GENI-VIOLIN: Distributed Suspend and Resume for GENI Experiments

Ardalan Kangarlou
Sahan Gamage
Dongyan Xu

PURDUE UNIVERSITY

Pradeep Padala
Ulas C. Kozat
Ken Igarashi, Bob Lantz

NTT docomo
DOCOMO USA Labs

Scientists Come up with a Great Idea

I have to run a nano-technology experiment to test my theories

These are my requirements

- The experiment is long-running (hours)
- Requires resources from multiple sites

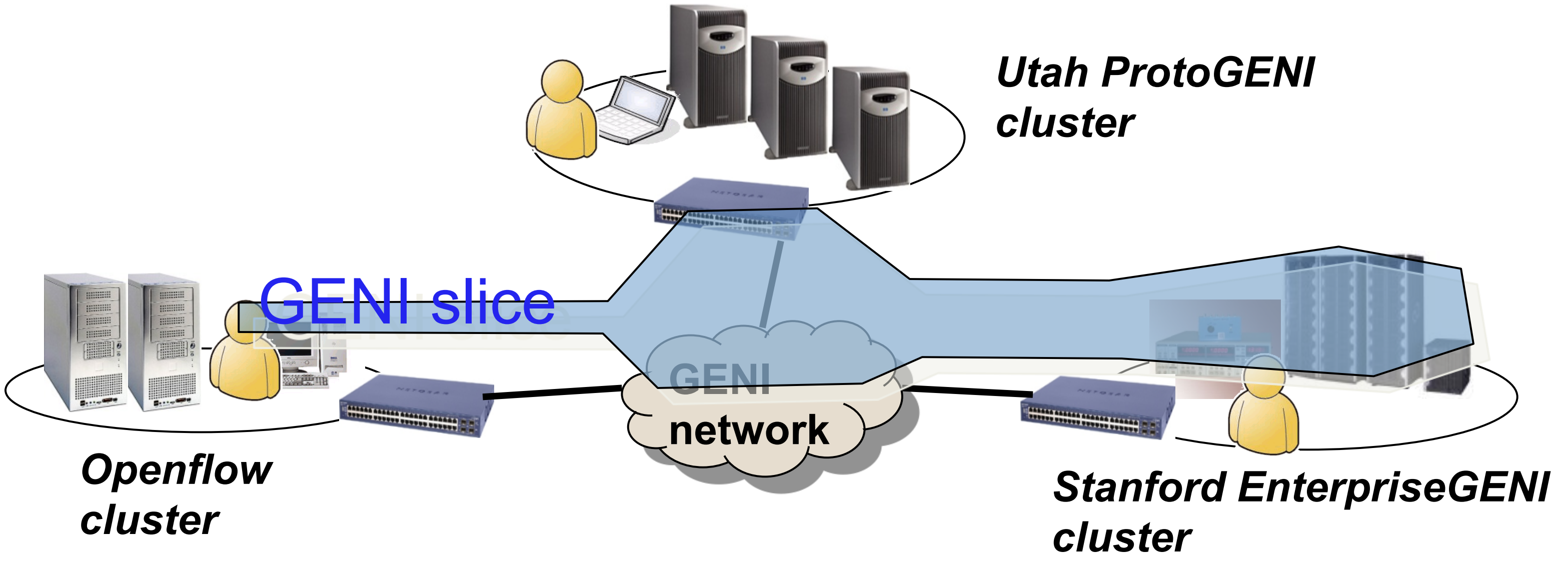
GENI-alpha: We can help!

Scientist builds a VIOLIN

- Has its own IP address space and admin privilege completely decoupled from the physical network domains
- Appears like a single LAN
- Contains VMs that are
  - Customized for specific scientific program execution and data access
  - Created and torn-down on-demand
  - Live-migratable across clusters
- Can be suspended and resumed

