CoreLab and VNode

Approach of Cluster Japan ☺

Aki Nakao
University of Tokyo
2010/7/21

Goals of This Talk

- Introducing our (similar) research efforts
  - GENI is cool, but we also do cool stuff ☺
- Seeking opportunities for collaboration
  - Especially on Federation etc.

Net-Virt Infrastructure for diverse NWs

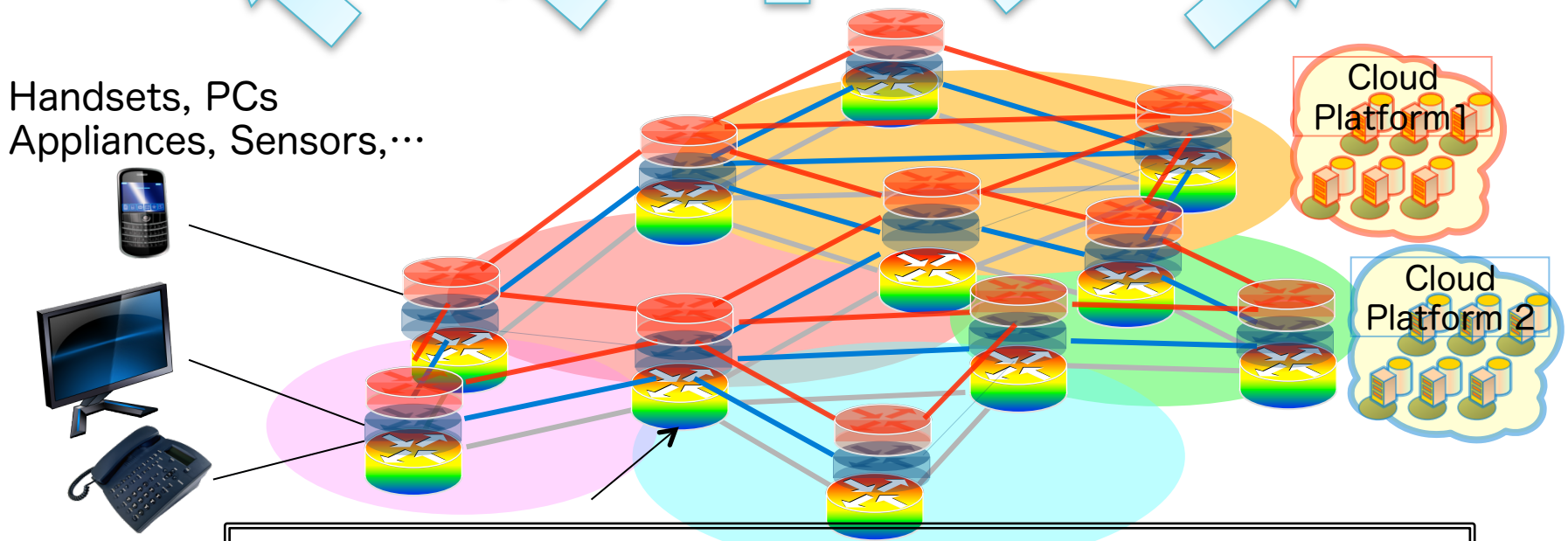
Slice 1 Slice 2 Slice N

Cache Oriented NW | Sensor Processing NW | Cloud Access NW | ID/Loc NW | Content Oriented NW | TV Broadcast NW | Legacy NW | Future NW

Slices accommodate diverse NWs

Slicing Resources (CPU Cycles, Network, Storage, etc)

Handsets, PCs
Appliances, Sensors,…



Network Virtualization Infrastructure

Network Virtualization Research in Japan

✧NV Research Lab (UTokyo+NICT)

We are trying to understand “Network Virtualization” Infrastructure

✧CoreLab (UTokyo+NICT)

✧Enabling net-virt via only S/W on COTS x86 machines

✧VNode (UTokyo+NICT+NTT+NEC+Hitachi+Fujitsu)

✧Enabling net-virt via designing H/W based on production routers

✧Applications

✧In Network Processing

✧Non-IP Protocols

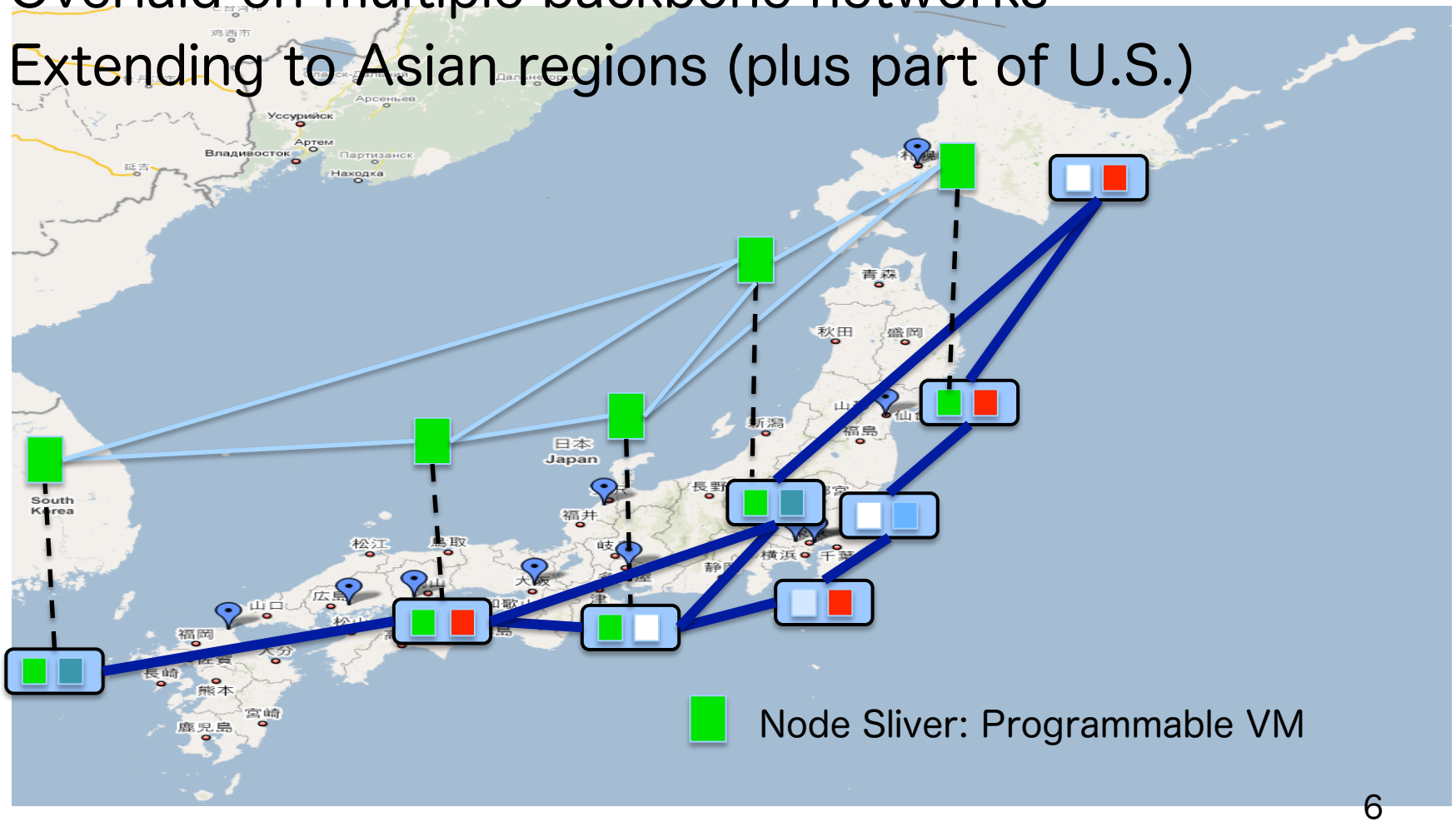
CoreLab

(UTokyo & NICT)

Enabling net-virt via only S/W on COTS x86 machines

CoreLab

- Shares DNA with PlanetLab (w/ more Net Virt Support)
- Creates slices across Japan (24+ nodes)
- Overlaid on multiple backbone networks
- Extending to Asian regions (plus part of U.S.)



CoreLab Highlights 1/2

1. Flexible Virtualization

Host-based Virtualization (KVM)
Resource Container (OpenVZ, LXC)

2. FlowSpace Isolation

GRE-tap Tunnels between Slivers

Classify traffic from/to slices via OpenFlow Switch

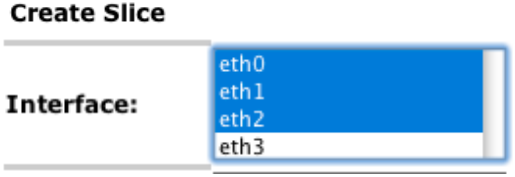
3. Switch Emulation

DEMO

OpenFlow Switch In A Slice (OFIAS)

4. Multi-Homing

Support multi-homing via multiple physical NICs



CoreLab Highlights 2/2

5. VLAN Support

Assign any number of VLANs to a slice

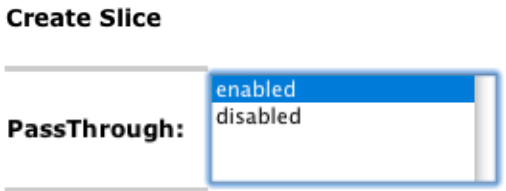
6. DCN Integration

Isolate bandwidth through tagged VLAN and DCN capable L2 switches

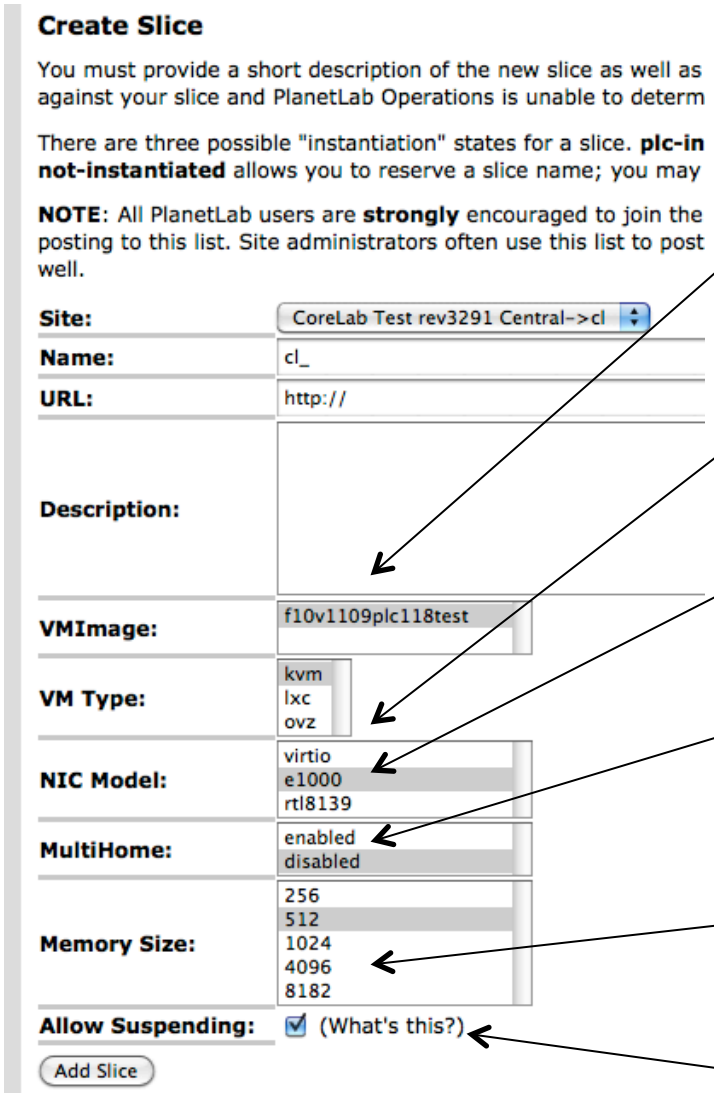
DEMO

7. Guest-Host Pass-Through

Support PCI NIC Pass-Through (both Wired and Wireless)
Dedicate 10Gbps NIC to a sliver



Highlight1 : Flexible Virtualization



VM Image
Arbitrary OS images registered

VM Type
KVM, LXC, OVZ, and more coming

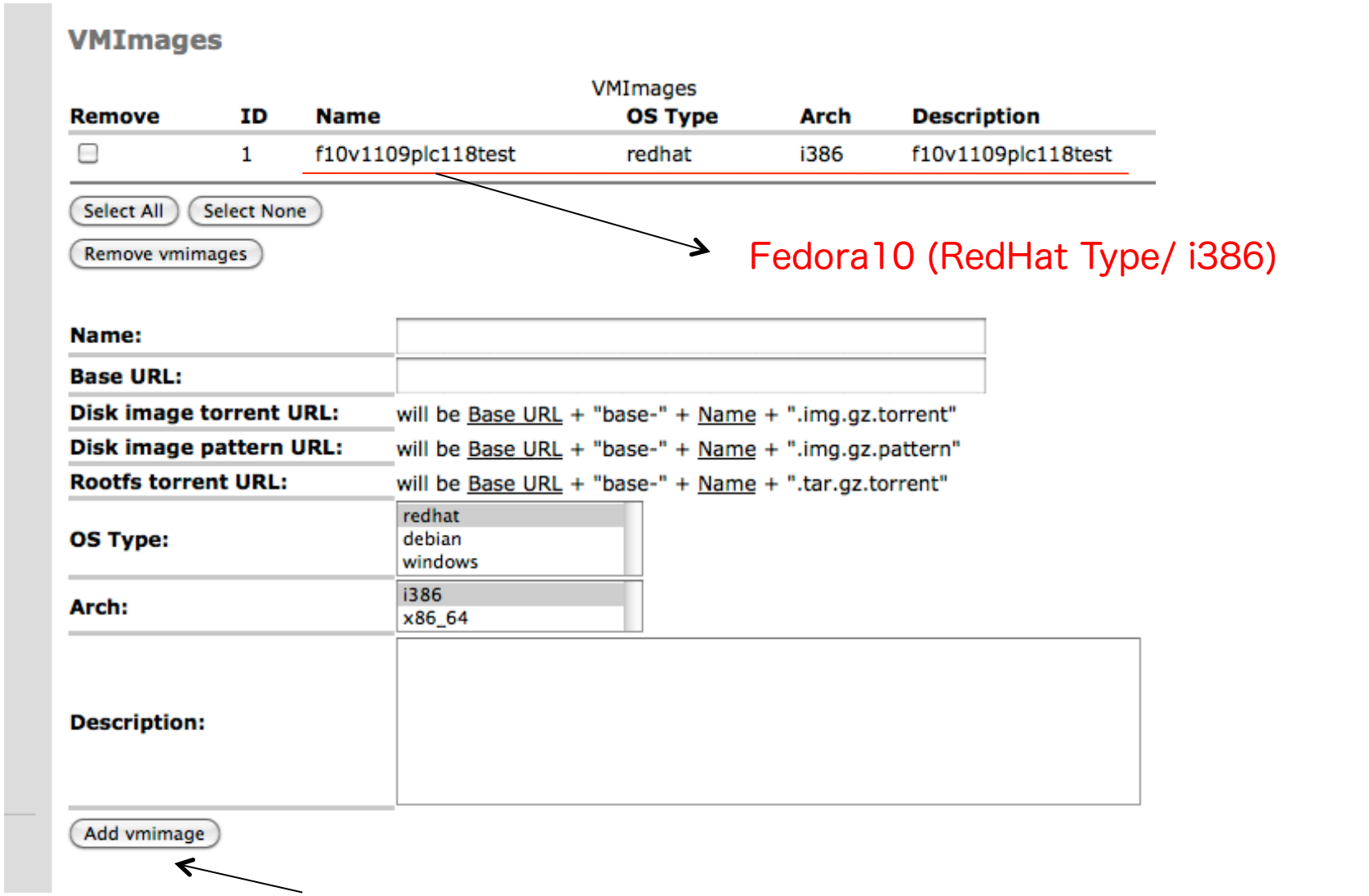
NIC Model
Vanilla, Para-Virtualized Drivers

