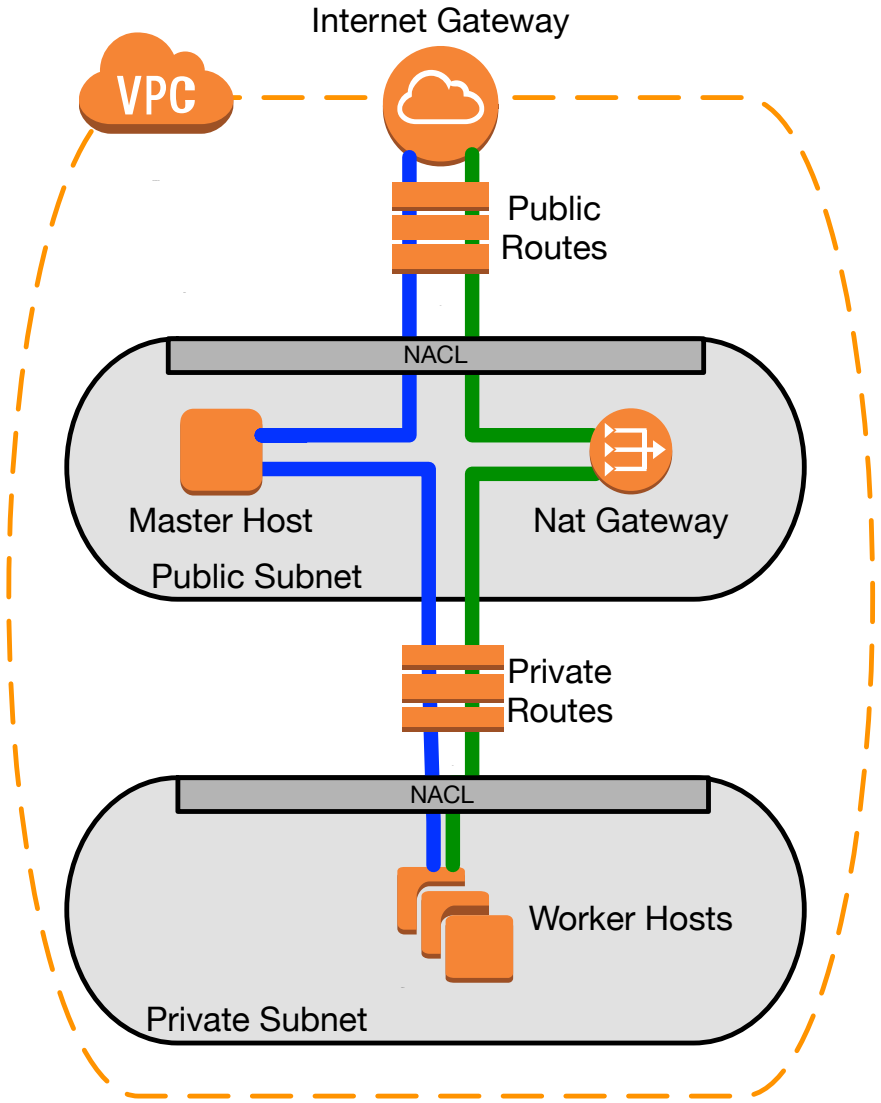
Experiments AWS Random

- **Vhost and Vswitch:** 2 Cores and 8Gb Memory, bw=0.2Mbps
- **Vtopo:** Random



[5] https://github.com/atomassi/mapping_distrinet, A.Tomassilli

AWS Configuration



Amazon AWS limitations:

- Max 5 public IP per region
- No Multicast IPs Allowed

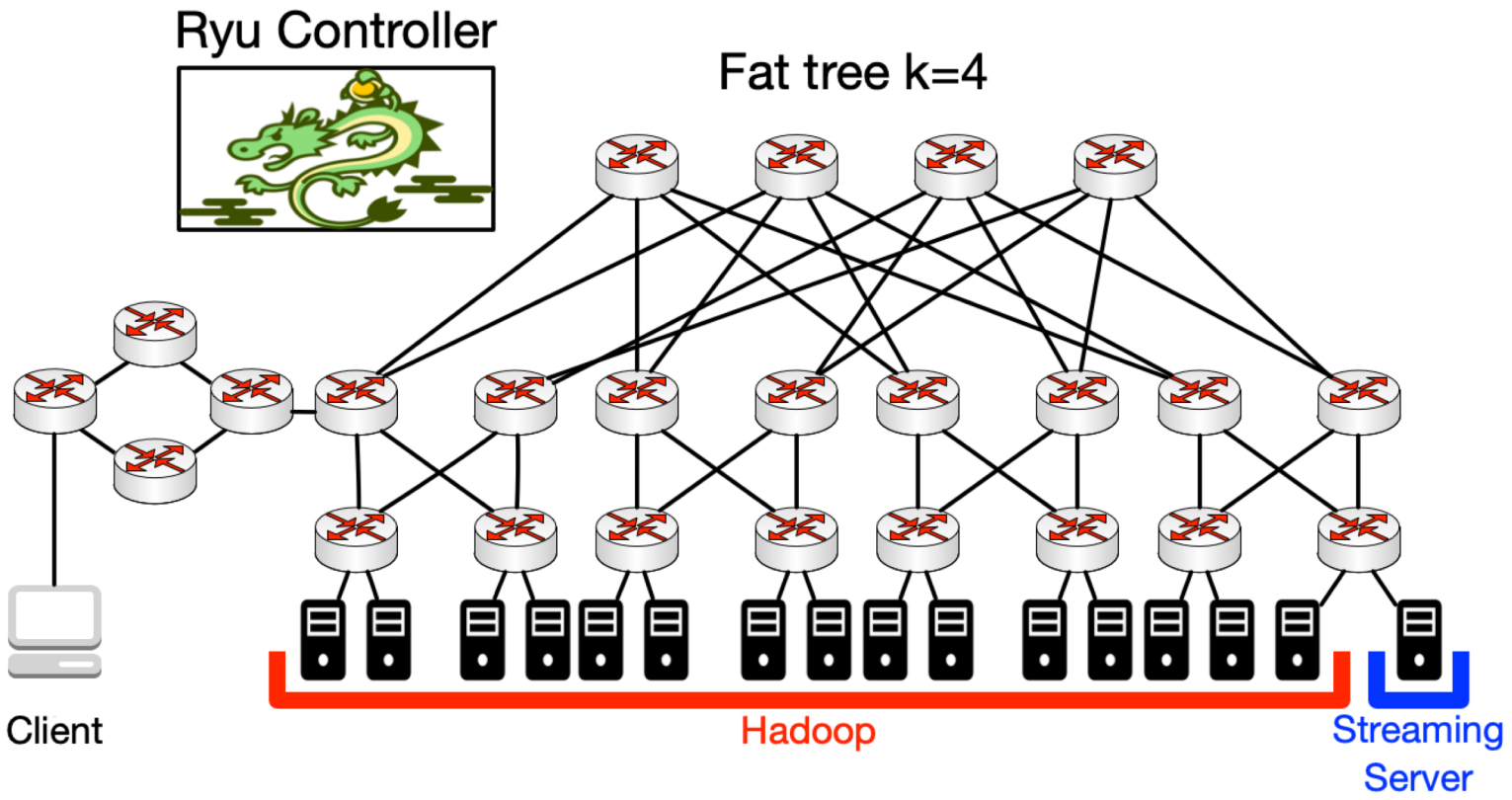
Problem:

- LXD use Multicast VxLan Tunnels

Solution:

- Implement the networking part of LXD using Unicast Tunnels

Example



- Everything is emuleted with 13 Physical Hosts
- Each hosts has:
  - CPU Intel Core i7-2600 processor
  - 8GB RAM
  - 240 GB SSD