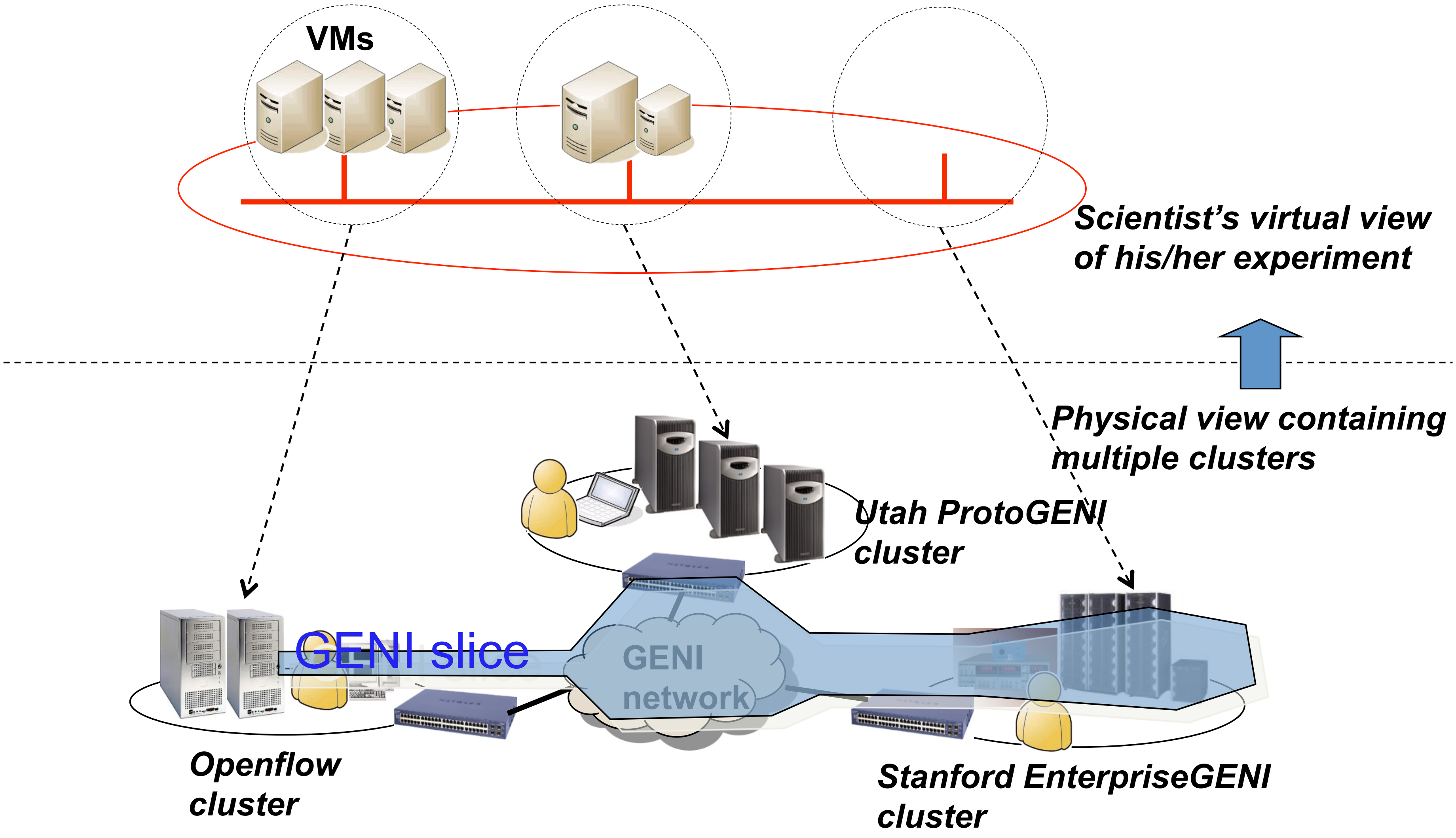
VIOLIN = Distributed Virtual Appliance

Scientist Provisions a Slice of GENI



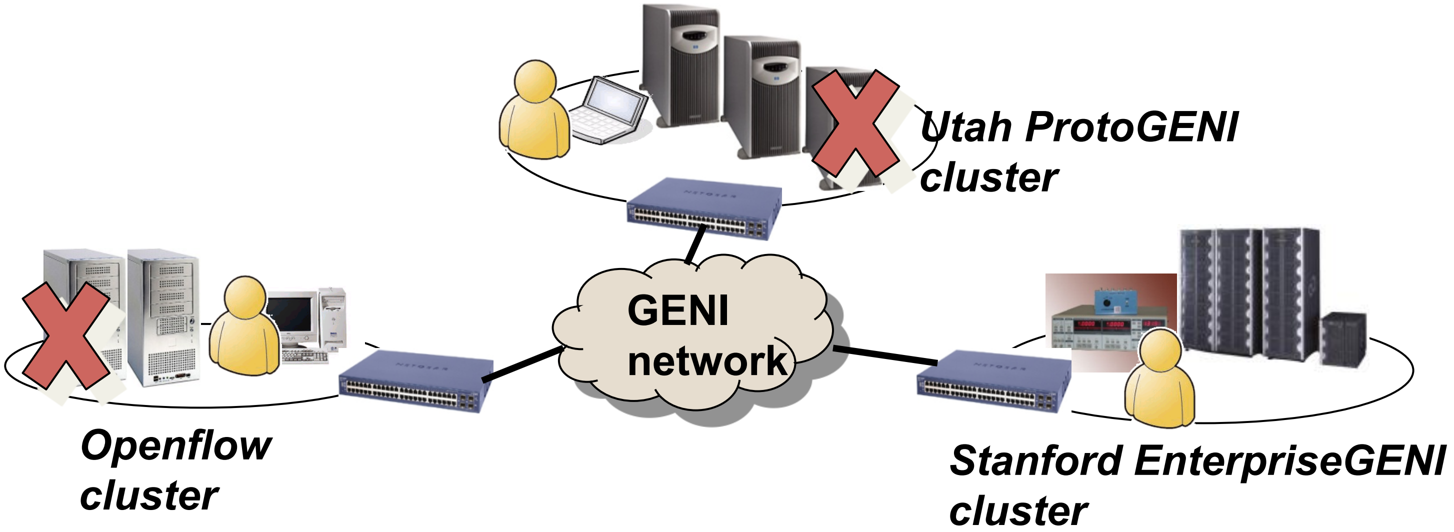
Slice spans multiple cluster sites

Scientist Deploys VIOLIN on a GENI slice


VMs
Scientist's virtual view of his/her experiment
Physical view containing multiple clusters
