ITU Workshop on Software Defined Networking (SDN) Standardization Landscape

(Geneva, Switzerland, 4 June 2013)

"Deeply Programmable Network" Emerging Technologies for Network Virtualization and Software Defined Network (SDN)

Akihiro Nakao

Associate Professor

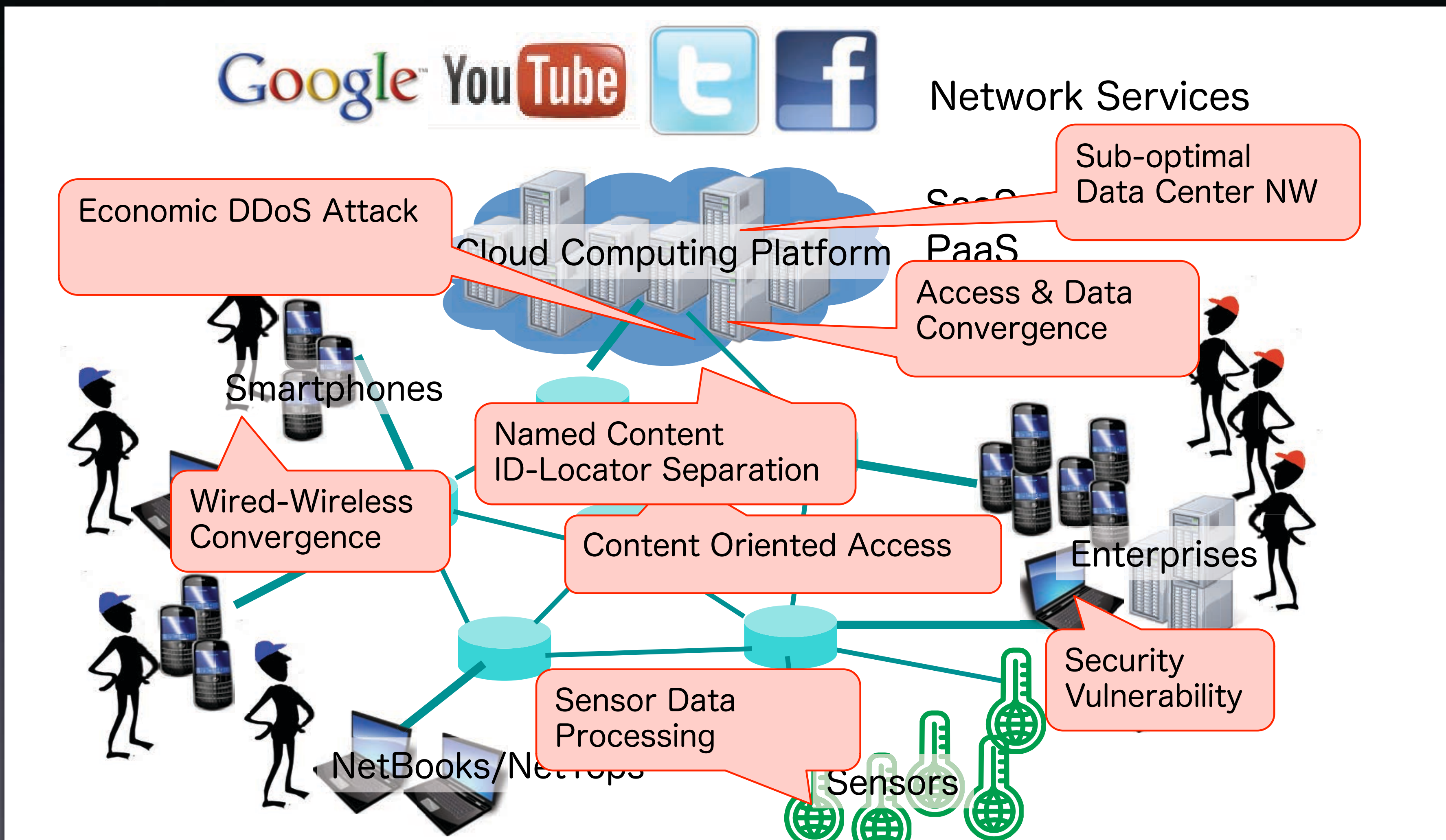
The University of Tokyo

nakao@nakao-lab.org

Future Network Research Proliferating...

- Future Internet Architecture (FIA) in U.S.
- Global Initiative in Network Innovations (GENI) in U.S.
- Framework Programme 7 (FP7) in EU
- Horizon 2020 (2014-) in EU
- New Generation Network (NwGN) in Japan

How can we resolve newly observed, constantly arising problems in the current Internet?



Emerging Areas of Study in Future Network Research

Introducing "programmability" into networking to flexibly and dynamically resolve constantly arising contemporary issues.

- Network Virtualization (NV)
- Software Defined Network (SDN)
- Network Functions Virtualization (NFV)
- And more?

Network Virtualization

In computing, network virtualization is the process of combining hardware and software network resources and network functionality into a single, software-based administrative entity, a virtual network. Network virtualization involves platform virtualization, often combined with resource virtualization.

http://en.wikipedia.org/wiki/Network_virtualization

.... the "advanced network virtualization", govern a collection of the resources ranging from links, networks, and node-software as a slice and create a virtualized network over the slice with dynamically controllable and programmable links and nodes.

http://nvlab.nakao-lab.org/nv-study-group-white-paper.v1.0.pdf

SDN

- Software Driven Network (IETF BoF)
- Software Defined Network
- Some Definition Needed :-)

Software-defined networking (SDN) is an approach to building computer networks that separates and abstracts elements of these systems

SDN decouples the system that makes decisions about where traffic is sent (the control plane) from the underlying system that forwards traffic to the selected destination (the data plane)...

http://en.wikipedia.org/wiki/Software-defined_networking

SDN enables programmability for control-plane so that OPEX in network operation and management can be reduced through automation...

Aki Nakao 2013

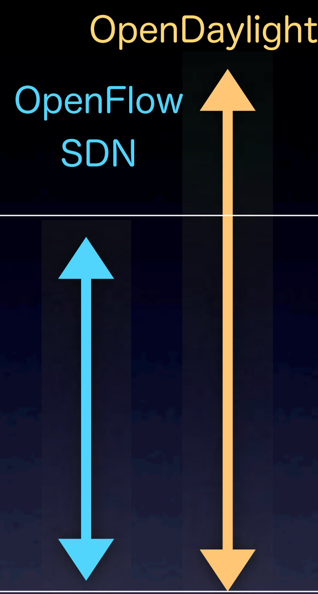
Standardization Activities

- ITU-T on Future Networks / Network Virtualization
- ETSI on Network Functions Virtualization (NFV)