Multihome
SINET, JGN2Plus, or both

Memory Size
256, 512, 1024, 4096, 8192 MB

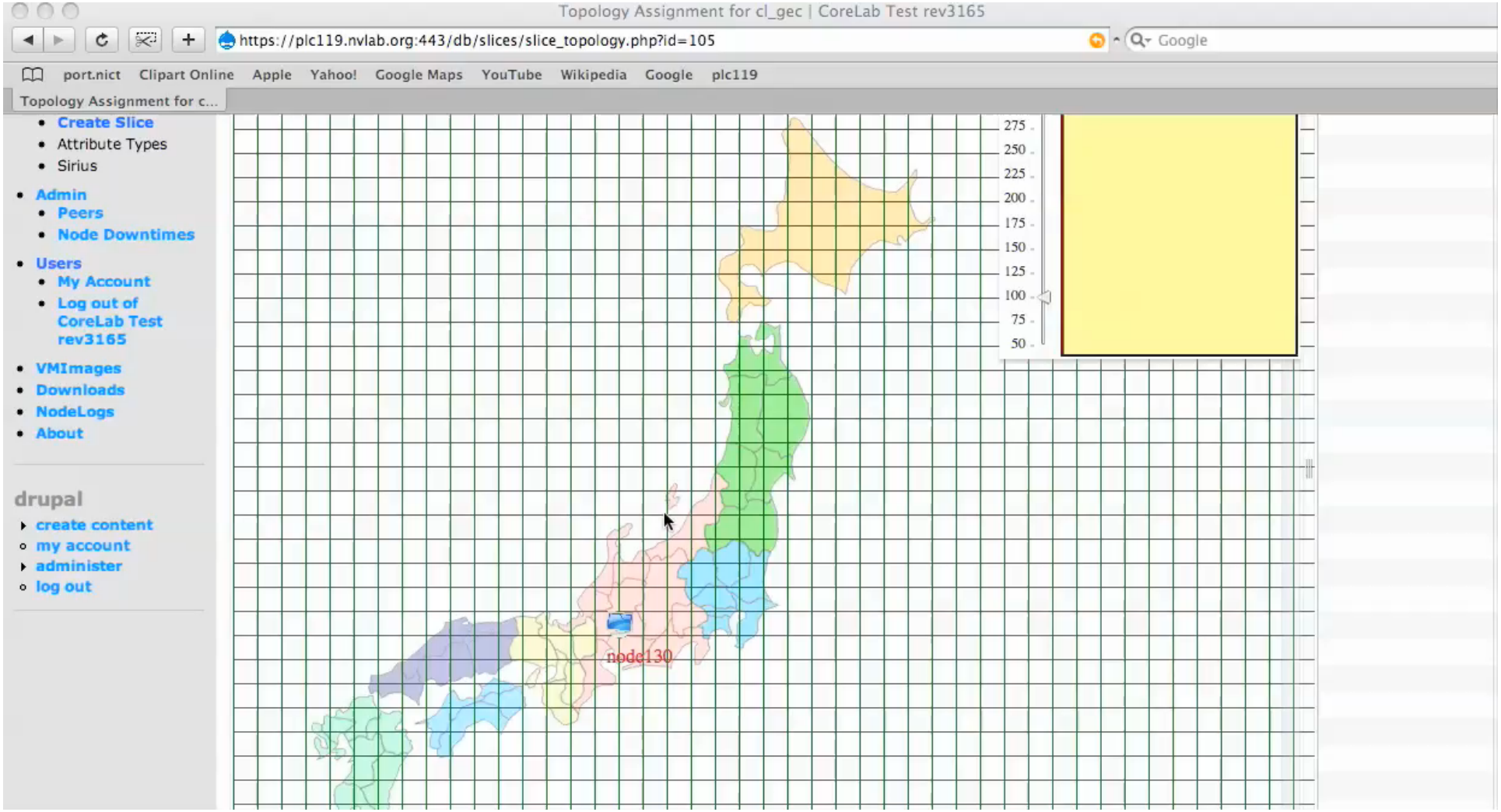
Suspend Feature
Suspend the slice when idol

Registering/Selecting VM Images

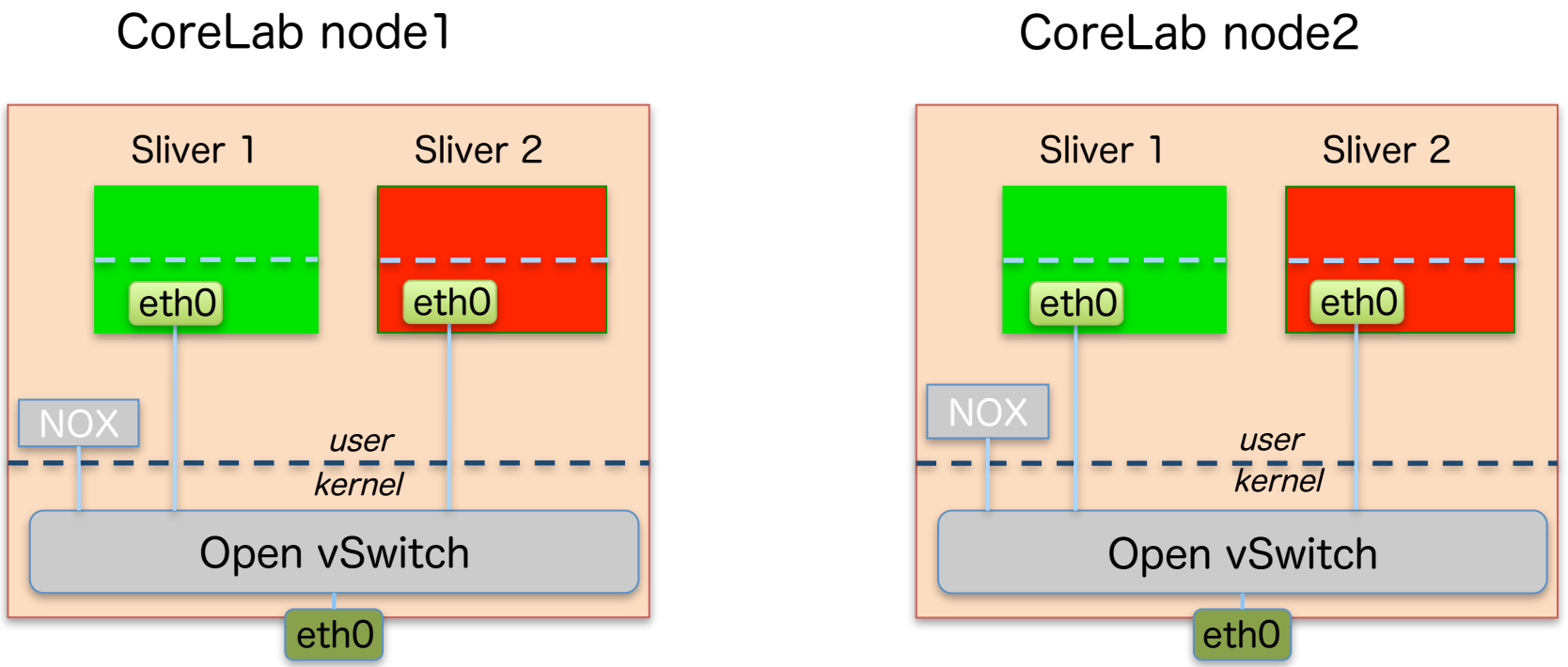


Graphical Topology Management

- GUI for editing a slice topology
- Web-based VNC to login a slice



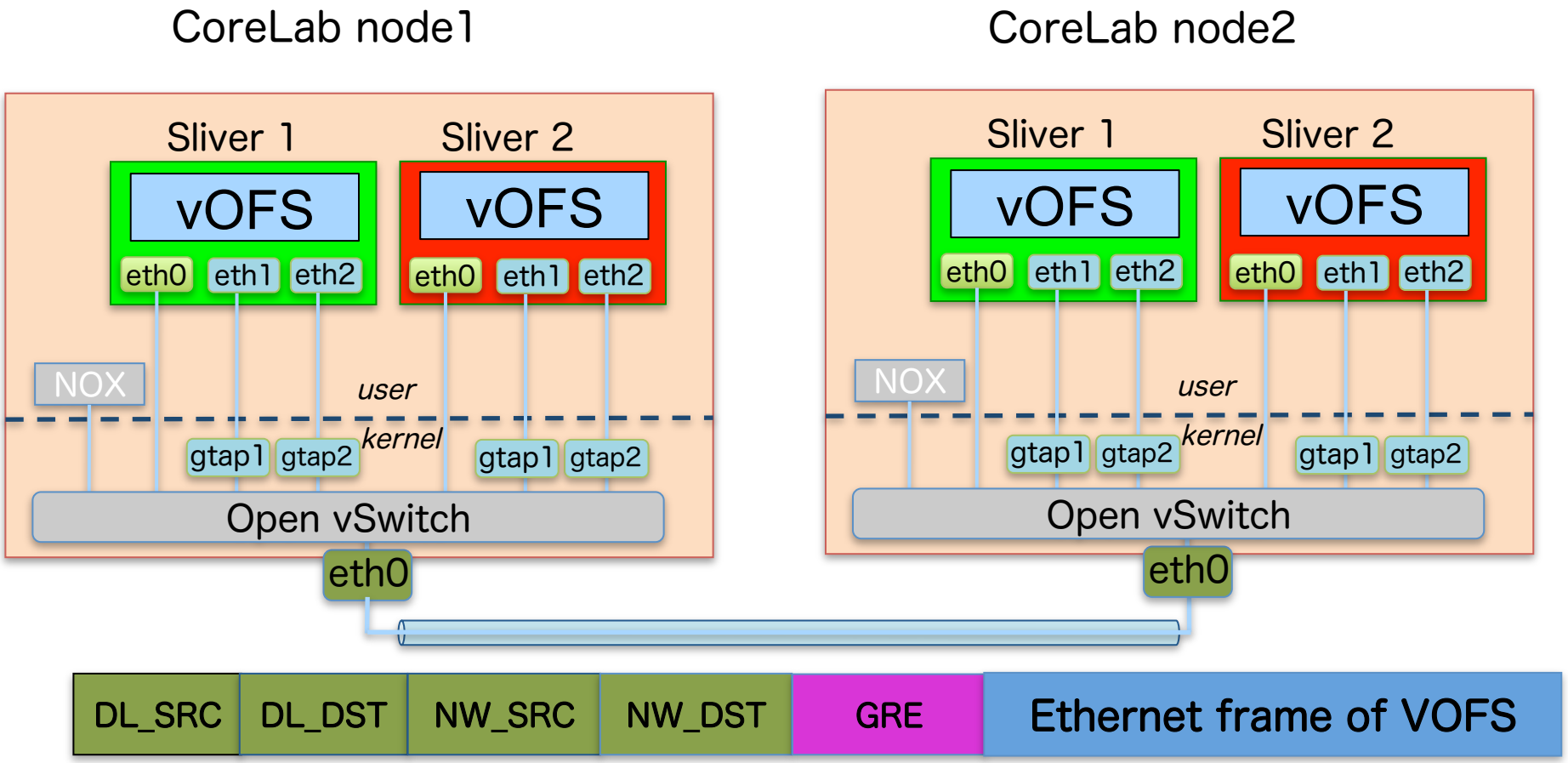
Highlight 2 : FlowSpace Isolation



- Classify traffic from/to slices via OpenFlow Switch
- Sliver1 and Sliver2 (guests) may have the same IP Address as Host
- In general, we can assign “flowspace” per slice.

Port-space isolation with Open vSwitch [Tridentcom2010]

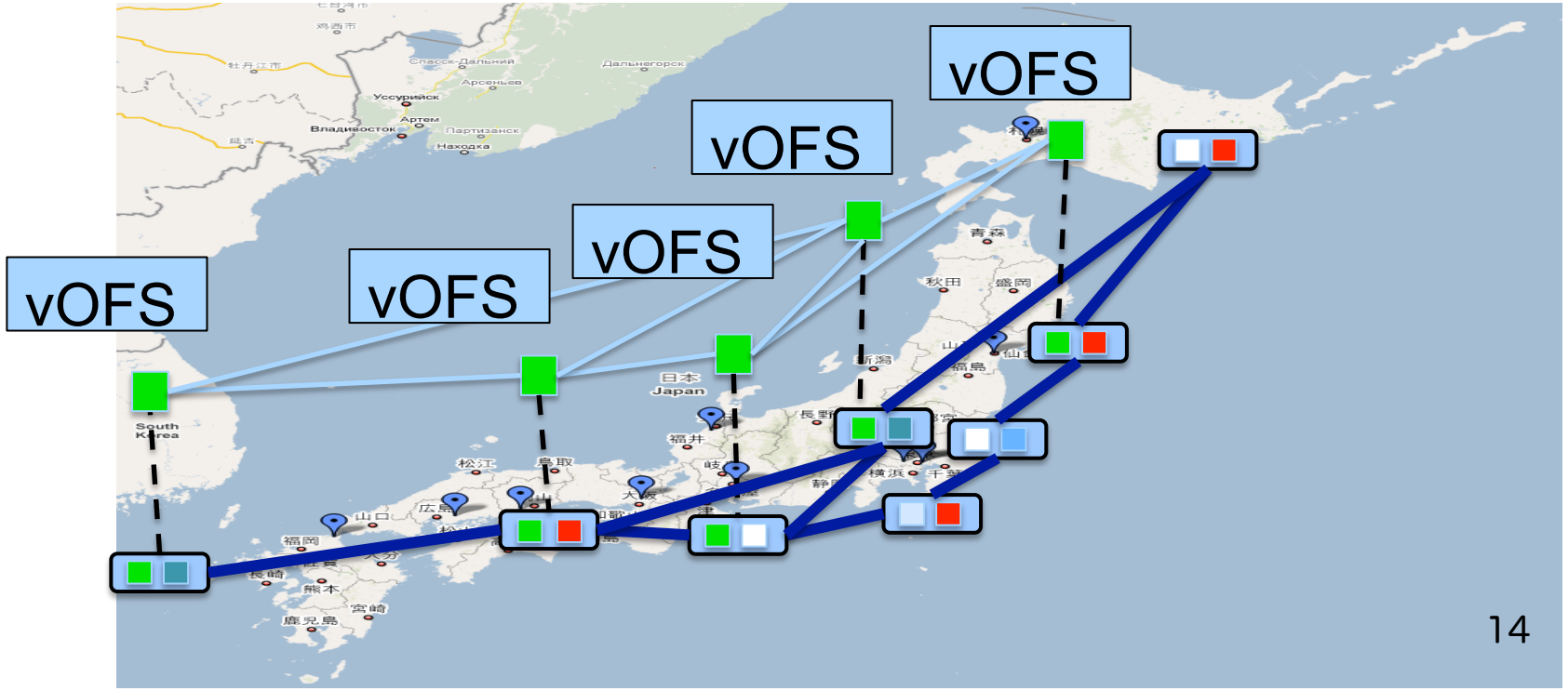
Highlight 3 : Switch Emulation



- Establish L2 GRE-TAP tunnels
- Install virtual OpenFlow swiches (vOFS's) in each sliver

OpenFlow In A Slice 1/2

- Benefit
  - Allow a slice to experiment with OpenFlow networking
- Ingredients
  - Slices created on CoreLab
  - vOFS: Virtual OpenFlow Switch within a sliver

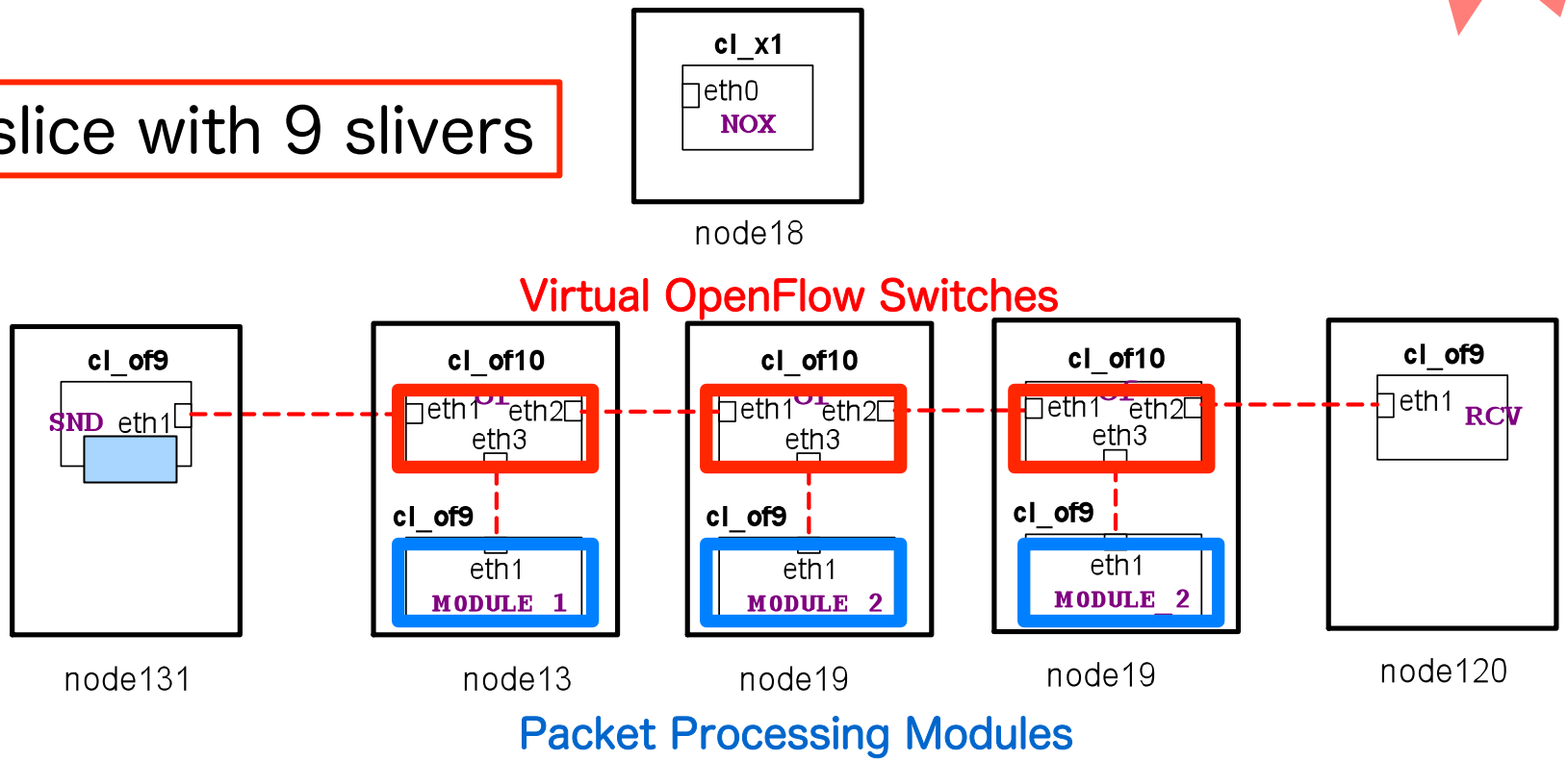


Openflow In A Slice 2/2

DEMO: OpenPipes In A Slice (OPIAS)