Utah ProtoGENI cluster
GENI slice
GENI network
Openflow cluster
Stanford EnterpriseGENI cluster

Experiment begins running

Failures Happen in Distributed Environment



Oh, No! Two nodes hosting my VMs failed. I have lost thousands of hours of CPU time ☹

GENI-alpha:

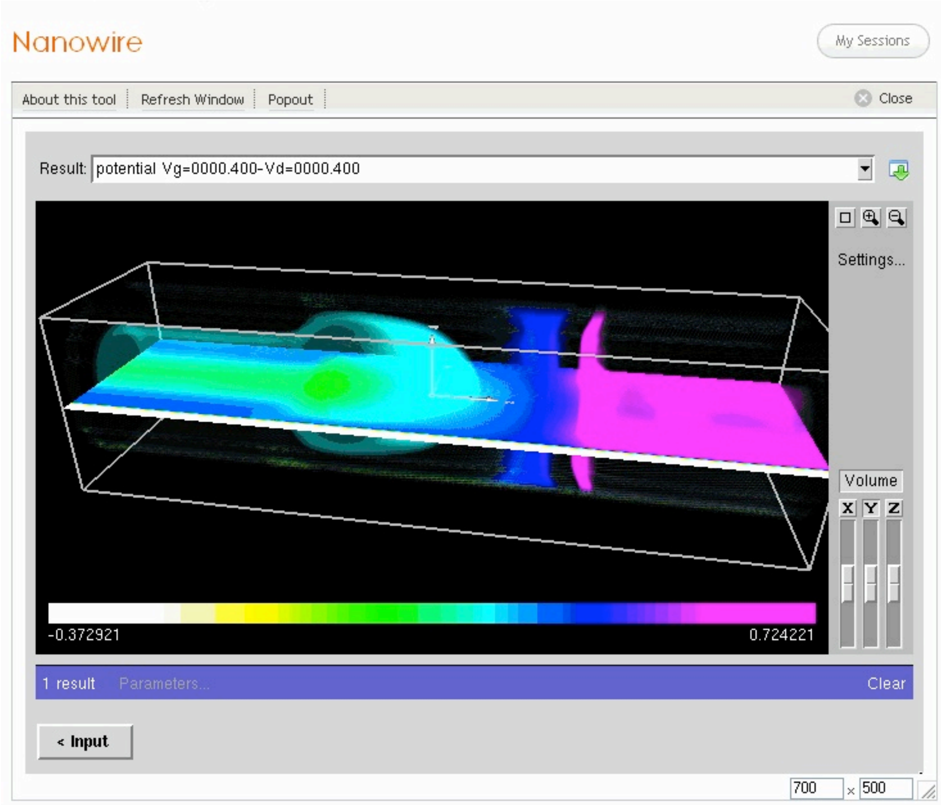
Wait! VIOLIN supports resume!