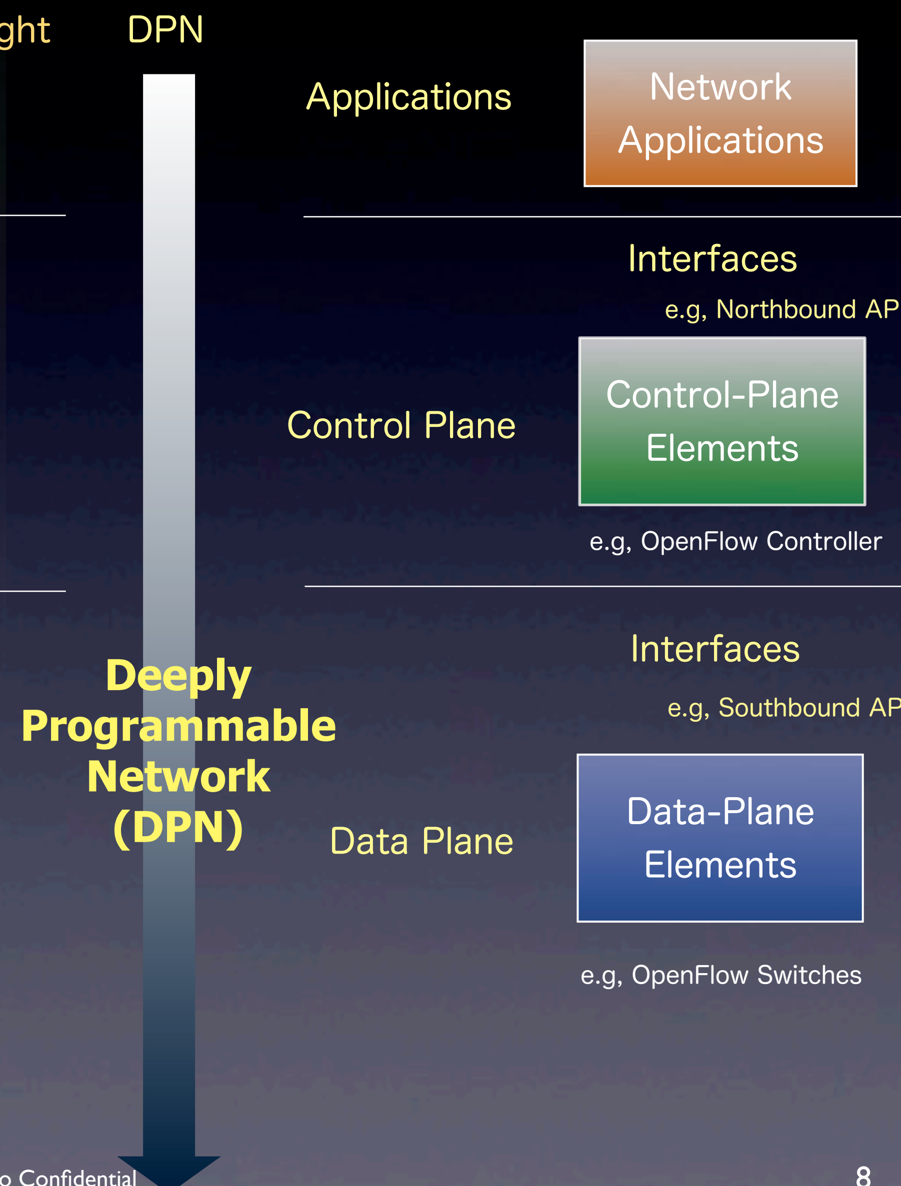
- IETF BoF on Software Driven Networks (SDN)
- IRTF BoF on Network Virtualization
- IETF WG on Interface to Routing System (I2RS) (conceptually similar to SDN)
- ONF (Open Networking Foundation) on
  Software Defined Networks (SDN) /OpenFlow
- OpenDaylight on Software Defined Networks (SDN)

Deep Programmability within Network

- Application Programmability
- Control-Plane Programmability
  - Interfaces
  - Functions
    - Route Control
    - Access Control
    - Network Management



- Data-Plane Programmability
  - Interfaces
  - Functions
    - Packet Data Processing
      - Network Appliances (DPI, BRAS, EPC)
      - In-Network processing (Cache, Transcode)
      - Wide-Area generic processing
    - Handling New Protocols
      - IPvN (N>6), New Layer2, CCN



Discussions necessary at ITU-T

Existing standardization activities

- Network Virtualization (NV)
- Software Defined Network (SDN)
- Network Functions Virtualization (NFV)

Further study and standardization needed at ITU-T (SG13, SG11 or FG-DPN)

- Systematic view and organization of related technologies (NV, SDN, NFV and DPN)
- Deeply Programmable Network (DPN) Technologies
  - Programmability for in-network processing
  - Programmability for new (non-IP) protocols
  - Data-Plane Programmability Interfaces
  - Accommodation of multiple isolated programmable environments

These missing pieces should be studied among industries and academia (GENI, FIA, FP7 NwGN related academia)