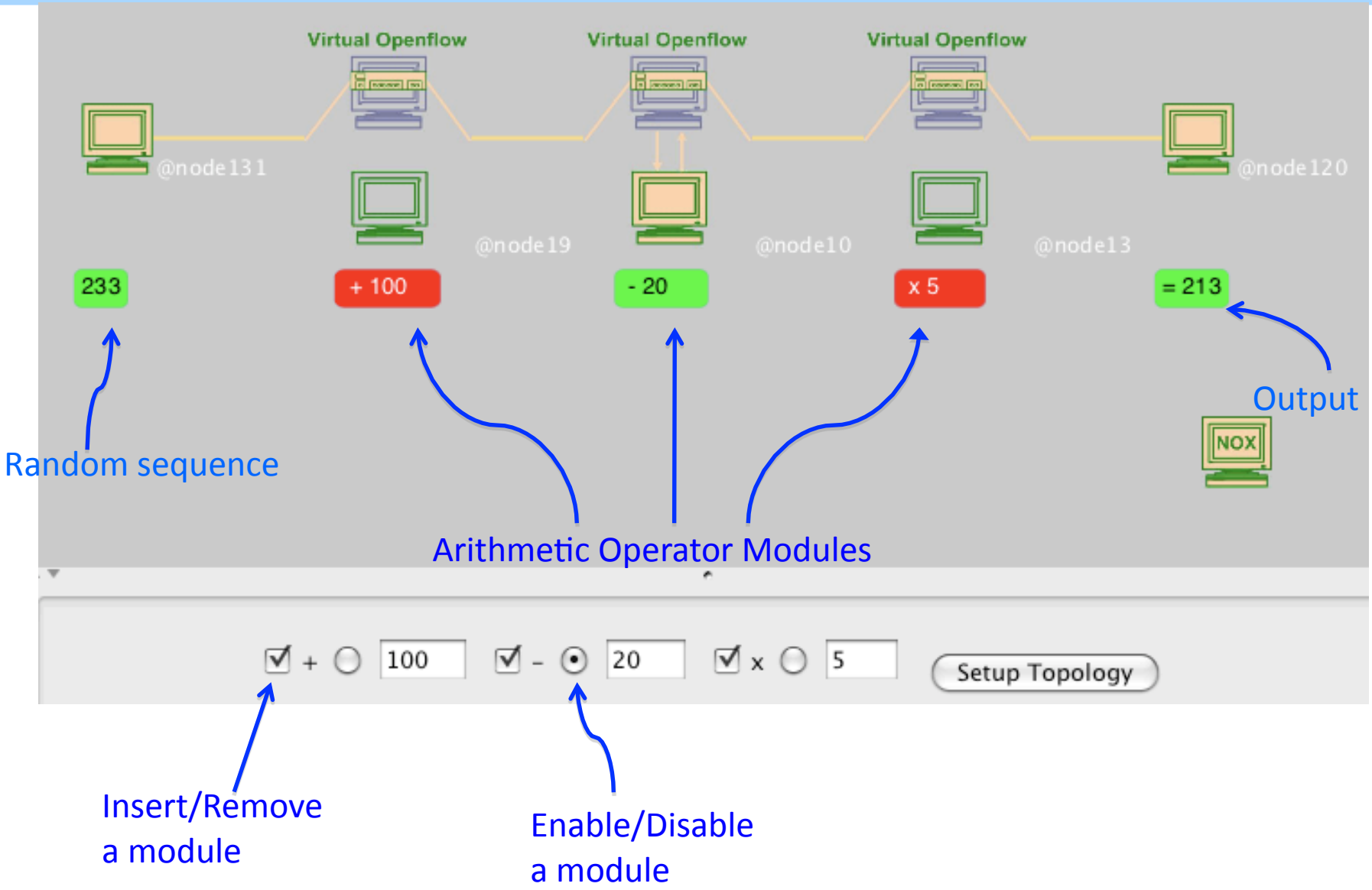
DEMO

A slice with 9 slivers

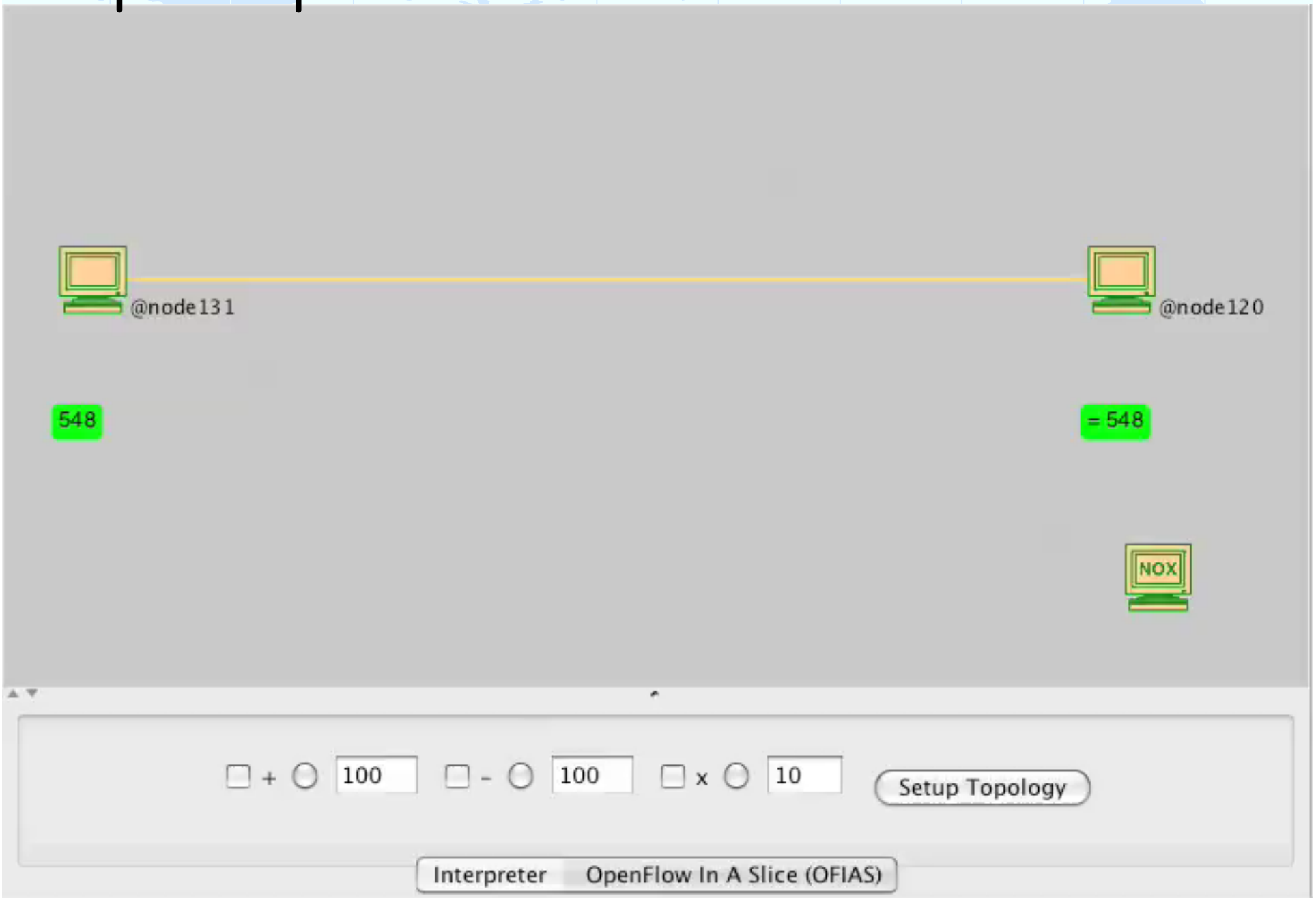


- Chaining Packet Processing Modules Along Data Path
- Virtualize the original OpenPipes demo at SIGCOMM 2009

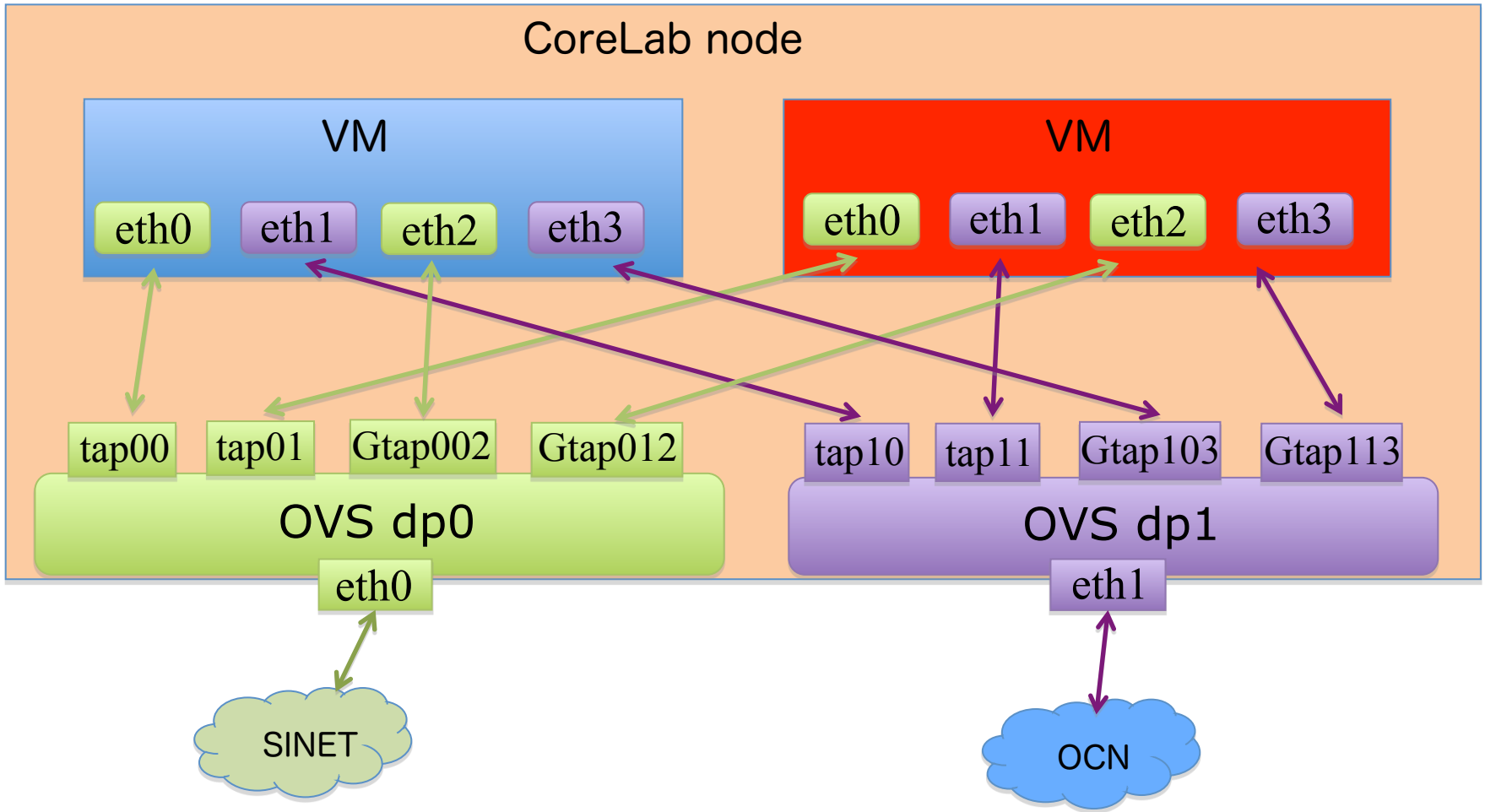
OpenPipes Demo Screenshot



OpenPipes Demo

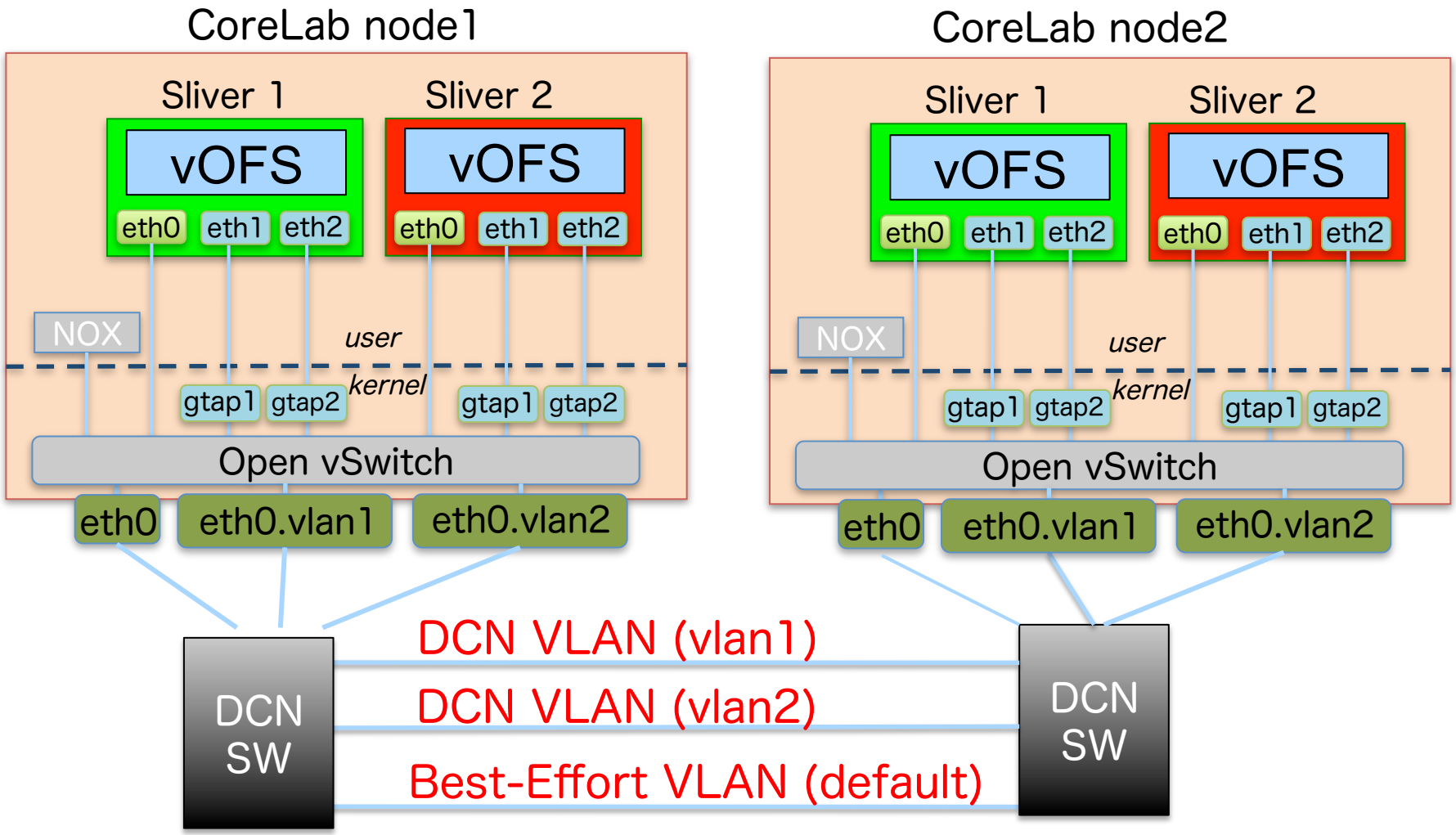


Highlight 4 : Multi-homing L2 Tunnels

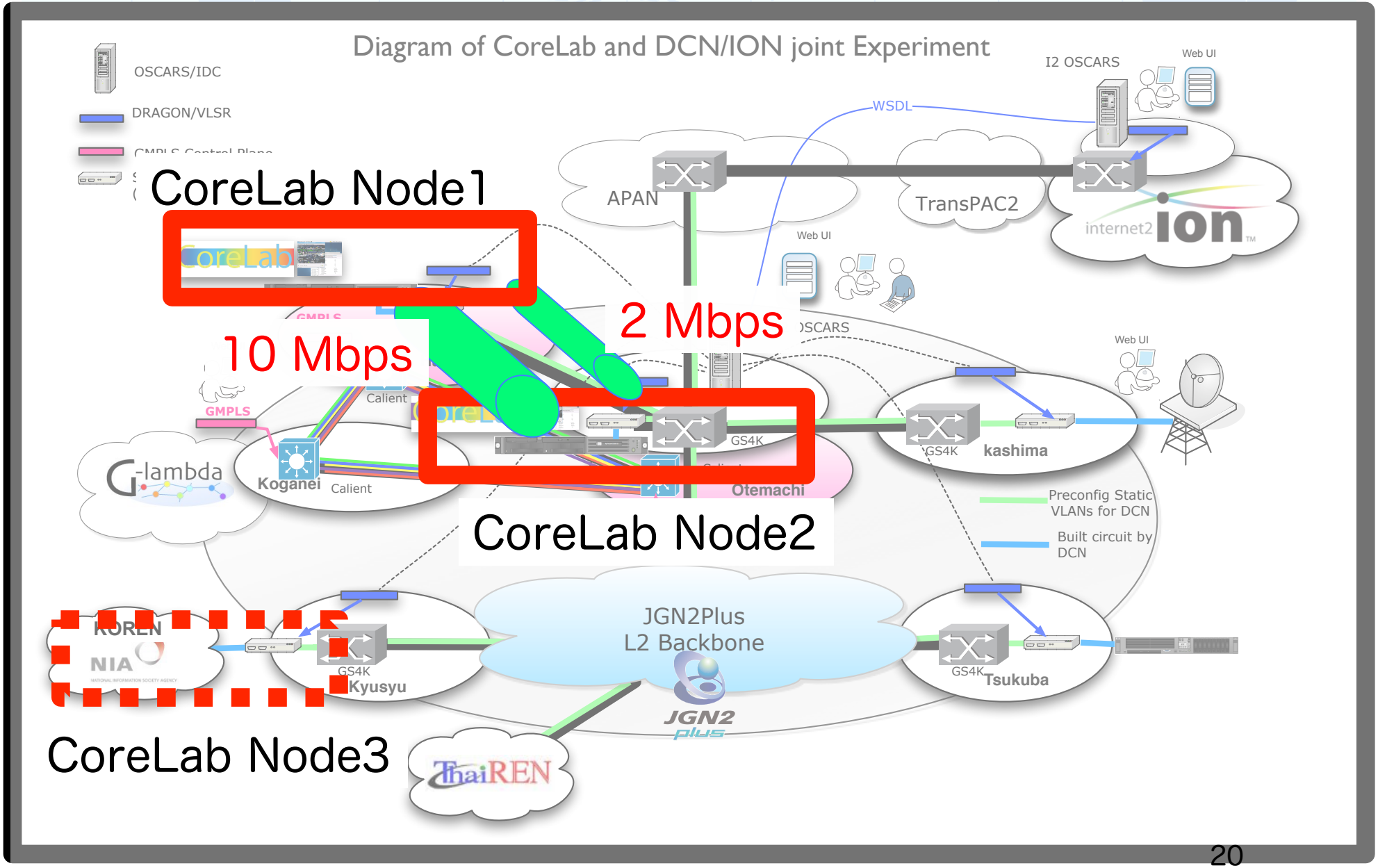


- Virtual Interface eth0 shares the global IP address with Physical interface eth0 (SINET)
- Virtual Interface eth1 shares the global IP address with Physical interface eth1 (OCN)
- Virtual Interfaces eth2, eth4, eth${2n} attach to Ethernet tunnels through SINET.
- Virtual interfaces eth3, eth5, eth${2n+1} attach to Ethernet tunnels through OCN.