VIOLIN Resumes the Experiment

VIOLIN's Snapshot/Resume saves the day

Secret Sauce: VIOLIN takes periodic snapshots of entire slices

sees



Result: potential Vg=0000.400-Vd=0000.400

700 × 500

GENI-VIOLIN goals

- Provide "live snapshot" facility to GENI-alpha experiments
  - Fault tolerance
  - Debugging
  - Slice management
- Minimal disruption to application performance
- Transparent to the applications and guest OSs
- Non-stop execution of the application

GENI-VIOLIN: Distributed Suspend and Resume for GENI experiments

GENI-VIOLIN status

- VIOLIN is ported to Emulab environment
- All Emulab experiments can use VIOLIN now!
- Current VIOLIN uses UDP tunneling and a few other tricks to create a single virtual L2 network

- Openflow implementation is in progress that provides same features with better performance
- GENI-alpha/GEC9: VIOLIN + Openflow on ProtoGENI
- Snapshotting entirely in the network, no end-host support other than hypervisor required

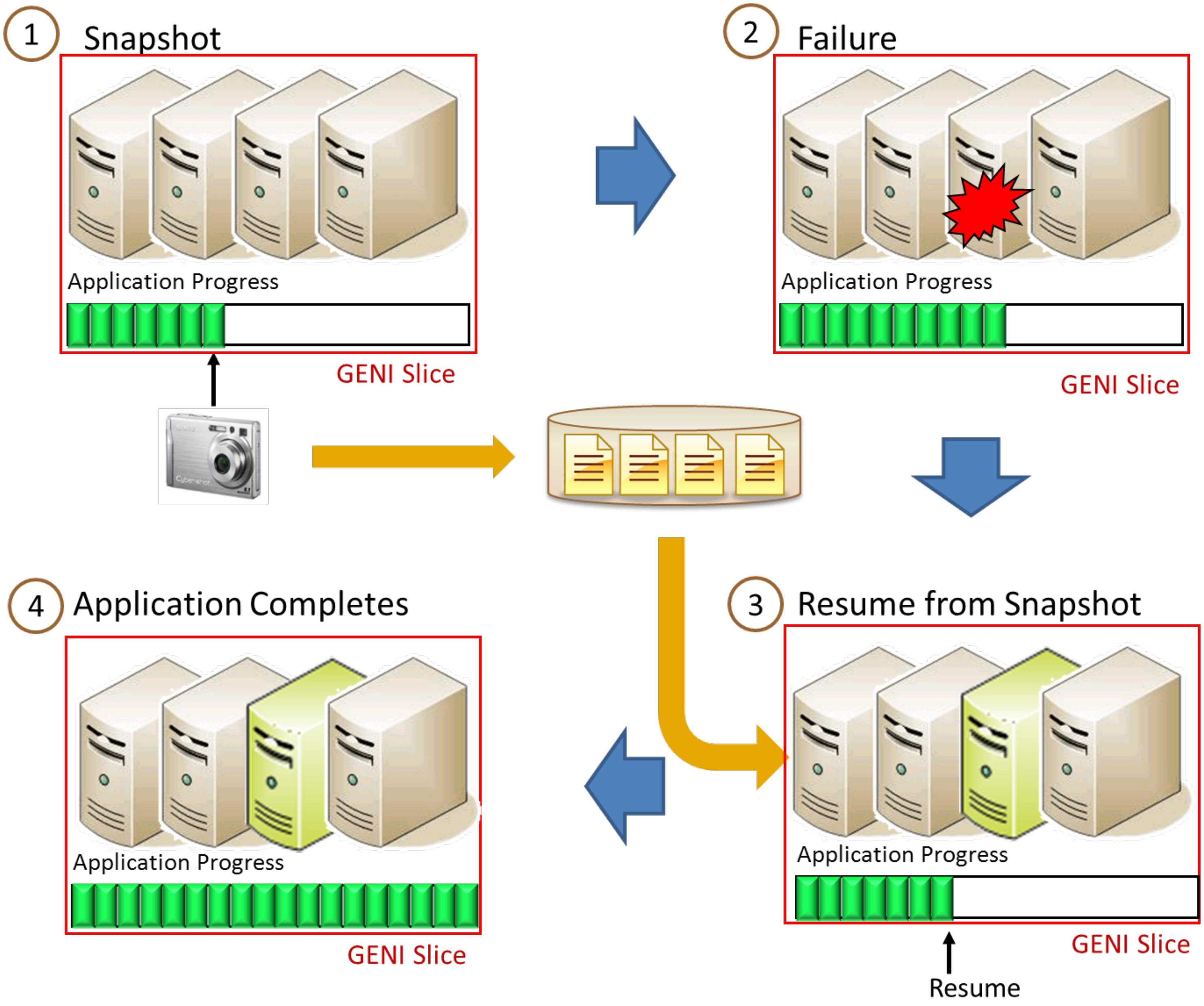
GENI-VIOLIN GEC8 demo

Fault-tolerance to distributed GENI experiments

Challenge

How to do distributed suspend/resume?