DPN Research: FLARE

FLARE

SDN
Control Plane Programmability

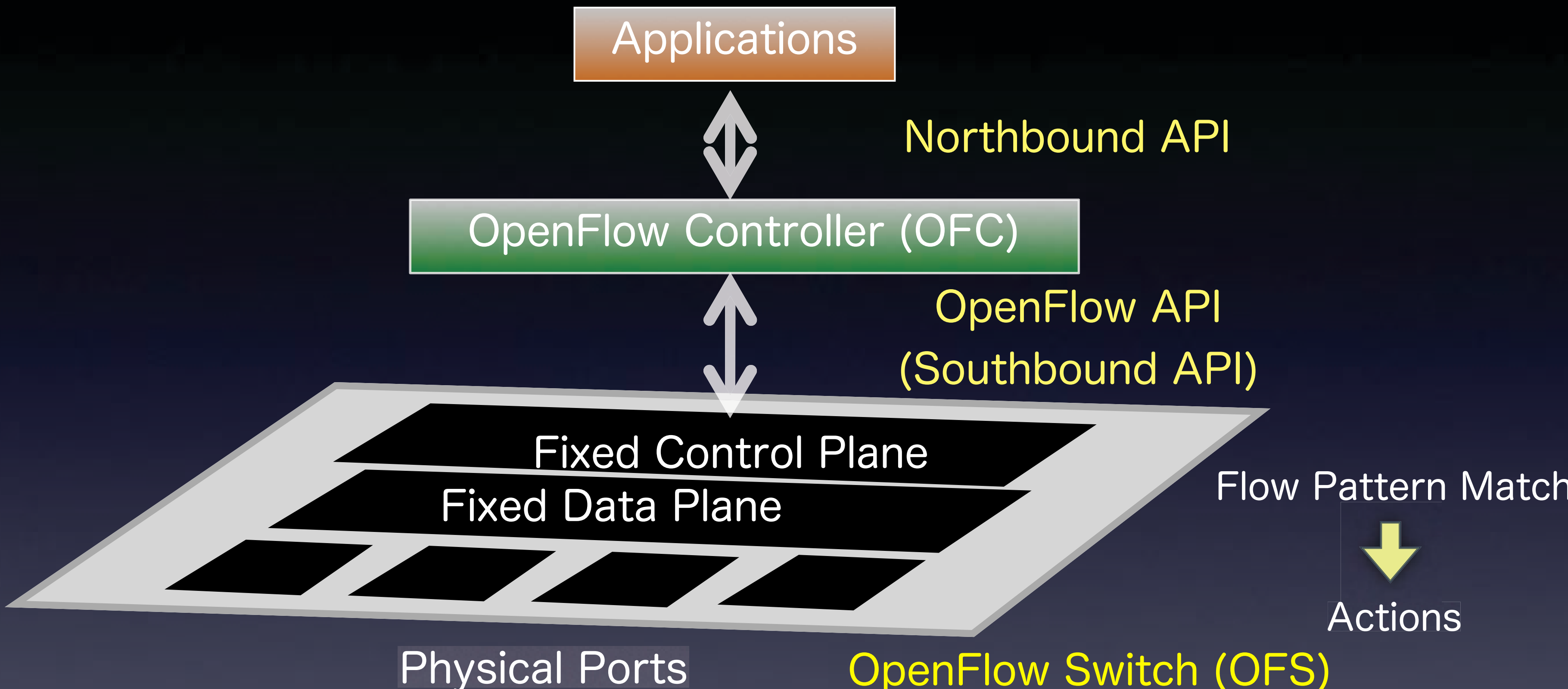
DPN
SDN + Data Plane Programmability

Deeply Programmable Node

Network Virtualization

New Protocol Capability

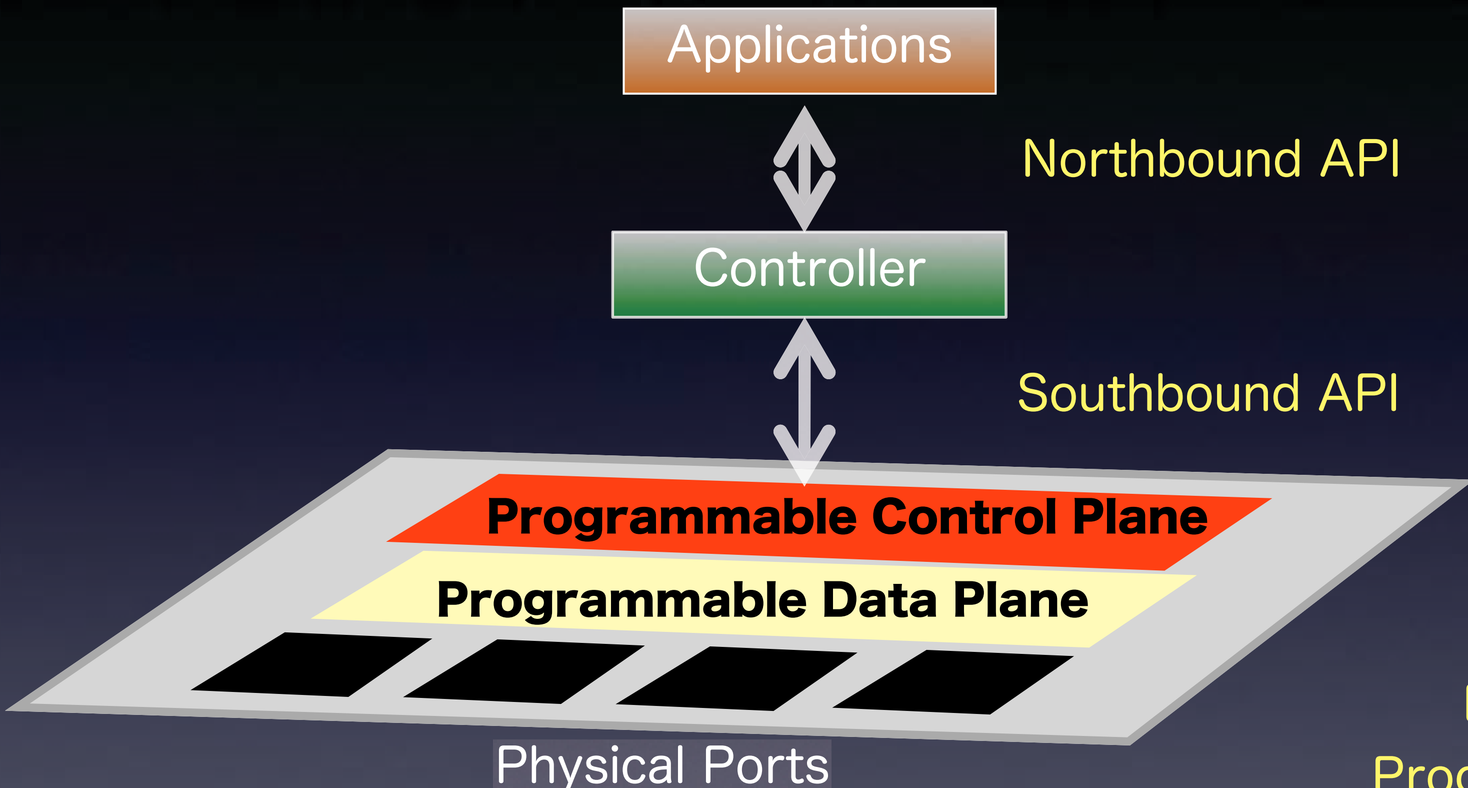
OpenFlow Switch



Although flexible control is achieved to some extent...

• Complex processing not supported	Data-plane programmability
• Non Internet protocols not supported	New protocols
• L7 pattern match not suppoted	New classification
• Proprietary actions cannot be executed	Proprietary actions
• Proprietary API cannot be added	Proprietary APIs (re)definition

Fully Programmable Switch?