Highlight 5&6: DCN+VLAN Integration



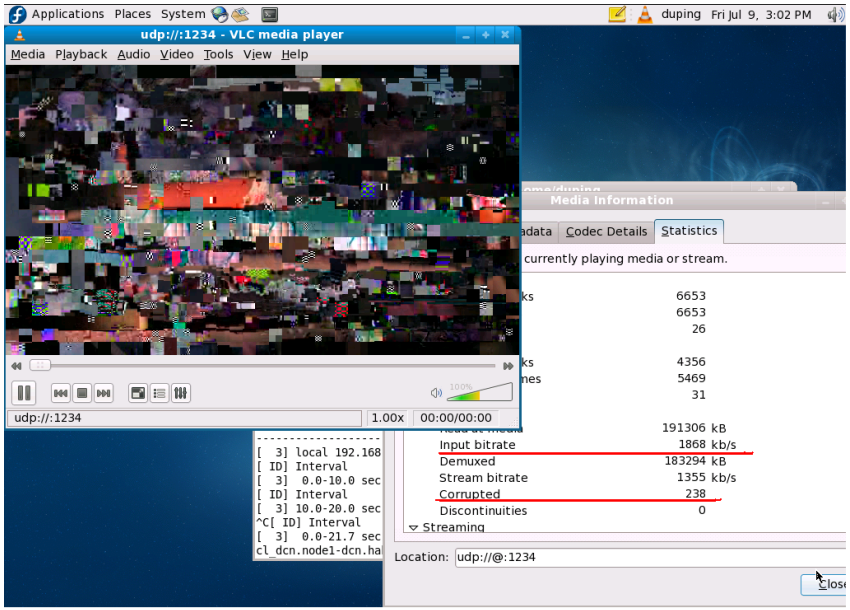
Creating Two DCN paths: 2Mbps/10Mbps



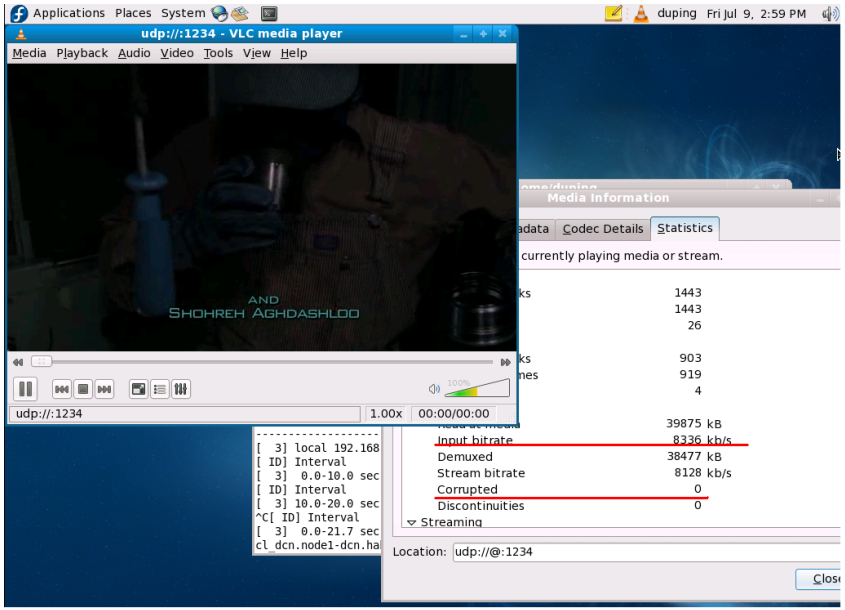
Streaming Video b/w Two CoreLab slivers

Screenshots at Hakusan node through VNC :

Poor Video



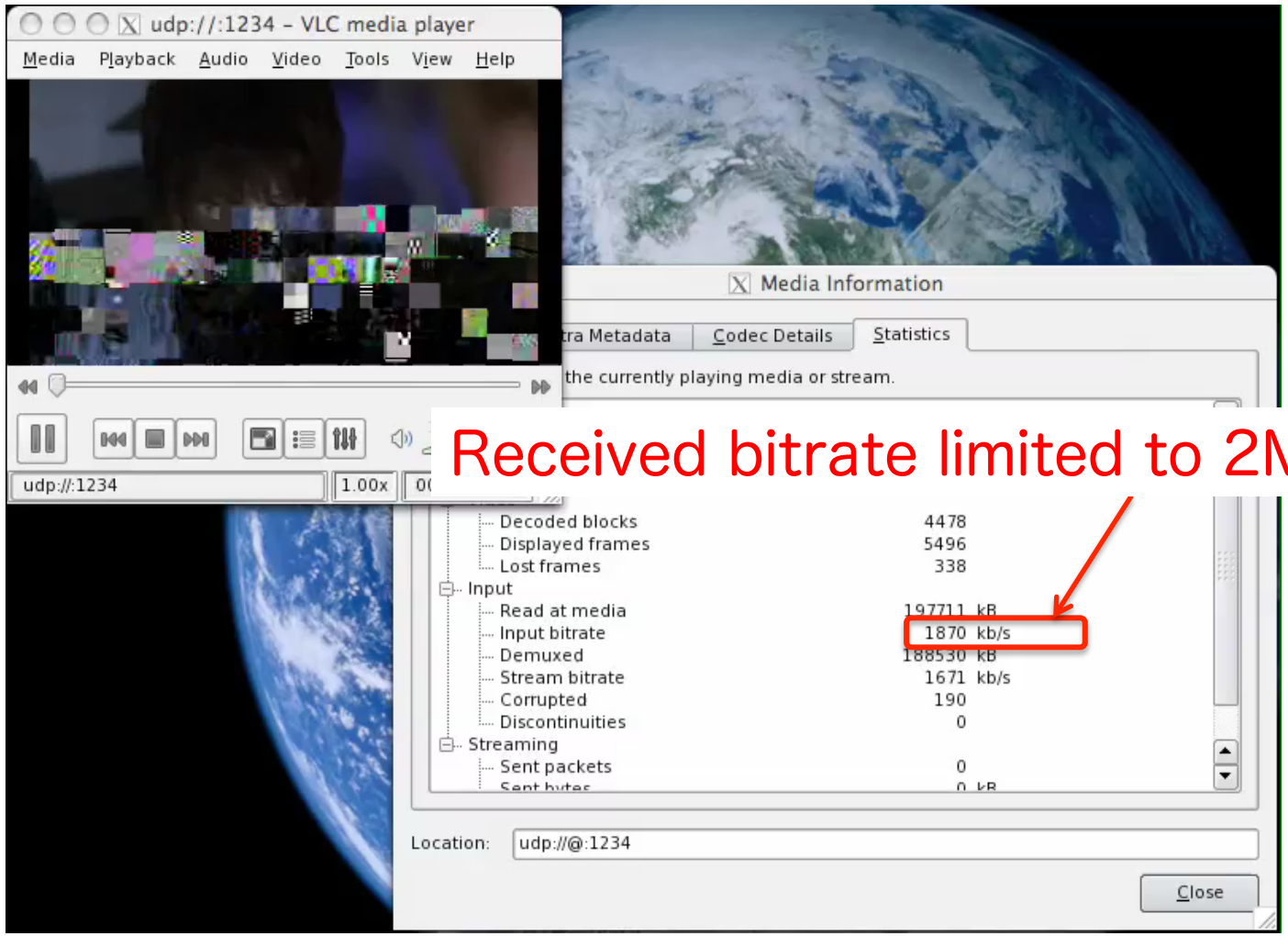
Fine Video



DCN VLAN bandwidth: 2Mbps

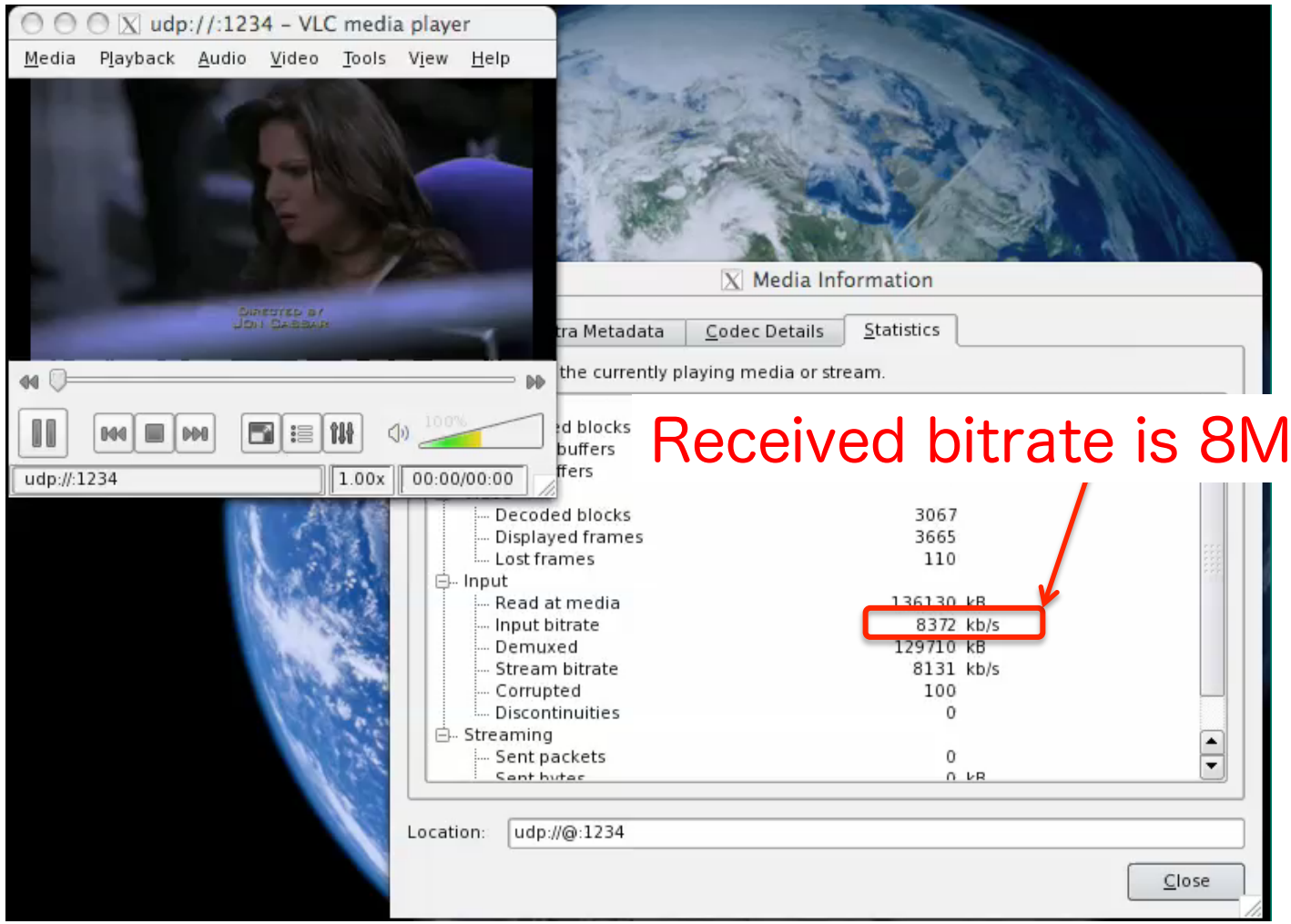
DCN VLAN bandwidth: 10Mbps

2Mbps Not Enough for Replaying Video



DCN VLAN Bandwidth Limit : 2Mbps

10Mbps Achieves Fine Video Playback



Received bitrate is 8Mbps

DCN VLAN Bandwidth Limit: 10Mbps

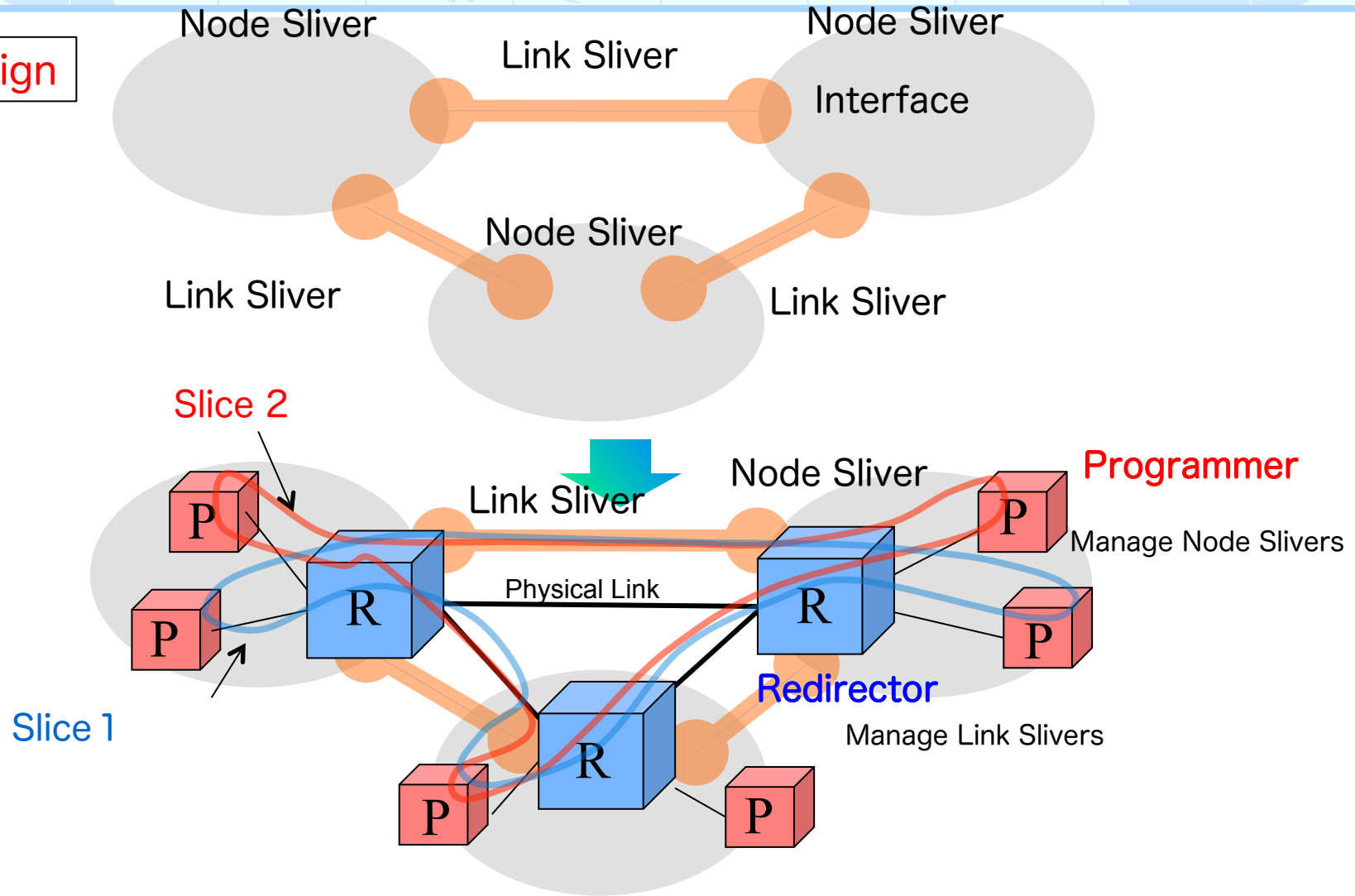
VNode Project

(UTokyo, NICT, NTT, NEC, Hitachi, Fujitsu)

Learned lessons from PlanetLab and CoreLab
Starting over and designing a new “slice” mechanism
Enabling net-virt via H/W based on production routers