Demo scenario: Recover from failures



1 Snapshot

Application Progress

GENI Slice

2 Failure

Application Progress

GENI Slice

3 Resume from Snapshot

Application Progress

GENI Slice

Resume

4 Application Completes

Application Progress

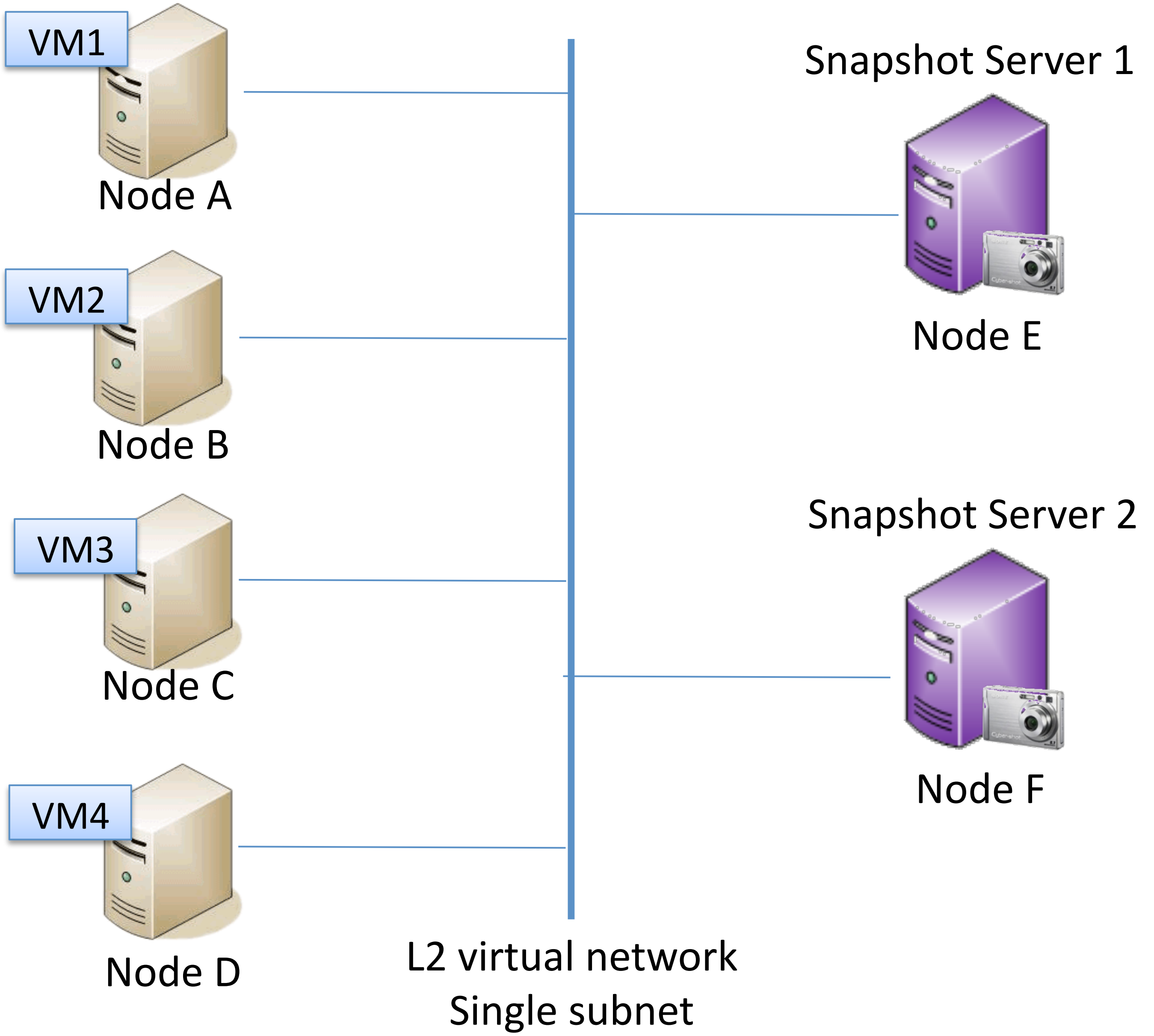
GENI Slice

Emulab script

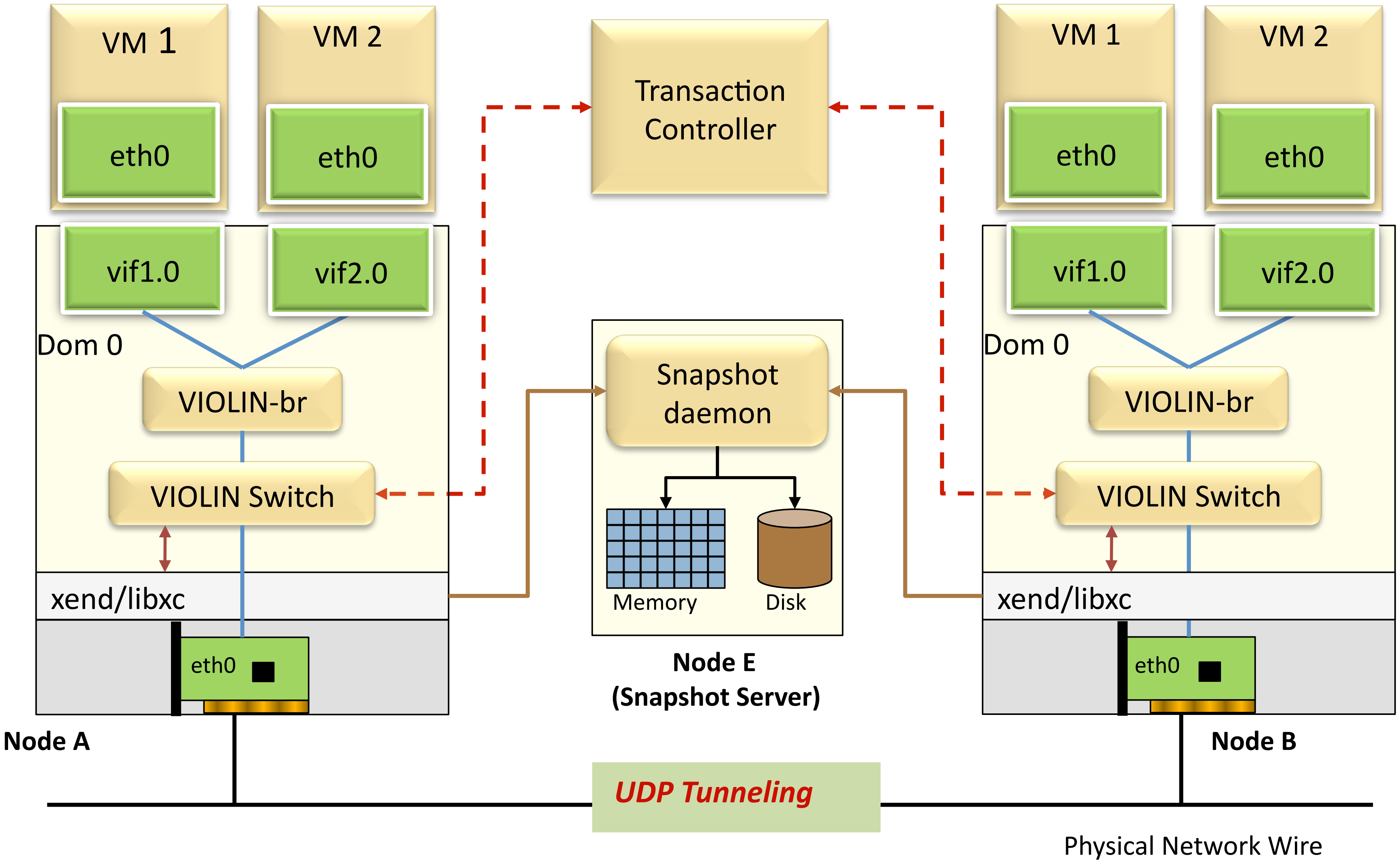
```
set ns [new Simulator]
source tb_compat.tcl

set nodeA [$ns node]
set nodeB [$ns node]
set nodeC [$ns node]
set nodeD [$ns node]
set nodeE [$ns node]
set nodeF [$ns node]

set lan0 [$ns make-lan "$nodeA $nodeB $nodeC $nodeD $nodeE $nodeF" 1000Mb 0ms]

tb-set-node-os $nodeA VIOLIN
tb-set-node-os $nodeB VIOLIN
tb-set-node-os $nodeC VIOLIN
tb-set-node-os $nodeD VIOLIN
tb-set-node-os $nodeE VIOLIN
tb-set-node-os $nodeF VIOLIN

$ns run
```