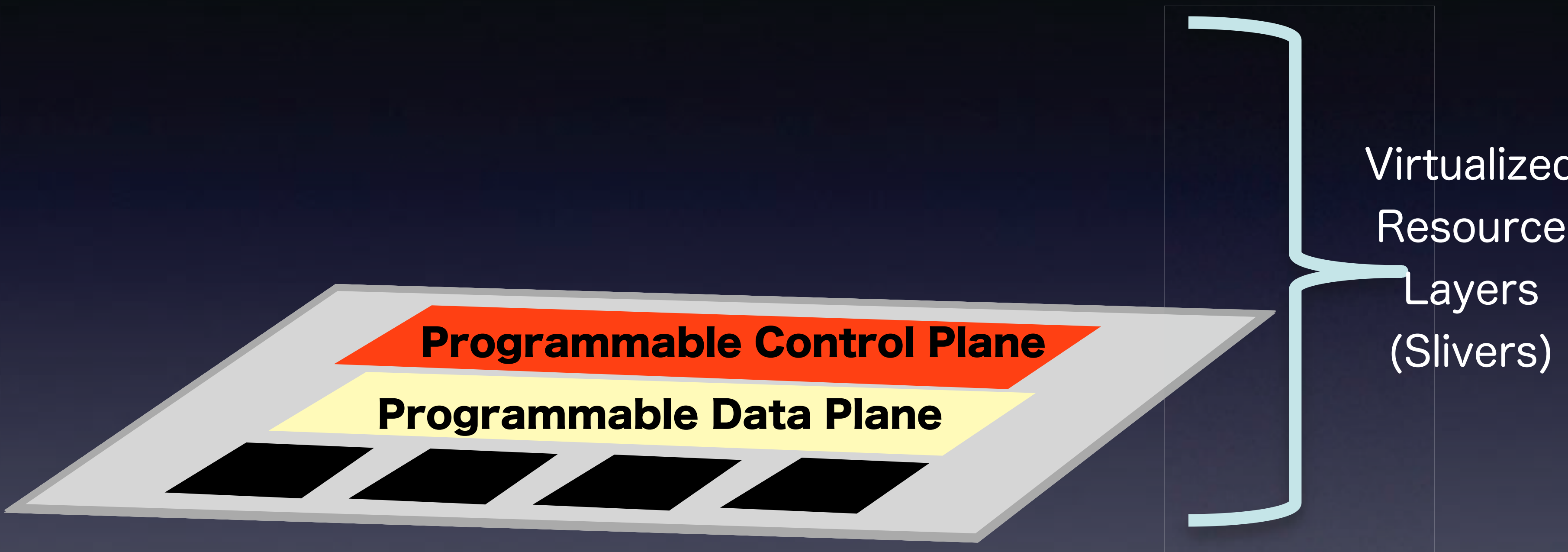


Deeply Programmable Switch

Challenges:

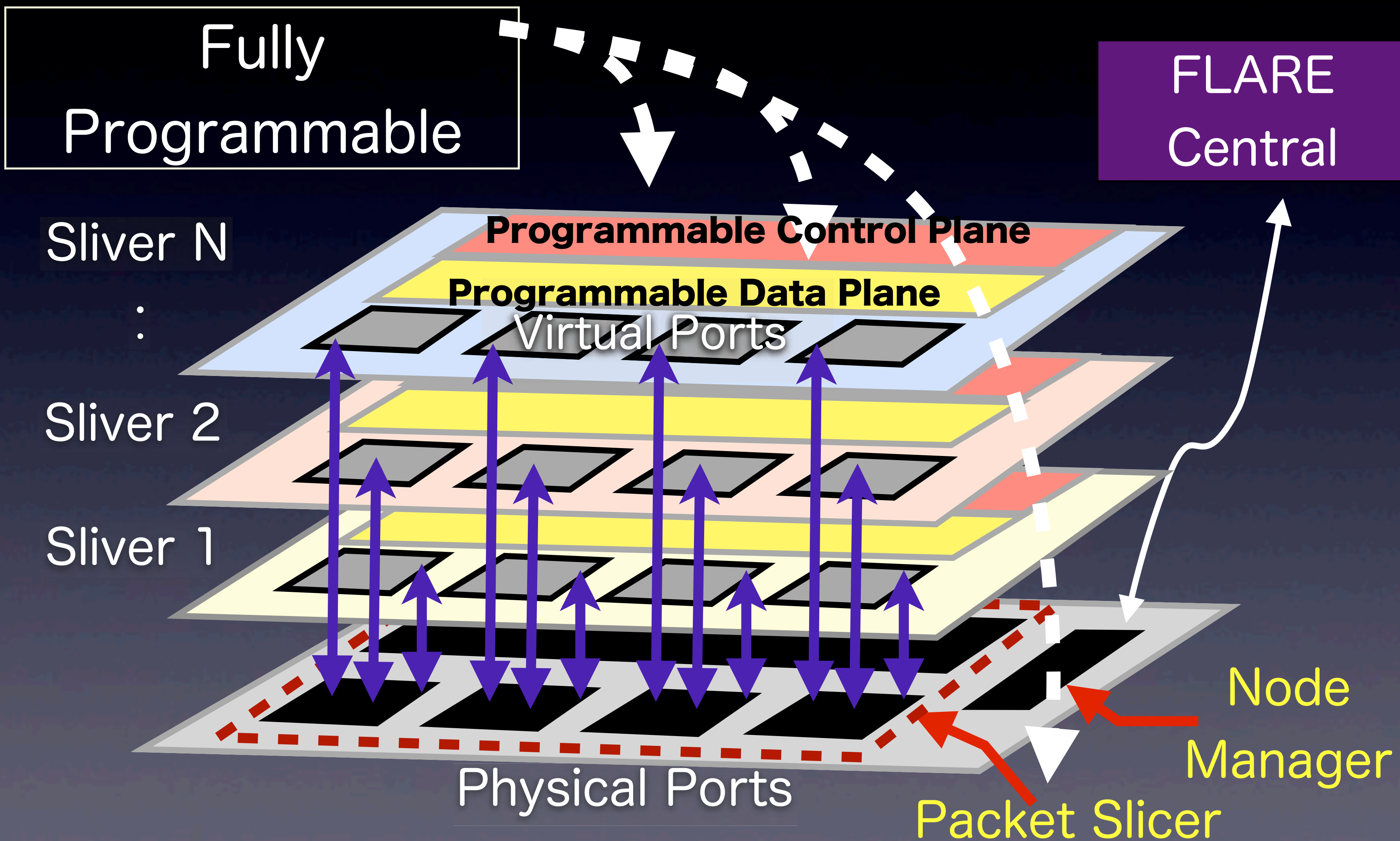
- Tradeoff between performance and flexibility
- Ease of programming
- Supporting multiple protocols/instant switch/concurrent use

Multiple Fully Programmable Layers (Slivers)



"Resource virtualization" within a single node enables multiple switching logics/controls

FLARE Node Architecture
(multiple fully programmable slivers)