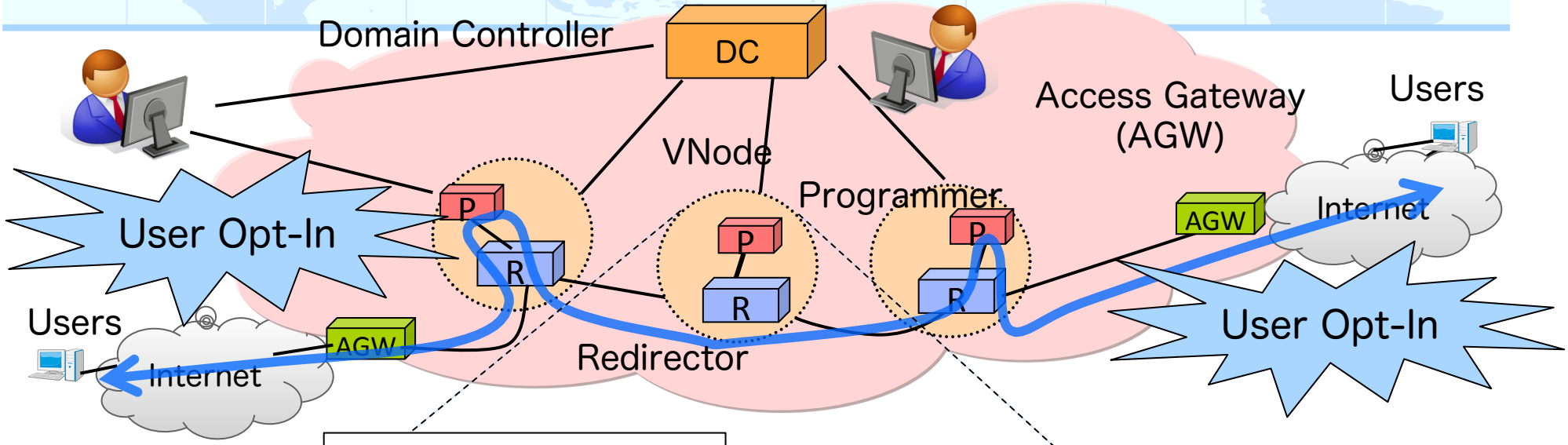
VNode Architecture 1/2

Slice Design

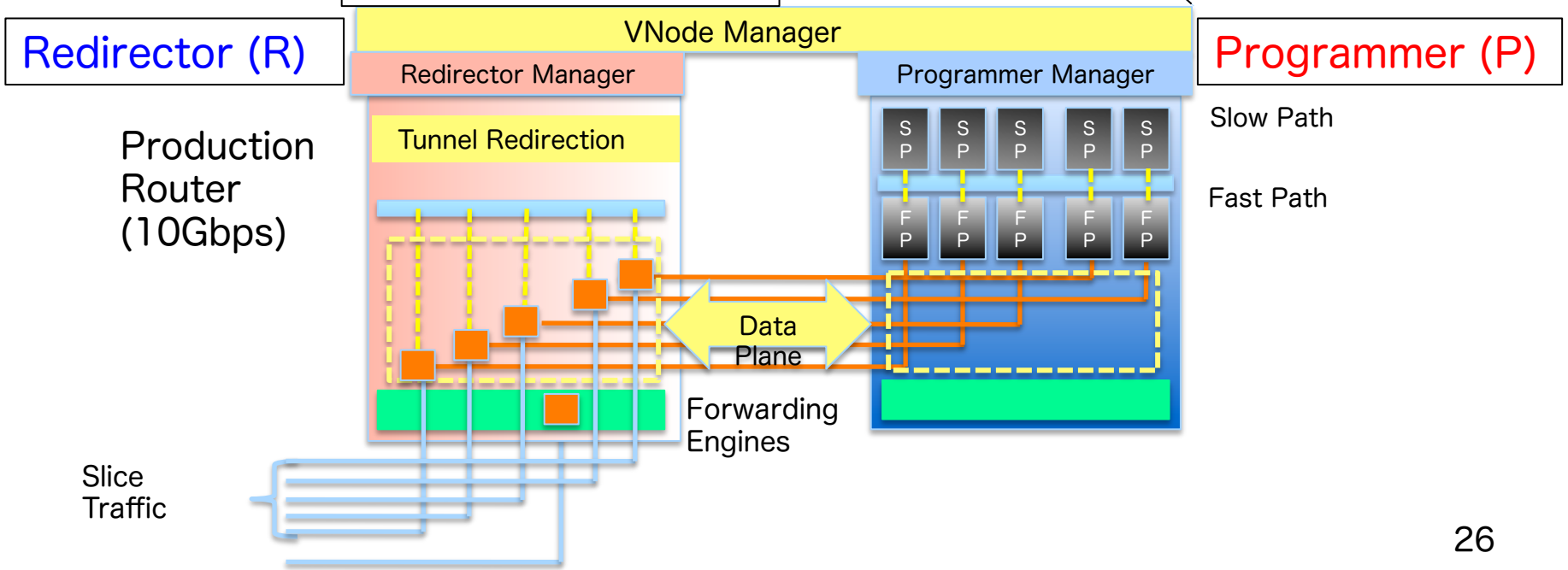


- Separation between Connectivity and Programmability
- Redirector and Programmer may evolve independently

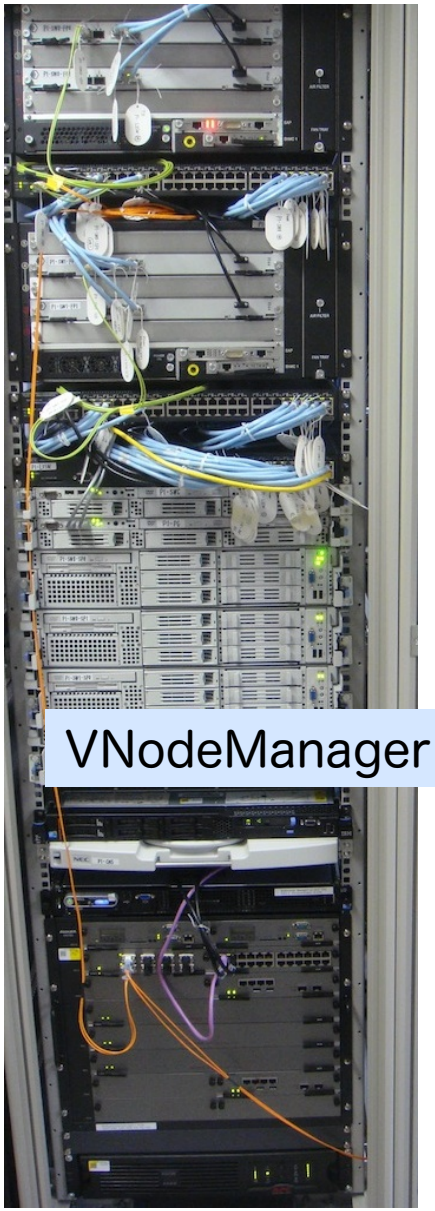
VNode Architecture 2/2



VNode Architecture



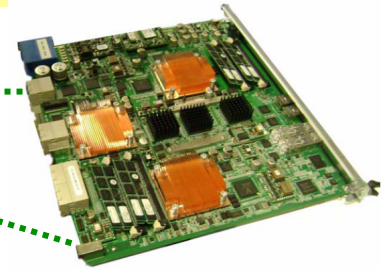
VNode: Nuts and Bolts View



VNodeManager

Programmer Part
(IA Serverx4 +ATCAx2+OpenFlow SWx2)

OpenFlow Switch

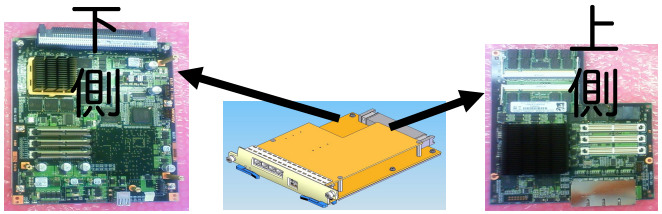


Fast-Path
Network Processor
Card

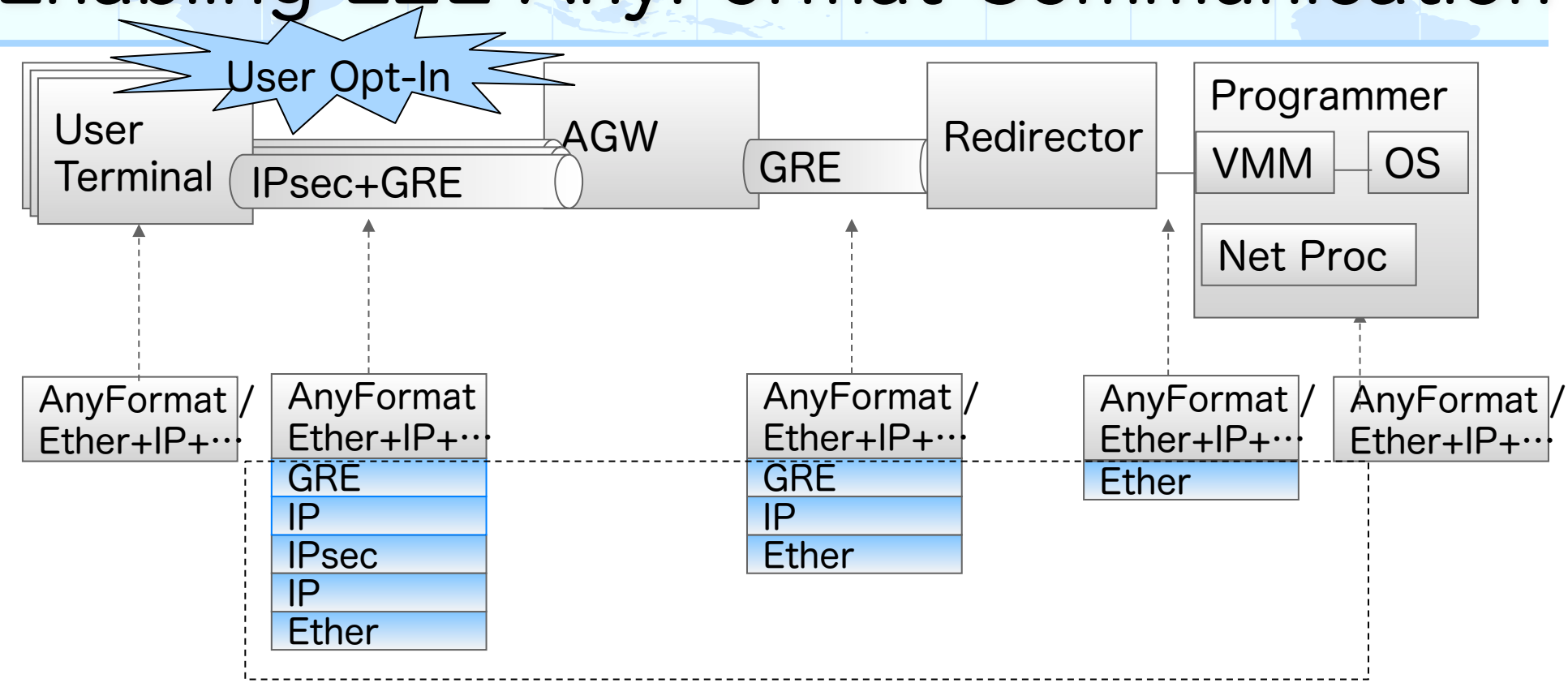
Redirector Part (AX6700+SMCx2)

AX6708S

Service Module Card



Enabling E2E AnyFormat Communication

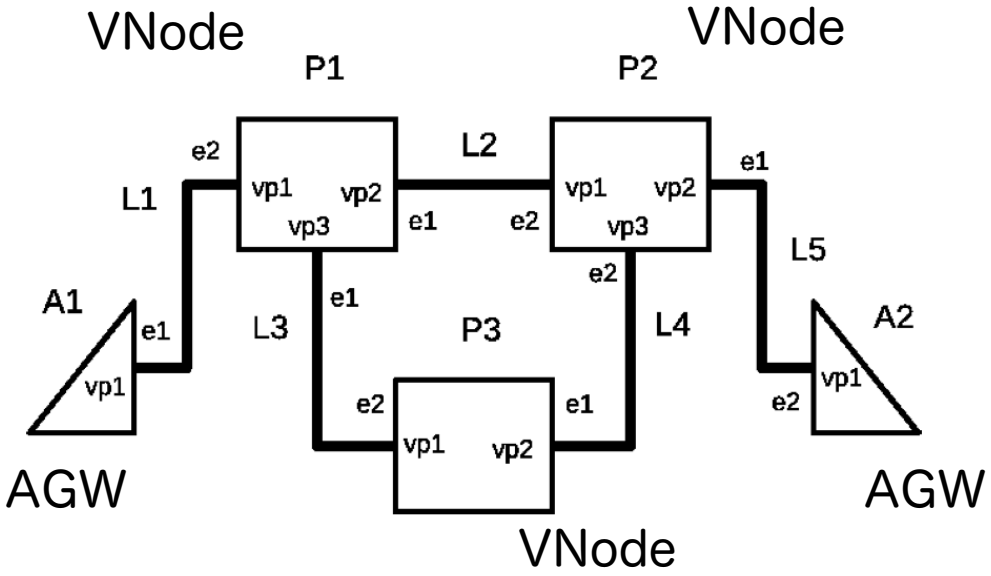


This part could be replaced with any underlay technology by new “Redirectors”

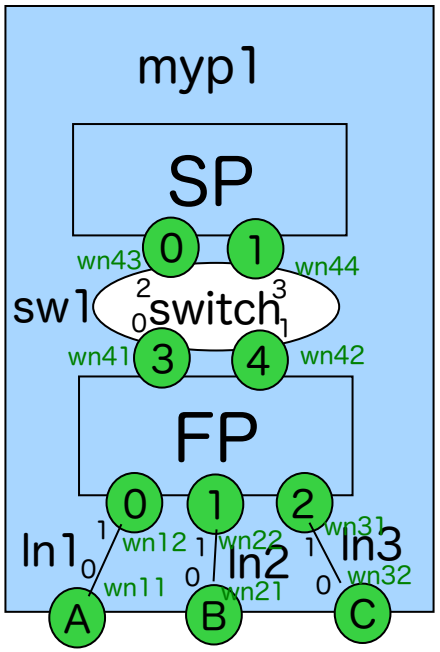
- Enable End-to-End ”AnyFormat” Communication
- Use GRE-tap but may extend this to MPLS, VLAN, and Optical Path