Our customized
Xen + Linux image

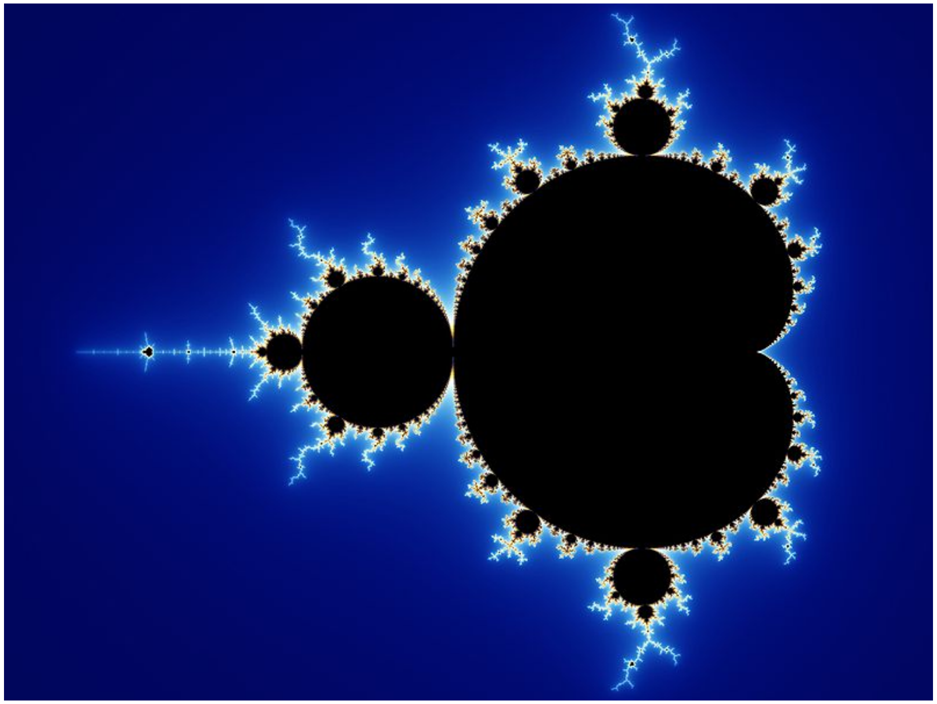
Demo setup: 4 VMs and 2 snapshot servers



Under the hood: VIOLIN



Demo application: Distributed Mandelbrot



```
For each pixel on the screen do
{
  while(x*x + y*y <= (2*2) AND
      iteration < max_iteration) {
    xtemp = x*x - y*y + x0
    y = 2*x*y + y0
    x = xtemp
    iteration = iteration + 1
  }

  if (iteration == max_iteration)
    color = black
  else
    color = iteration
  plot(x0,y0,color)
}
```

- Color of pixel needs to be calculated
- Distributed MPI processes

Demo