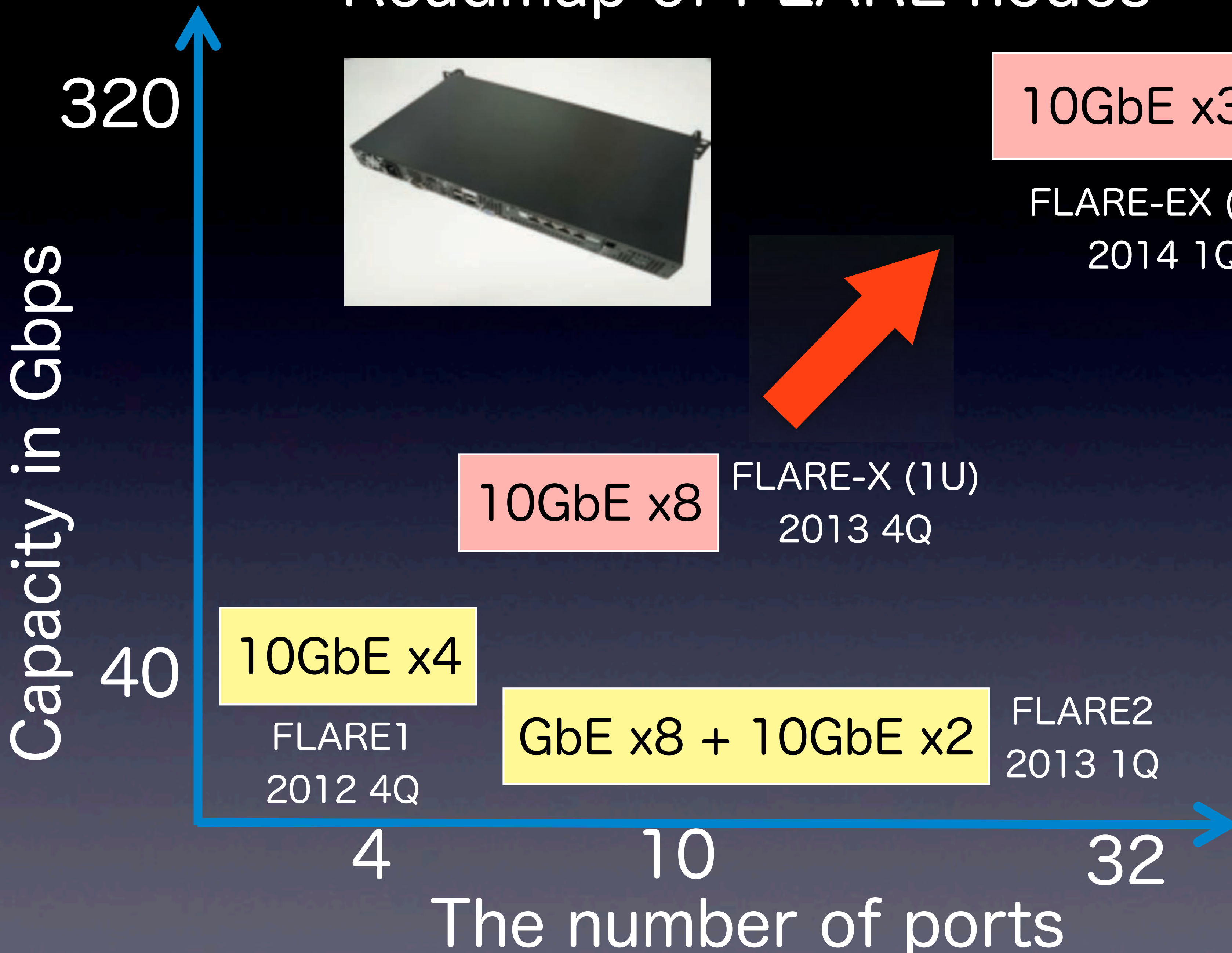


FLARE Node Implementation

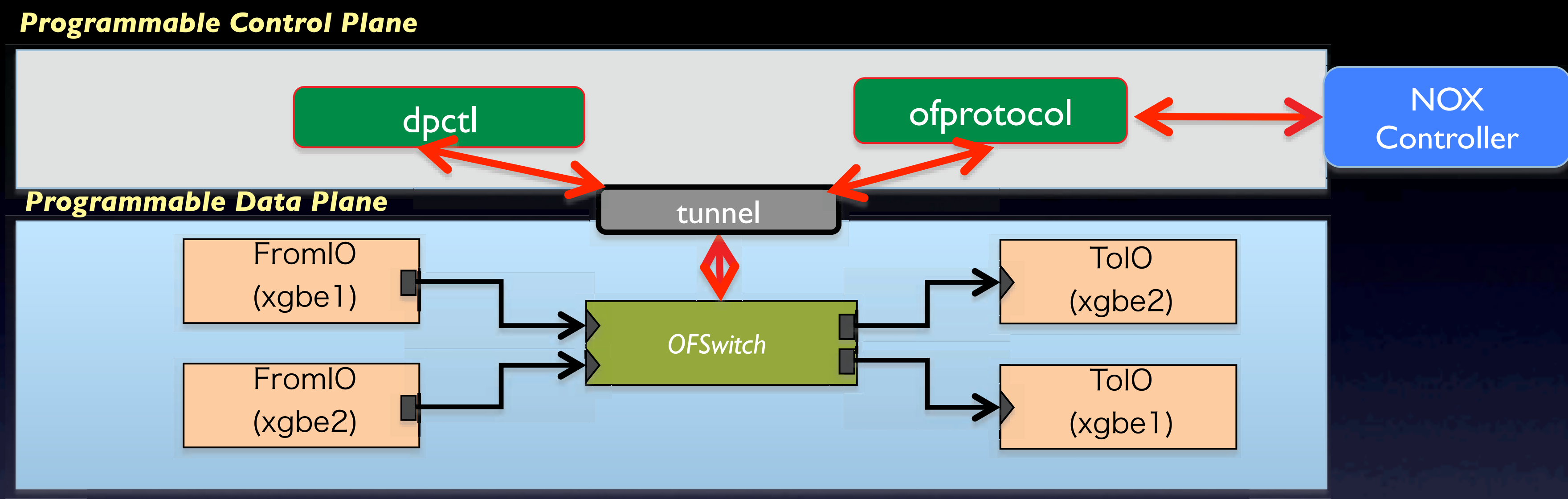
- Multiple, isolated, deeply programmable environments
- OS Virtualization on many-core processor (D-plane) and x86 processor (C-plane)
- Multi 10Gbps ports
- 1U / 1U Mini Form Factor
- Control Plane & Data Plane Linux Programmability
- Flexible programmability and reasonable performance



Roadmap of FLARE nodes



FLARE Programming Model in Sliver

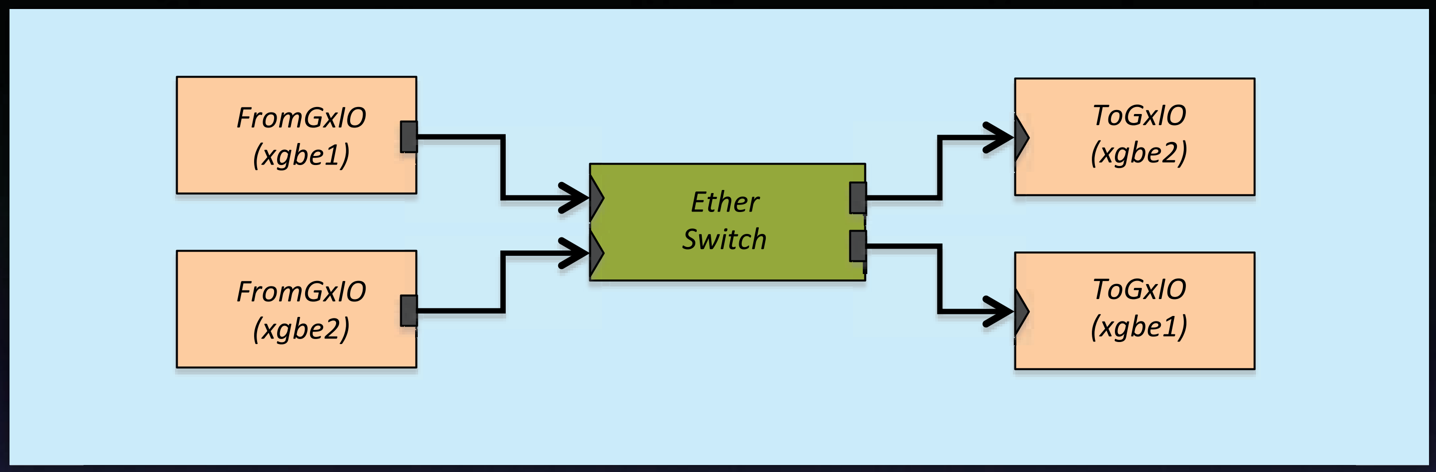


Multi-Threaded Modular Programming

e.g., Click Software Modular Router (multi-threaded)