Slice Design for a Virtual Network

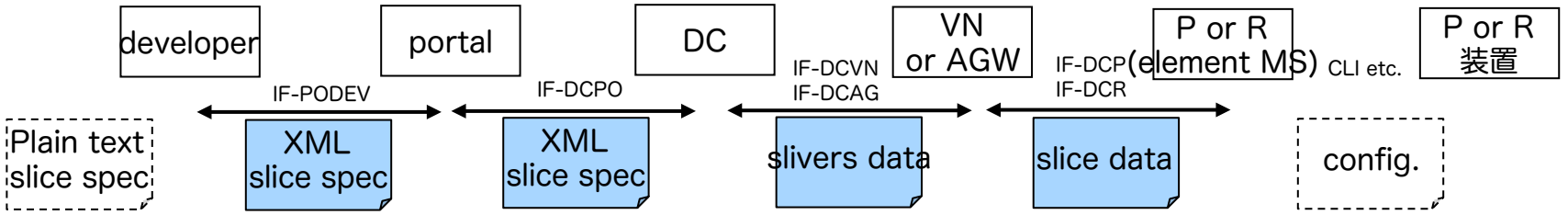
Slice Specification



Node Sliver



Slice Specification Flow



Slice Design XML Format

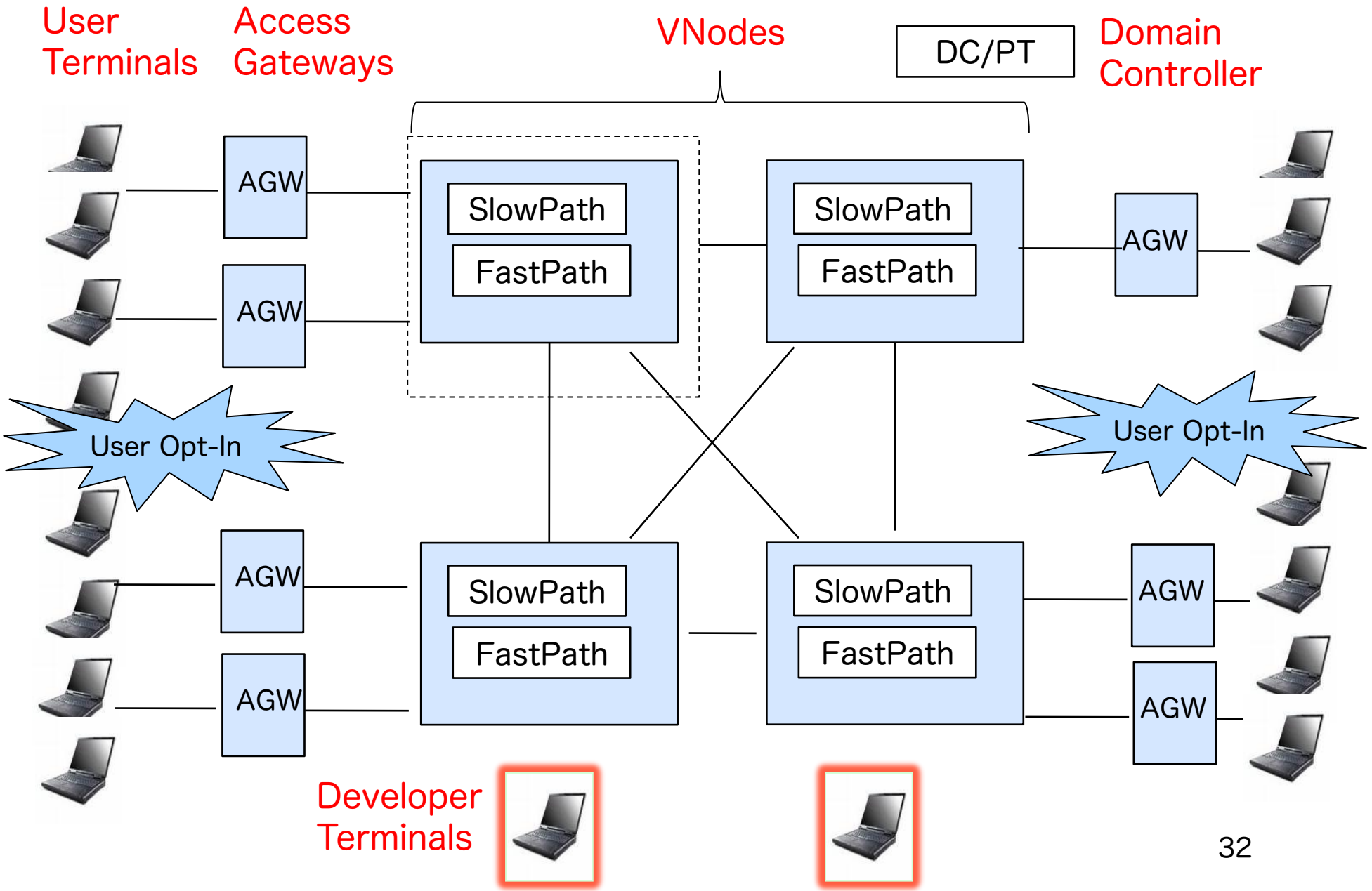
```
<NodeSliver>
 <sliverID>Sample_Node_Sliver</sliverID>
 <sliceID>100</sliceID>
 <virutalPorts>
  <virtualPort><name>A</name></virtualPort>
  <virtualPort><name>B</name></virtualPort>
  <virtualPort><name>C</name></virtualPort>
 </virutalPorts>
 <components>
  <NodeSliverComponent>
   <name>FP</name>
   <type>FastPath_BareMetal</type>
   <subtype>RMI</subtype>
   <interfaces>
    <numOfInterfaces>5</numOfInterfaces>
   </interfaces>
   <resource>
    <cpu>1</cpu>
    <cpumode>dedicated</cpumode>
    <memory unit="MB">256</memory>
   </resource>
   <parameters>
    <bootImage>http://example.com/RMI_binary </bootImage>
    <bootParams/>
   </parameters>
  </NodeSliverComponent>
  <NodeSliverComponent>
   <name>SP</name>
   <type>SlowPath_VM</type>
   <subtype>KVM</subtype>
   <interfaces>
    <numOfInterfaces>2</numOfInterfaces>
   </interfaces>
   <resource>
    <cpu>1</cpu>
    <cpumode>dedicated</cpumode>
    <memory unit="B">2048</memory>
   </resource>
   <parameters>
    <bootImage>http://example.com/KVM_disk.img </bootImage>
    <bootParams/>
   </parameters>
  </NodeSliverComponent>
 </components>
```

```
 <connectionMap>
  <NodeConnection type="P-to-P">
   <source>
    <VirtualInterface>
     <component>_VPORT_</component>
     <interface>A</interface>
    </VirtualInterface>
   </source>
   <destination>
    <VirtualInterface>
     <component>FastPath</component>
     <interface>0</interface>
    </VirtualInterface>
   </destination>
   <bidir>TRUE</bidir>
  </NodeConnection>
  <NodeConnection type="P-to-P">
   <source>
    <VirtualInterface>
     <component>_VPORT_</component>
     <interface>B</interface>
    </VirtualInterface>
   </source>
   <destination>
    <VirtualInterface>
     <component>FastPath</component>
     <interface>1</interface>
    </VirtualInterface>
   </destination>
   <bidir>TRUE</bidir>
  </NodeConnection>
  <NodeConnection type="P-to-P">
   <source>
    <VirtualInterface>
     <component>_VPORT_</component>
     <interface>C</interface>
    </VirtualInterface>
   </source>
   <destination>
    <VirtualInterface>
     <component>FastPath</component>
     <interface>2</interface>
    </VirtualInterface>
   </destination>
   <bidir>TRUE</bidir>
  </NodeConnection>
```

```
  <NodeConnection type="Switch">
   <VirtualInterface>
    <component>FastPath</component>
    <interface>3</interface>
   </VirtualInterface>
   <VirtualInterface>
    <component>FastPath</component>
    <interface>4</interface>
   </VirtualInterface>
   <VirtualInterface>
    <component>SlowPath</component>
    <interface>0</interface>
   </VirtualInterface>
   <VirtualInterface>
    <component>SlowPath</component>
    <interface>1</interface>
   </VirtualInterface>
  </NodeConnection>
 </connectionMap>
 <parameters>
  <authKey>Base64_Encoded_SshKeyFile</authKey>
 </parameters>
</NodeSliver>
```

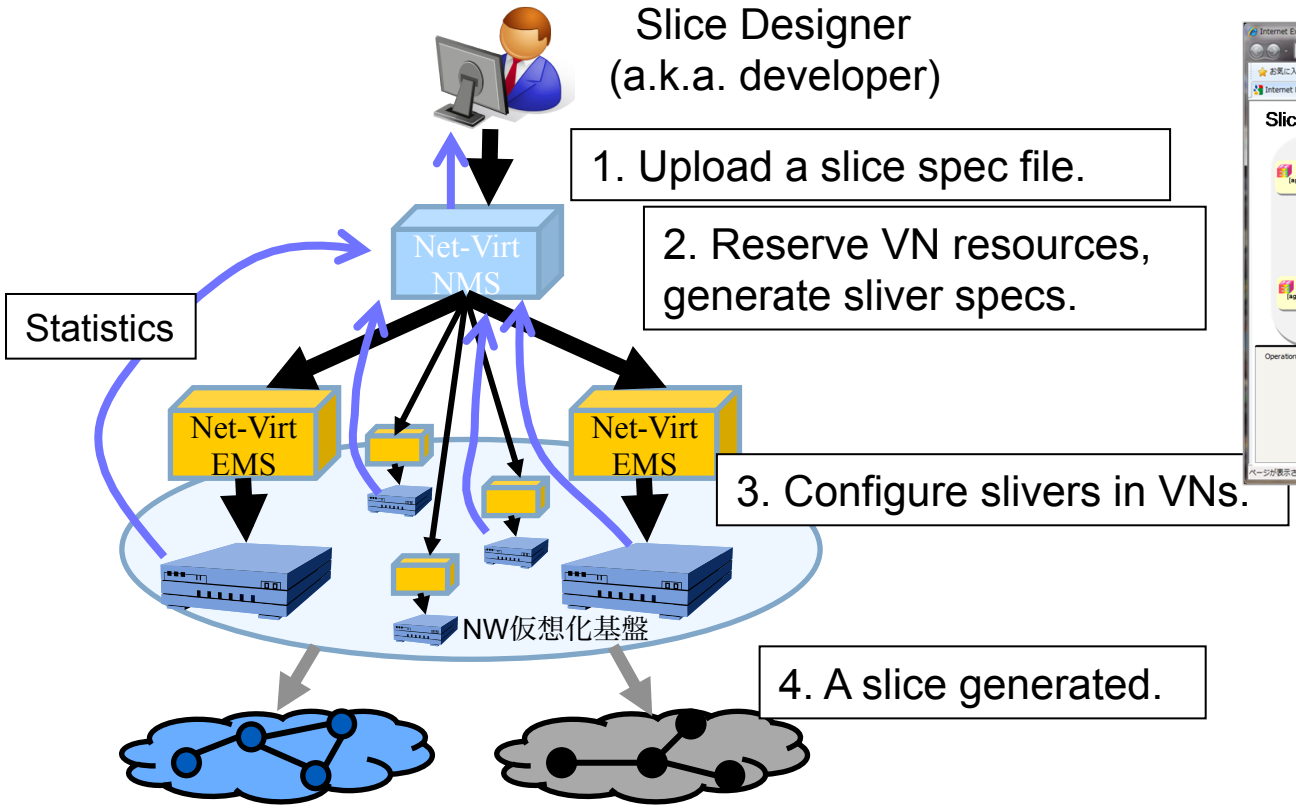
A Prototype System (4 VNodes)

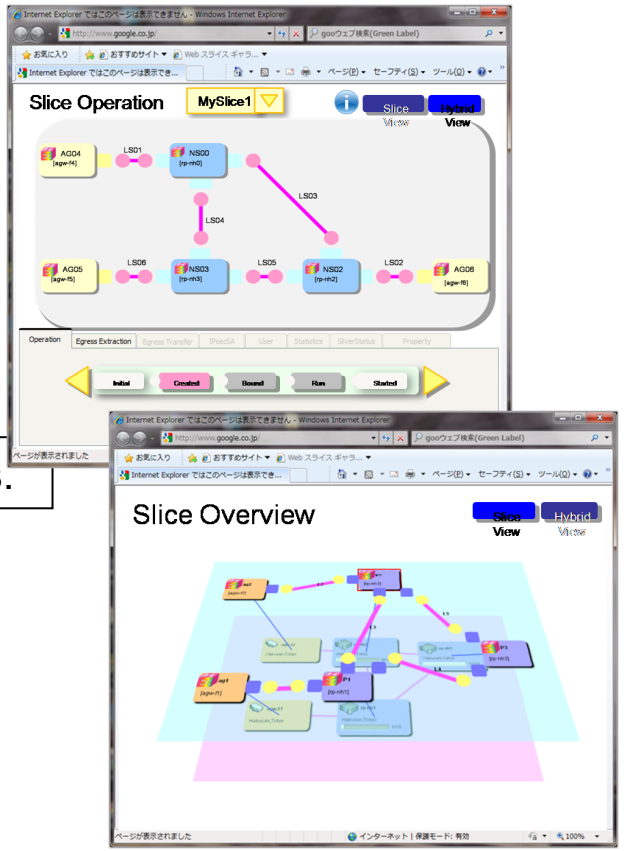
Simplified Picture of Prototype



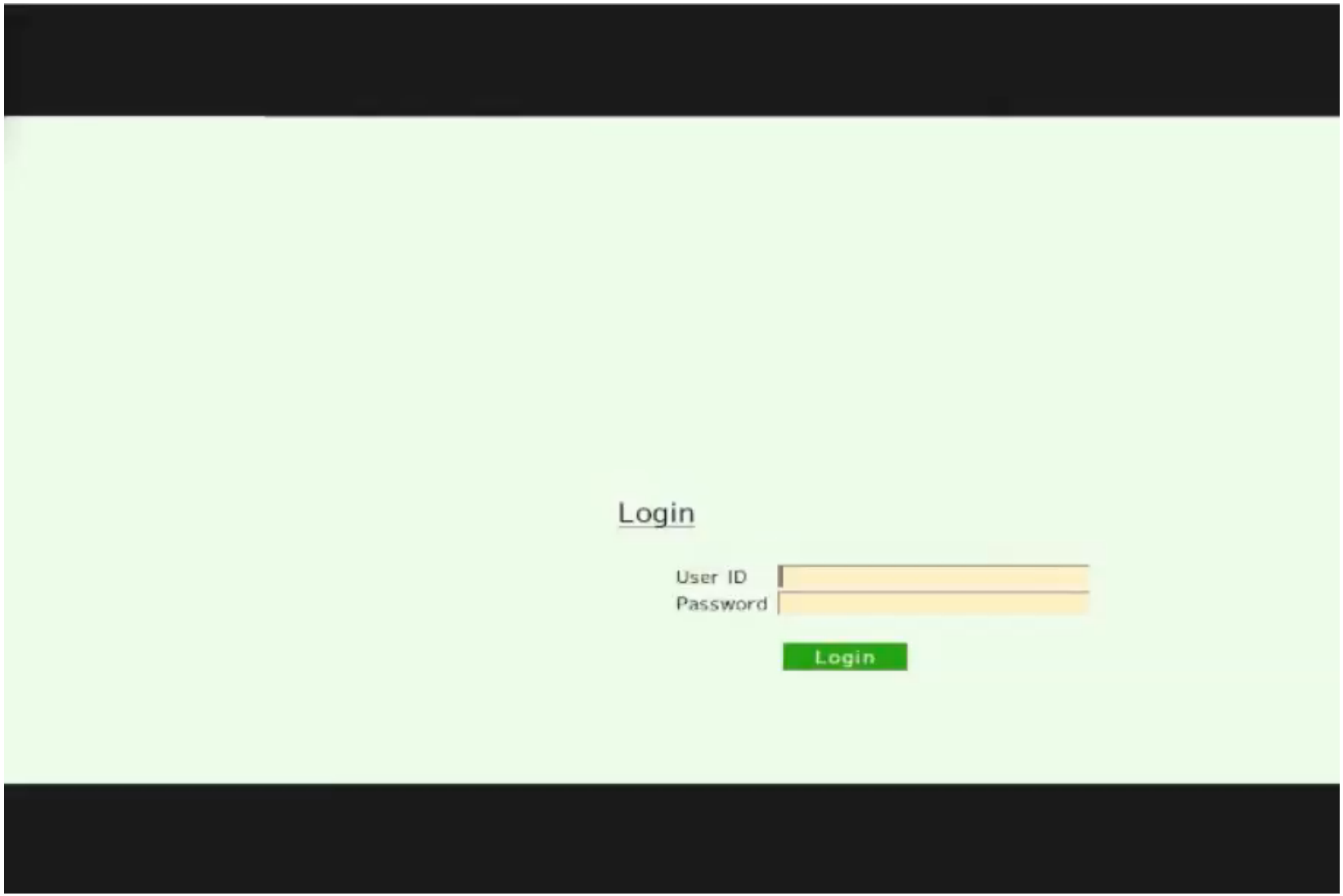
Management/Control for VNode Platform

Get requests from *developers* and send control messages to multiple *VNodes* to create *slices* dynamically





VNode Control Plane Demo



Benefit of VNode Infrastructure

(A) Accommodate Multiple Independent Networks
    => Meta Architecture
(B) Enable In-Network Processing
(C) Create App-Specific Networks
(D) Adapt Computational Resources
(E) Enable Non-IP E2E Communication

6 Slices Running on VNode Prototype

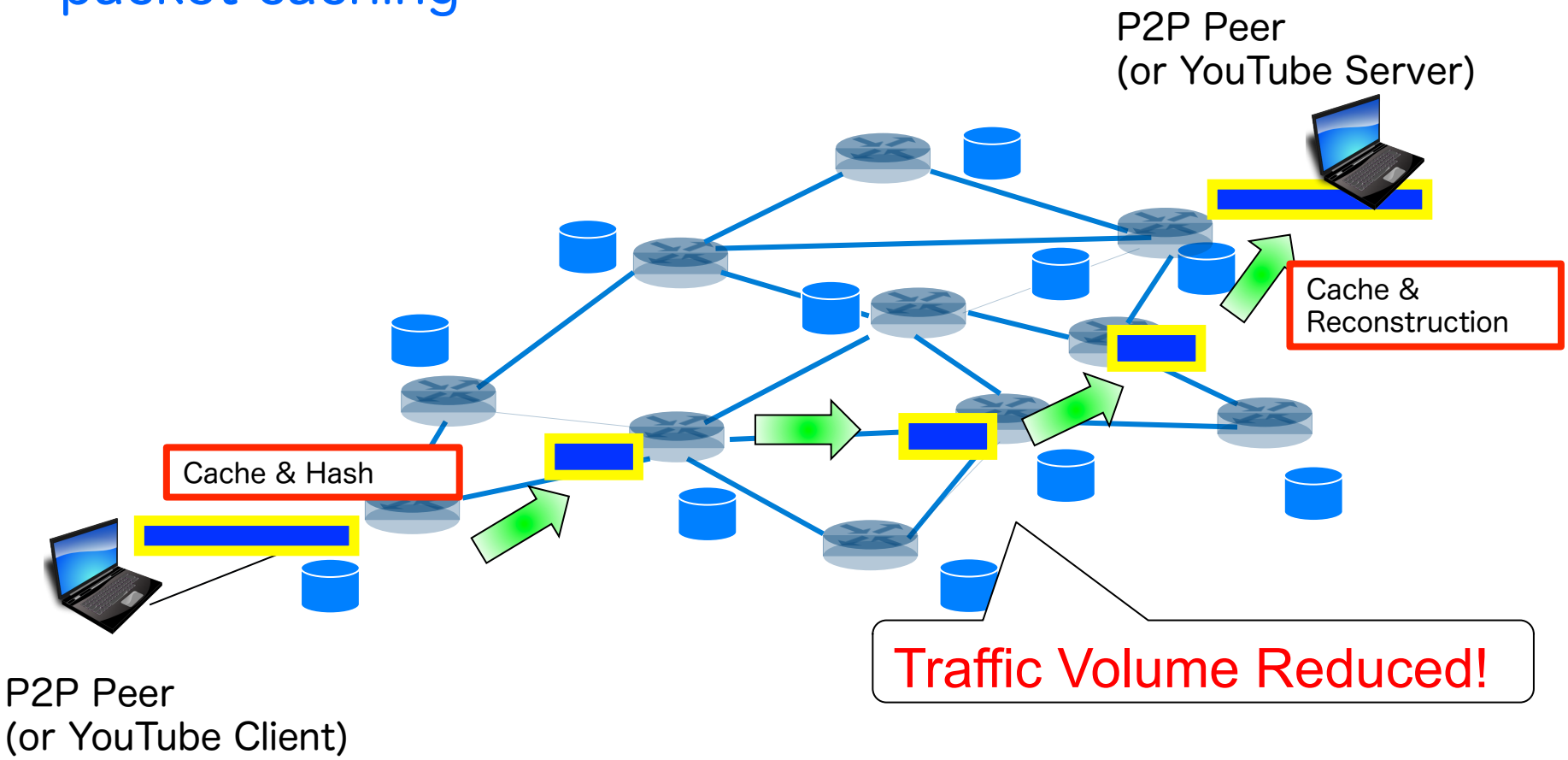
Which features of VNode Infra that each slice demonstrates

Group	NW Archiecture	Meta Architecture	In Network Processing	App Specific Network	Adaptive Computing Resources	Non-IP (E2E) Architecture
UTokyo N I C T	Packet Cache	Slice 1	Packet Cache		SlowPath FastPath ※	
N T T	Flexcast	Slice 2	Flexible Multicast			
Fujitsu	Sensor Specific Networking	Slice 3	Sensor Data Processing	Sensor Specific Network		
N E C	Stream Computing	Slice 4	Stream Computing		SlowPath FastPath	Non-IP Ephemeral Multicast
Hitachi1	IP-Ethernet-Chimera (IPEC)	Slice 5	IPEC (L2/L3)			IPEC (L2/L3)
Hitachi2	Data Sync for Distributed Data Center	Slice 6		Data Sync Network QoS		

Slice 1: Cache Oriented NW Architecture

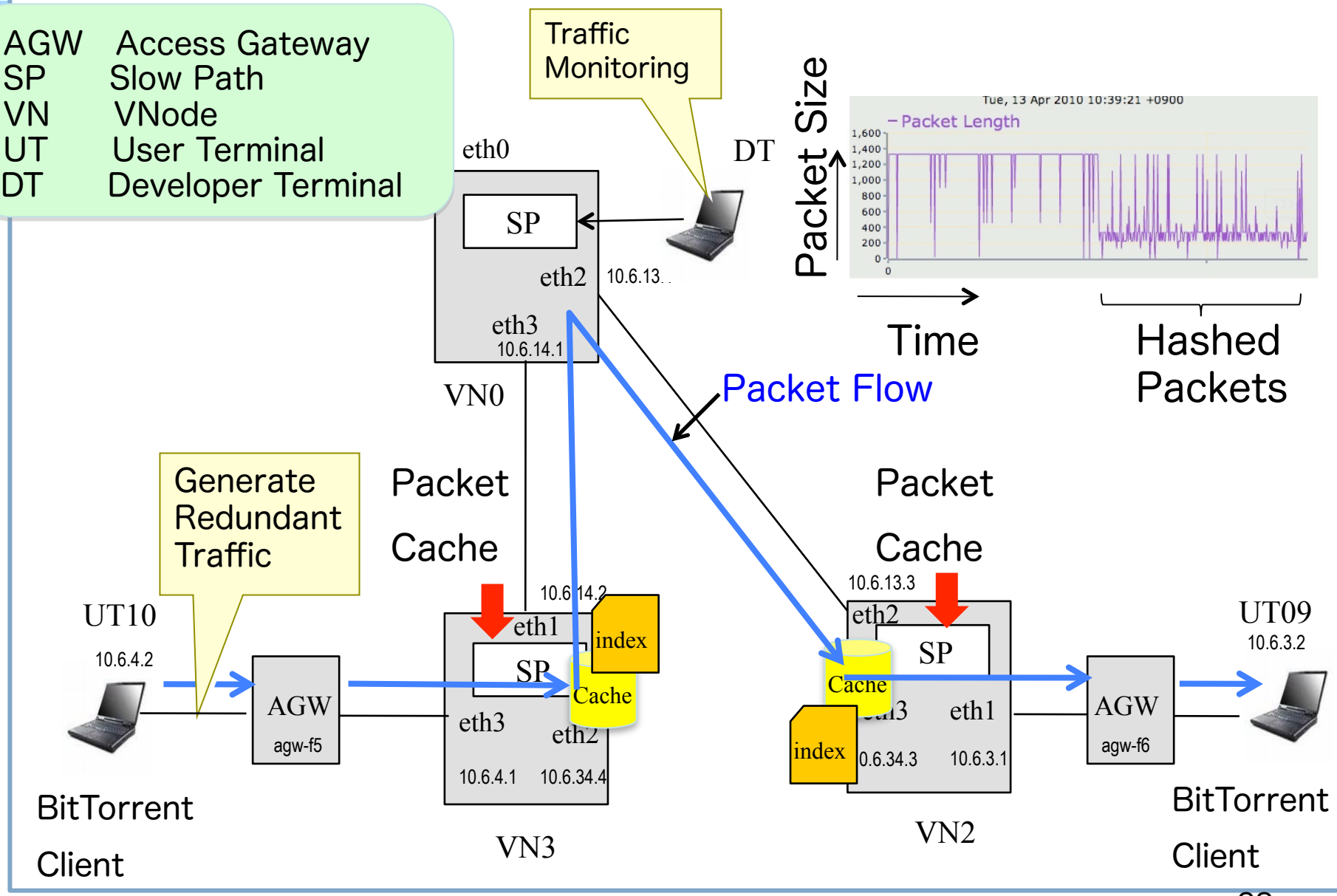
Goal : Reduce redundant traffic in P2P and Cloud Streaming

- Utilize programmability and storage in VNode
- Confine P2P and YouTube Traffic to a slice and perform packet caching

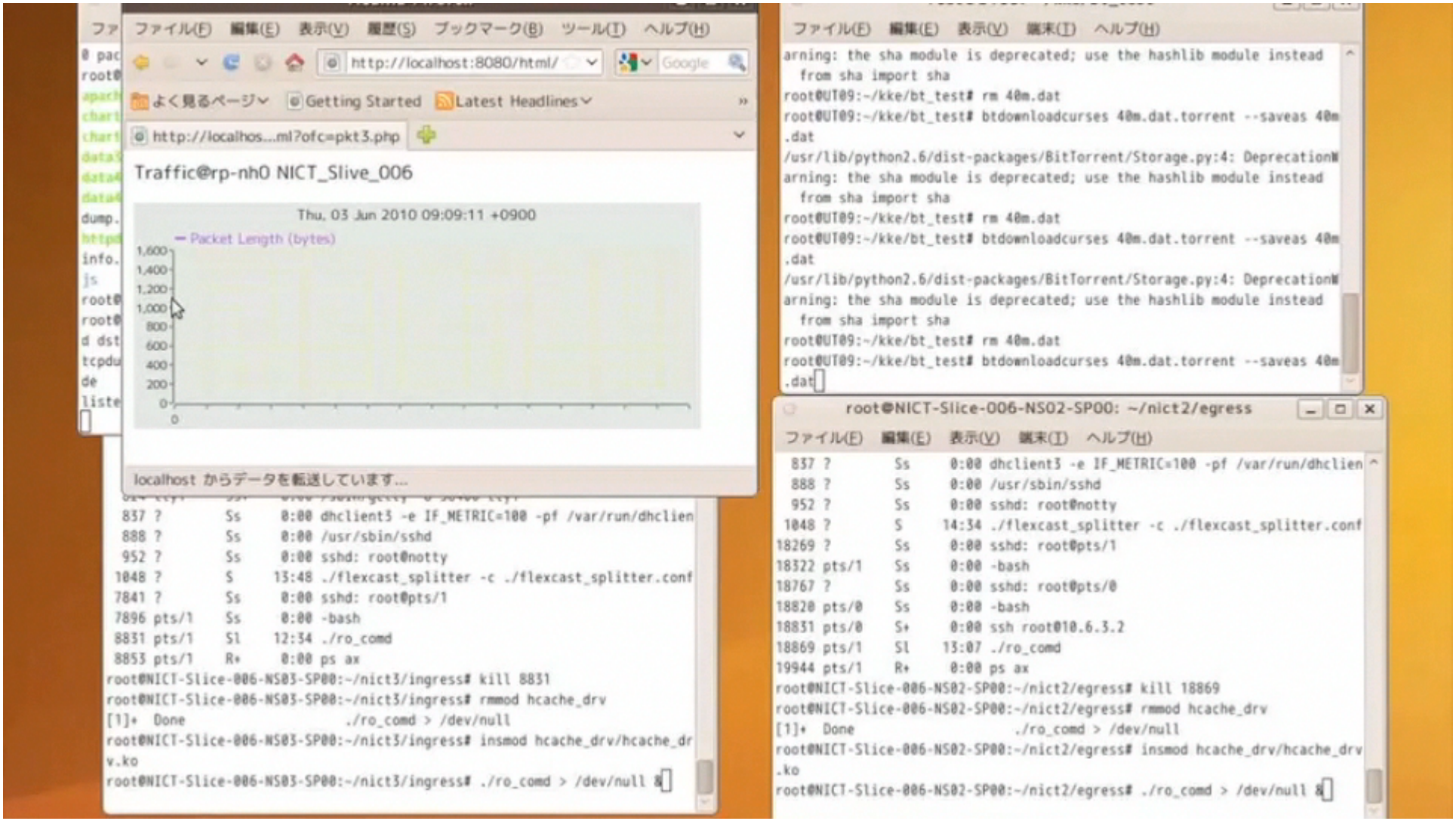


P2P Packet Caching

AGW Access Gateway
SP Slow Path
VN VNode
UT User Terminal
DT Developer Terminal

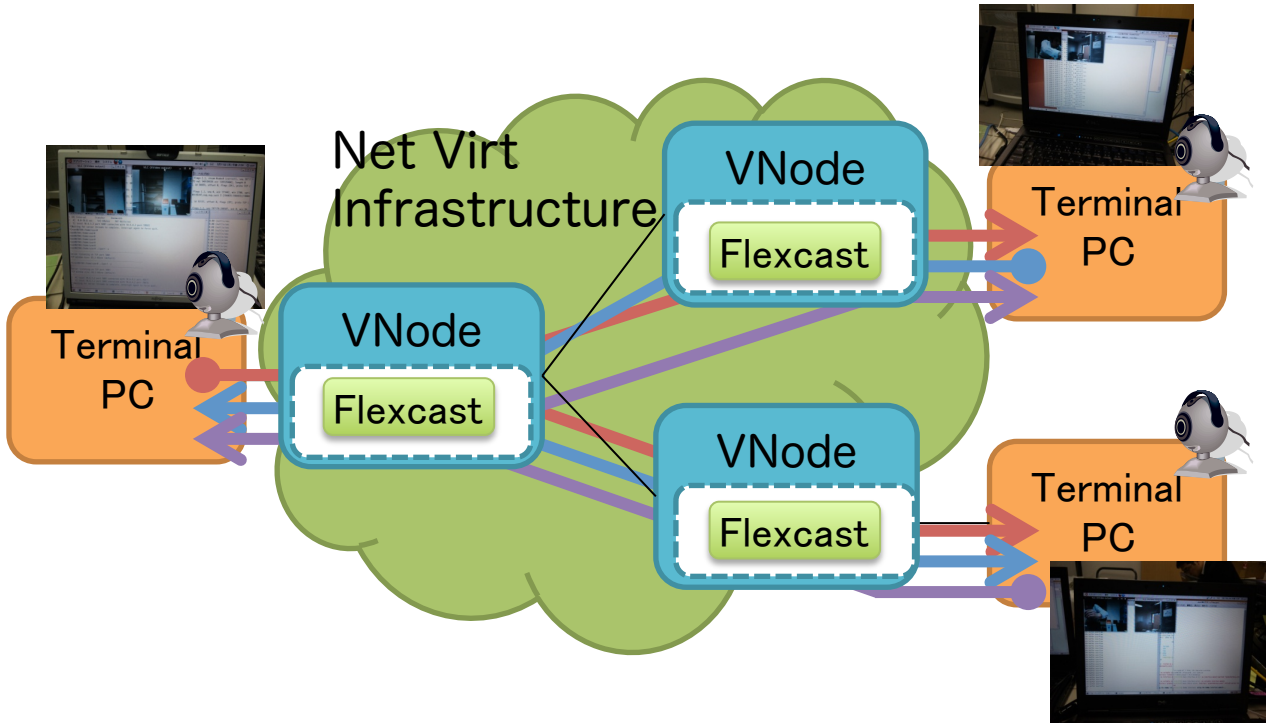


CONA Demo


ファイル(F) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)
http://localhost:8080/html/
よく見るページ
Getting Started
Latest Headlines
http://localhos...ml?ofc=pkt3.php
Traffic@rp-nh0 NICT_Slice_006
Thu, 03 Jun 2010 09:09:11 +0900
Packet Length (bytes)
1,600
1,400
1,200
1,000
800
600
400
200
0
localhost からデータを転送しています...
root@NICT-Slice-006-NS03-SP00:~/nict3/ingress# kill 8831
root@NICT-Slice-006-NS03-SP00:~/nict3/ingress# rmmod hcache_drv
[1]+ Done ./ro_comd > /dev/null
root@NICT-Slice-006-NS03-SP00:~/nict3/ingress# insmod hcache_drv/hcache_drv.ko
root@NICT-Slice-006-NS03-SP00:~/nict3/ingress# ./ro_comd > /dev/null &
root@UT09:~/kke/bt_test# rm 40m.dat
root@UT09:~/kke/bt_test# btdownloadcurses 40m.dat.torrent --saveas 40m.dat
root@NICT-Slice-006-NS02-SP00: ~/nict2/egress
root@NICT-Slice-006-NS02-SP00:~/nict2/egress# kill 18869
root@NICT-Slice-006-NS02-SP00:~/nict2/egress# rmmod hcache_drv
[1]+ Done ./ro_comd > /dev/null
root@NICT-Slice-006-NS02-SP00:~/nict2/egress# insmod hcache_drv/hcache_drv.ko
root@NICT-Slice-006-NS02-SP00:~/nict2/egress# ./ro_comd > /dev/null &

Slice 2: Ephemeral Streaming Protocol

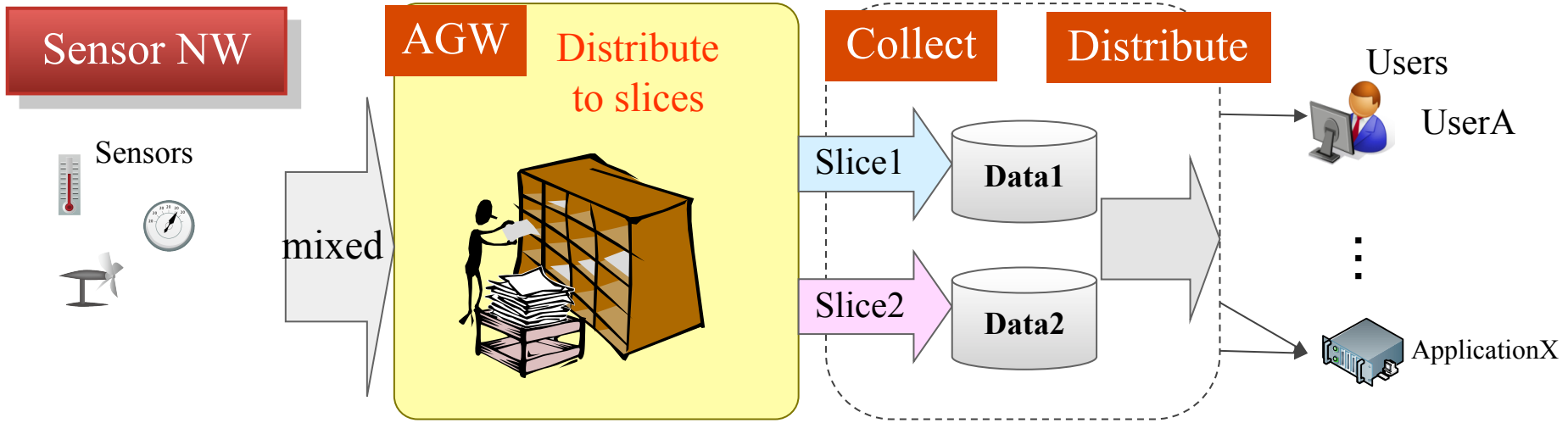
- Enabling *ephemeral multicast streaming* of videos
- Instantly deploy and shutdown *traffic splitters in NW*
- Flexcast: Non-IP protocol (planned)



Slice 3: Sensor Data In-Network Processing

Sensor Specific Network In A Slice

- Implemented NW service demo system to gather various sensor data, and distribute it after processing



- Issues
  - Massive sensor data --> Scalability
  - Add new type of sensors --> Increase development cost

- Effect of network virtualization
  - Offload application processing into network
  - Hide sensor specific network protocols

- Users / Applications can concentrate on service processing.