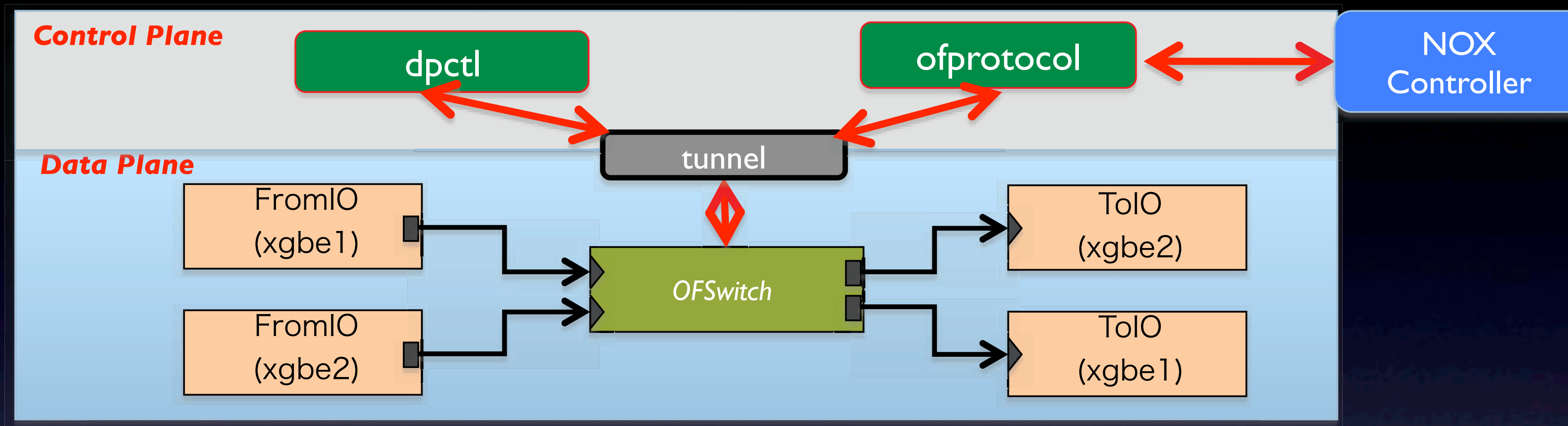
- Arbitrary switch logic(s) can be implemented

Ethernet Switch



Switching Performance

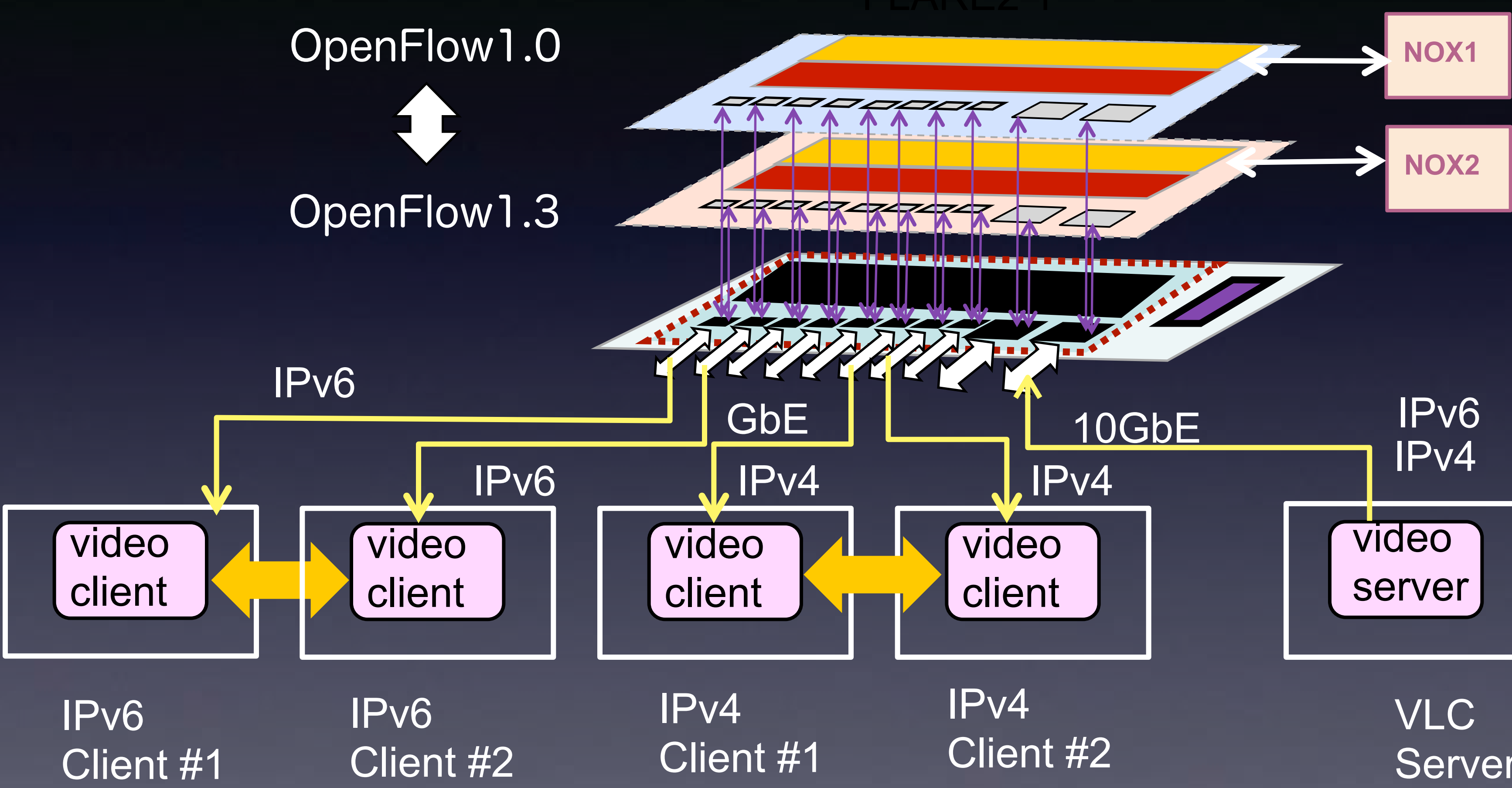
OpenFlow



Switching Performance

What can we do with FLARE
that others cannot do
in a simple manner?

Multiple SDN Logics (OpenFlow 1.3 and OpenFlow 1.0)



Multiple SDN Logics

Purpose:

Dynamically changing SDN control logics for different flow spaces

Benefit:

Instant upgrade/downgrade of switching software

Incremental upgrade while keeping compatibility with old technologies

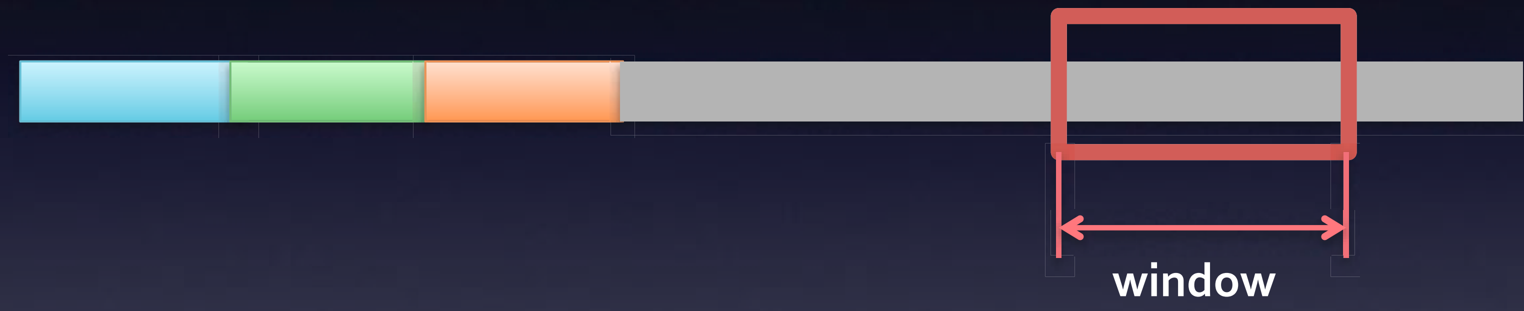
Enable “evolve-able” network architecture

Solution:

FLARE can implement multiple SDN logics (southbound APIs) in slivers

Window-based Arbitrary Bit Matching

Arbitrary bit matching as in openflow pattern matcher is costly due to expensive memory operation per packet



Set a window to minimize per-packet memory operations

Improve performance while keeping flexibility

Leon Lee, Ping Du and Akihiro Nakao, "Ouroboros: SDN Beyond Flow-Tuple Matching," IEICE NS Technical Report, Mar. 2013

Window-based Arbitrary Bit Matching e.g., RTP-SSRC Matching

Purpose:

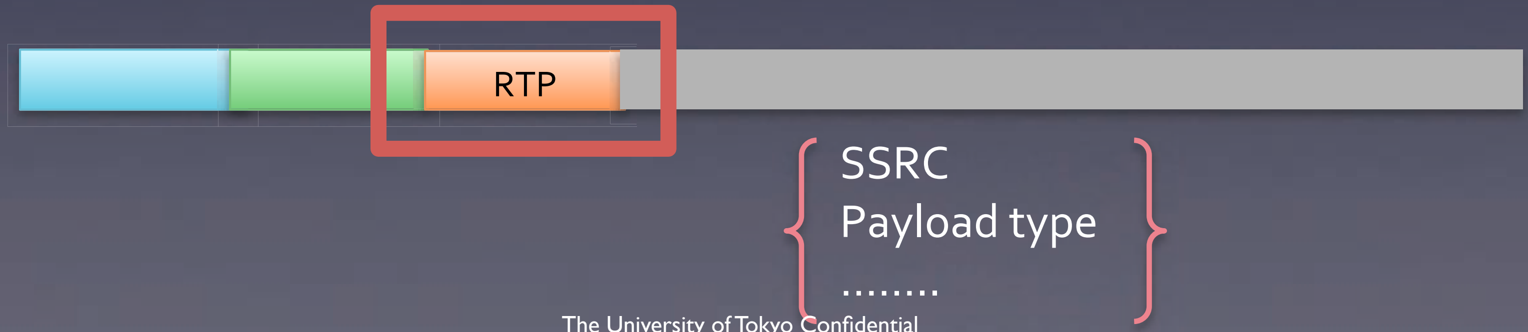
Specific Real-time audio/video traffic control, based on “stream”, not “flow” (e.g. based on RTP SSRC field to enable routing according to the streaming-video IDs)

Benefit:

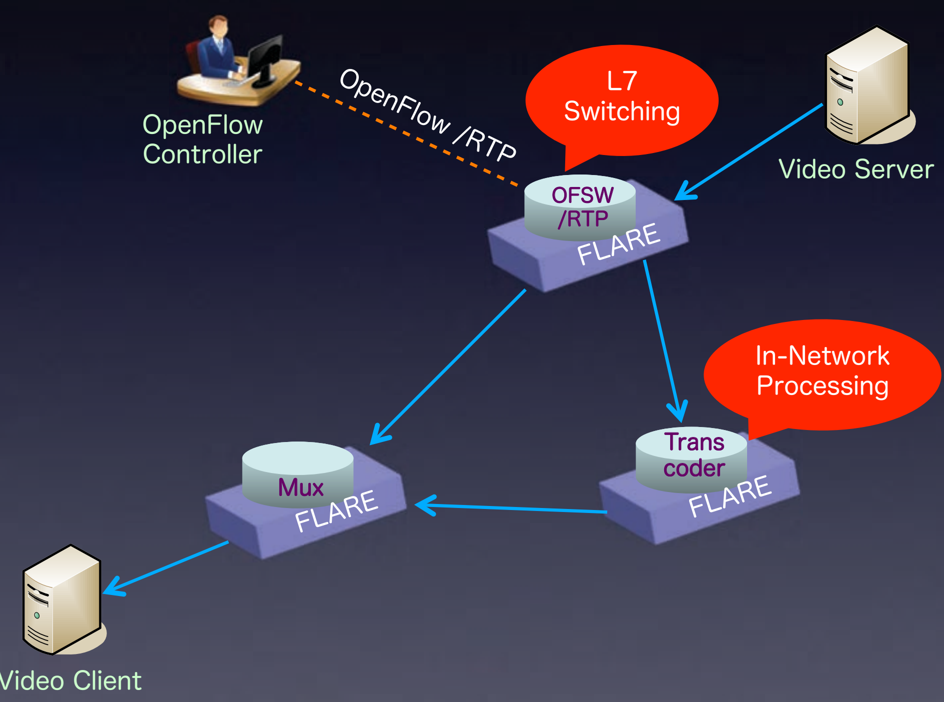
Application/Content specific routing

Solution:

Use window to extract information included in RTP headers



L7 Switching and In-Network Processing



L7 Switching

FLARE supports deeply programmable SDN solutions such as arbitrary-bits and arbitrary offset matching and definition of proprietary APIs achieving both flexibility and performance

In-Network Processing

Video transcoding can be preformed in real time on either D-plane (many-cores processor) or C-plane (Intel-CPU).