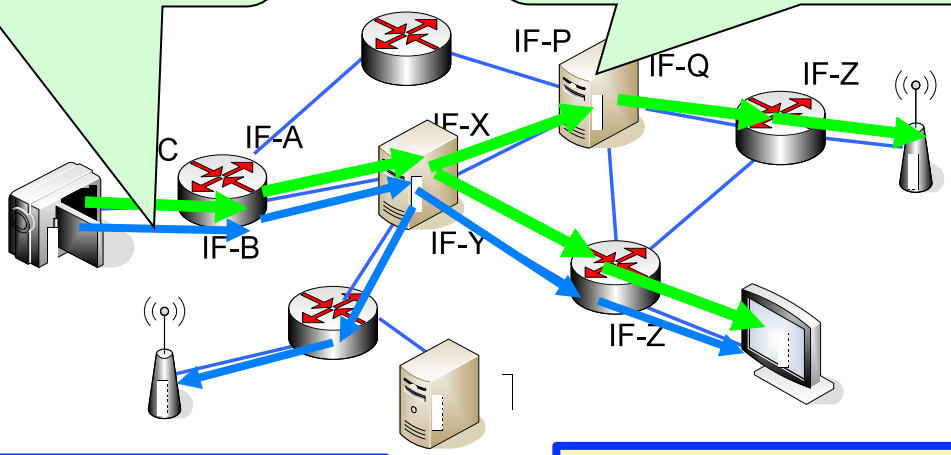
Slice 4: Stream Computing Platform

Chaining Computational Resources Along E2E Data Paths

Key technologies for Stream Computing Platform

Ephemeral Multicast
Efficient method for short time and frequent Multicast

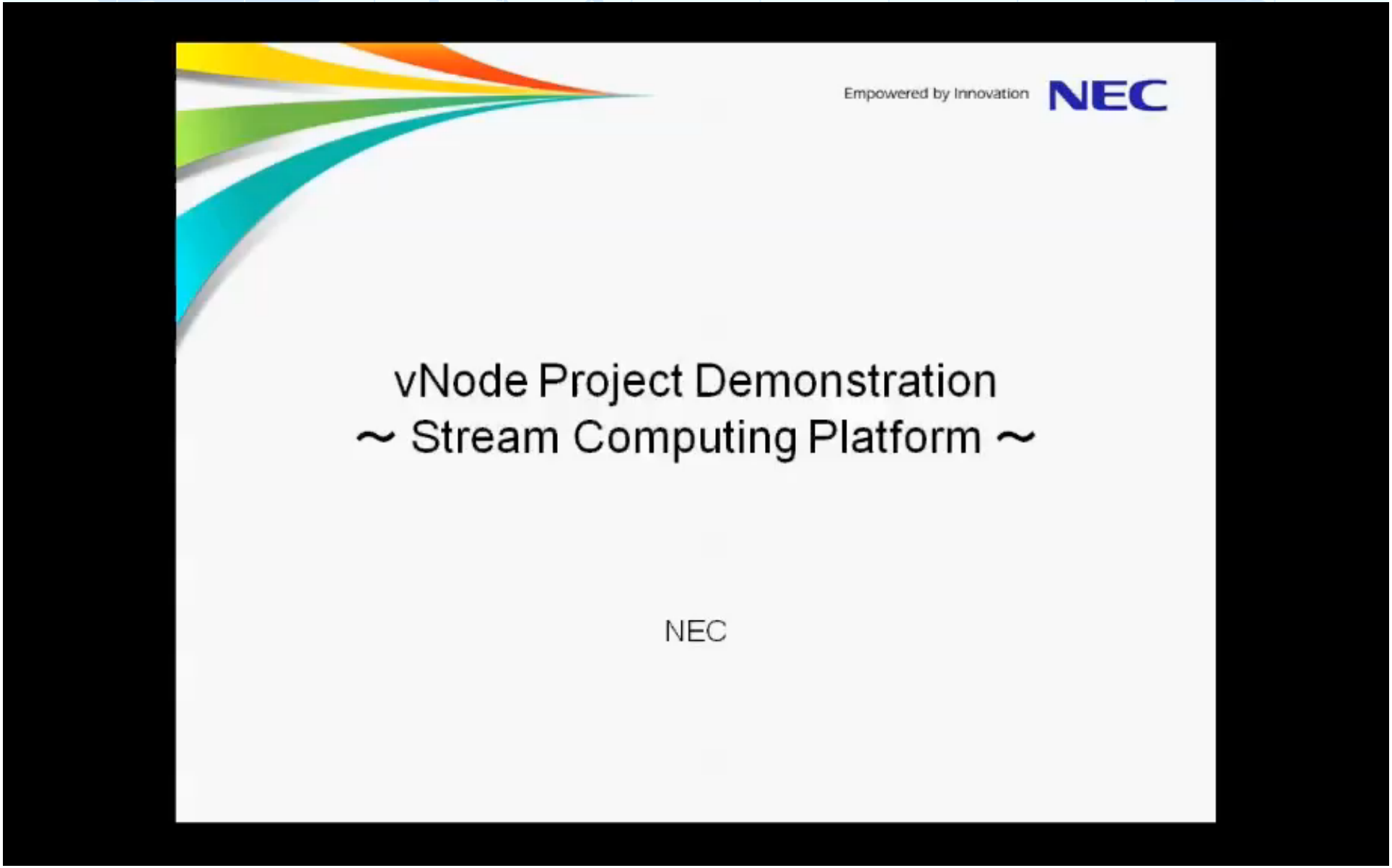
In-Network Processing
Support realtime on-the-fly processing for various data.



On VNode, we can
- Implement a original routing protocol, (Source Routing based on output INF)
- Use our own packet format (Not Ethernet or IP packet format)

On VNode, we can
- Use computing resource in network,
- Select appropriate computing resource according to the processing requirements
  - ✓ Slow-Path (VM)
  - ✓ Fast-Path (Communication Processor)

Stream Computing Platform (Demo)



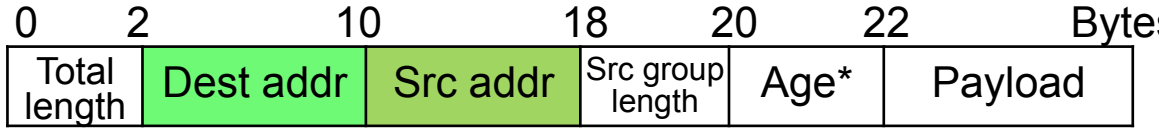
Slice 5: L2/L3 Consolidation (IPEC)

IPEC * — Non-IP Protocol that Operates on the Virtualization Nodes

* IP-Ether-Chimera

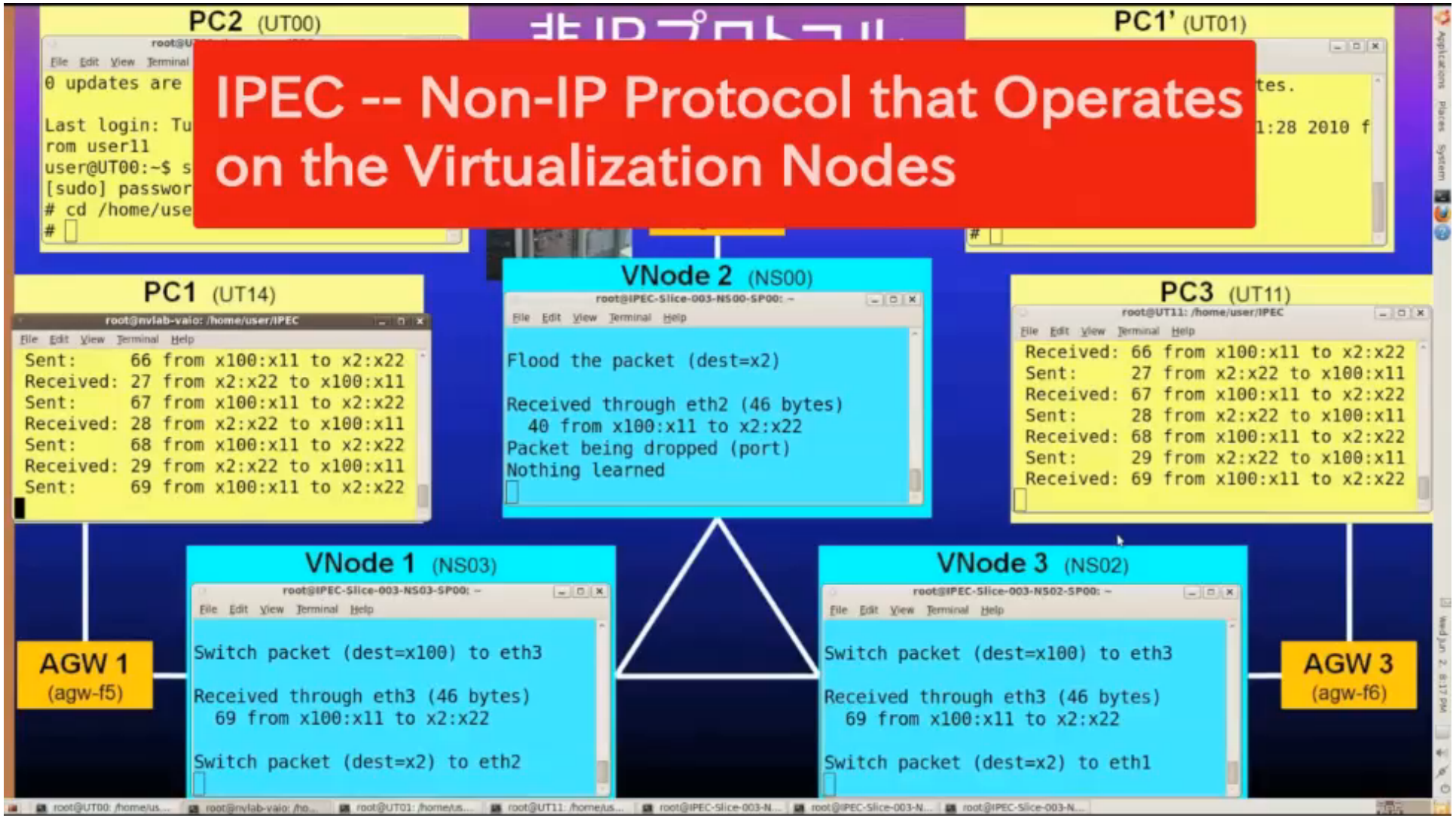
- **Features of IPEC**
  - IPEC has a *group-learning function* that is more scalable than Ethernet learning.
    - When a VNode learns a terminal (PC) in group G, it can switch packets to all the terminals in G.
    - IPEC networks are scalable because of the group-learning function — good for wide-area network.
  - Group motion is efficient in IPEC.
    - If terminals of a group moves to the same place, each VNode learns only once.
  - IPEC works well in networks with loops.
    - In contrast to Ethernet, the learning and forwarding work well even when the data path has loops.
- **The address format of IPEC**
  - An address consists of an ID and a Group.
    - **ID: the identifier of each terminal** — similar to MAC addresses in Ethernet networks.
    - **Group: the identifier of each user group** — used for structured address or location indicator

- **Packet format of IPEC (any format)**



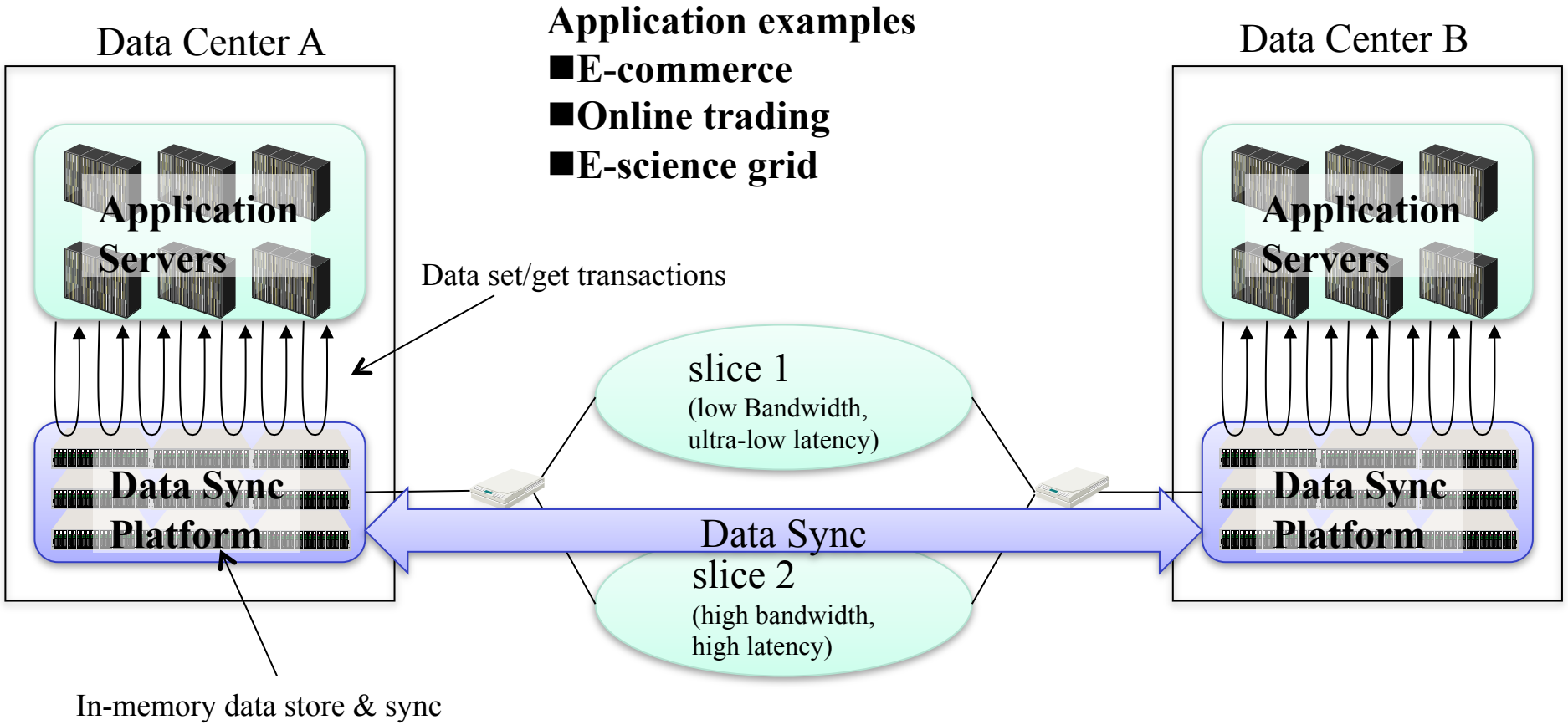
* Age is used to avoid looping.

L2/L3 Consolidation (IPEC) Demo



Slice 6: Data Sync for Distributed Data Centers

- Enables data sync between distributed data centers.
- Performs efficient data sync using various types of slices.
- Adaptive traffic control for dynamically changing sync demands. (demo)

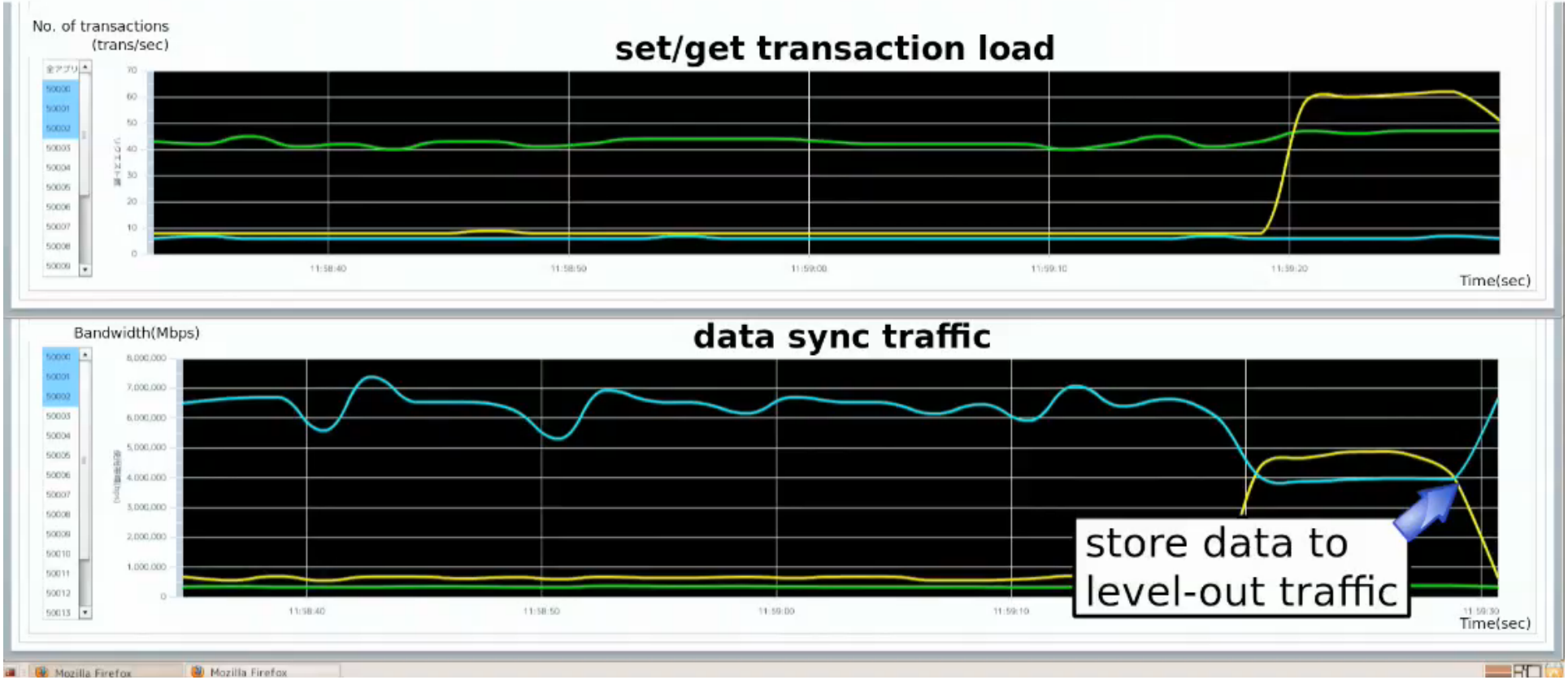


Part of this project is funded by National Institute of Information and Communications Technology (NICT).

Data Sync for Distributed Data Centers (Demo)

Priority:
App A > App B > App C

: App A (Slice 1)
: App B (Slice 2)
: App C (Slice 2)



VNode Deployment on JGN2Plus/JGN-X

Scheduled in the end of August 2010 (4 to 6 VNodes)

Future Directions

- Optical Path Integration
- Cloud Computing & Networking
- More Applications
- CoreLab Deployment and Integration
  - NetFPGA (Switch Blade) Integration
  - Multicore Network Processor Integration
- CoreLab & VNode Federation
- Federation (SFA)
  - CoreLab and PlanetLab
  - VNode and GENI (Cluster X)
- Network Operation and Business Model

Credits

- NICT
  - Ping Du, Maoke Chen, Yuji Nishida, Haruki Denpo, Ryota Ozaki, Kiyohide Nakauchi, et. al. (Hakusan)
  - Jin Tanaka, Hideki Otsuki (JGN2Plus)
- NTT
  - Atsushi Takahara, Noriyuki Takahashi, Yukio Tsukishima, Takeshi Yamamoto et. al.
- NEC
  - Motoo Nishihara, Takeo Hayashi, Akihiro Motoki, Junichi Higuchi, et. al.
- Hitachi
  - Shinji Nishimura, Daisuke Matsubara, Yasushi Kanada, Toshiaki Tarui, et. al.
- Alaxala
  - Yoshiyuki, Kurosaki, Makoto Kitani, et. al.
- Fujitsu
  - Tomohiro Ishihara, Kenichi Abiru, Kazumine Matoba, et. al.

People marked in red are around this room…

Conclusion

- Network Virtualization is one of the key technologies for proceeding further in defining NwGN architecture(s)
- We are turning this vision into reality..

Net-Virtualization Research Lab Contact:
nakao@iii.u-tokyo.ac.jp
info@nvlab.org
http://www.nvlab.org

THE UNIVERSITY OF TOKYO

NiCT National Institute of Information and Communications Technology

JGN2 plus

PLANETLAB Japan
An open platform for developing, deploying, and accessing planetary-scale services