Window-based Arbitrary Bit Matching e.g., Trailer Matching

Purpose:

Device/application/content specific traffic engineering

Benefits:

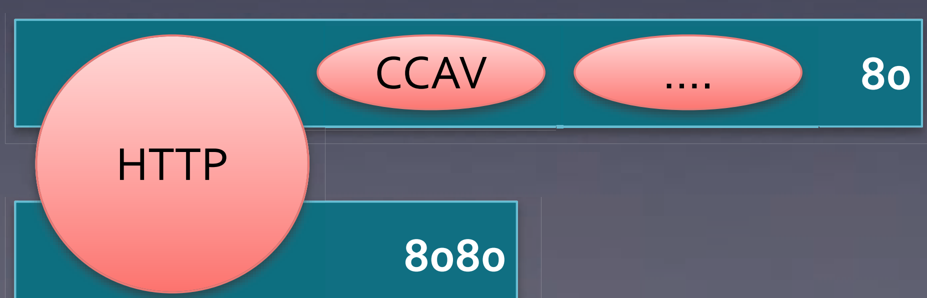
More specific recognition for packets bound to overly used TCP port (e.g., 80)

Traffic engineering for data transmitted from specific devices

Network Virtualization for non-IP protocols!

Solution:

After attach/detach trailers, establish control in intermediate FLARE switches



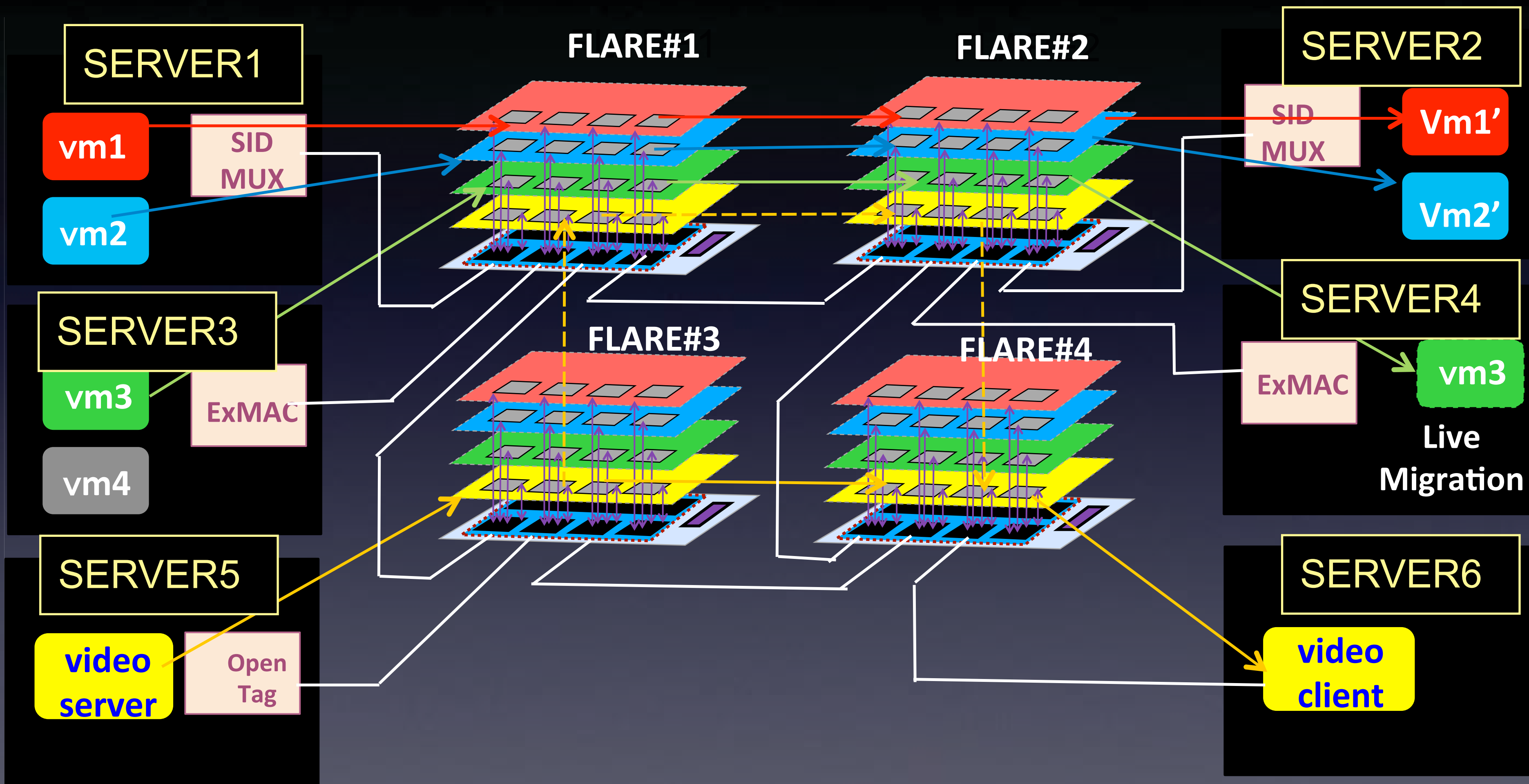
Window-based Arbitrary Bit Matching e.g., Trailer Matching

Trailer is an extra section attached at the end of each packet

End/Edge nodes are responsible for attaching and detaching Trailers

Leon Lee, Ping Du and Akihiro Nakao, "Ouroboros: SDN Beyond Flow-Tuple Matching," IEICE NS Technical Report, Mar. 2013

Network Virtualization with Trailer Slicing



Drastic Examples

L2 Programmability
Extended (96bit) MAC switching

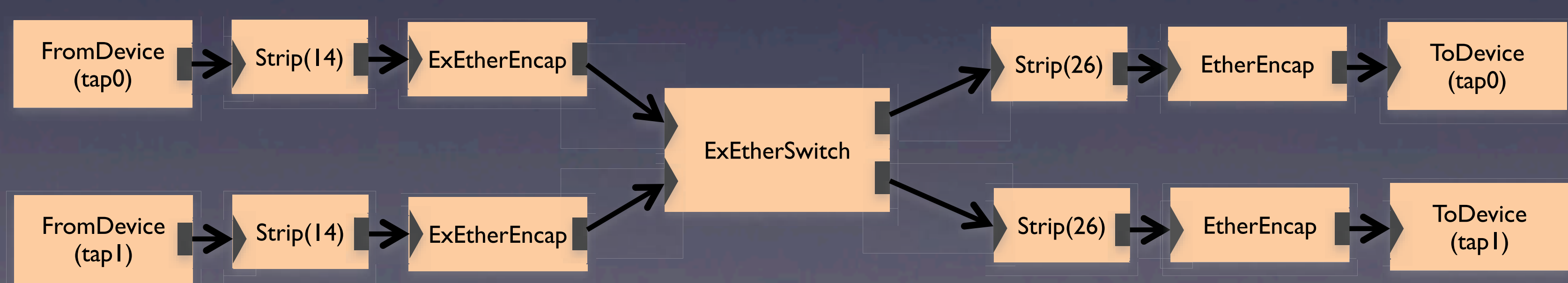
Traditional Ethernet Frame:

DMAC (48bits)	SMAC (48bits)	Type	IP PayLoad

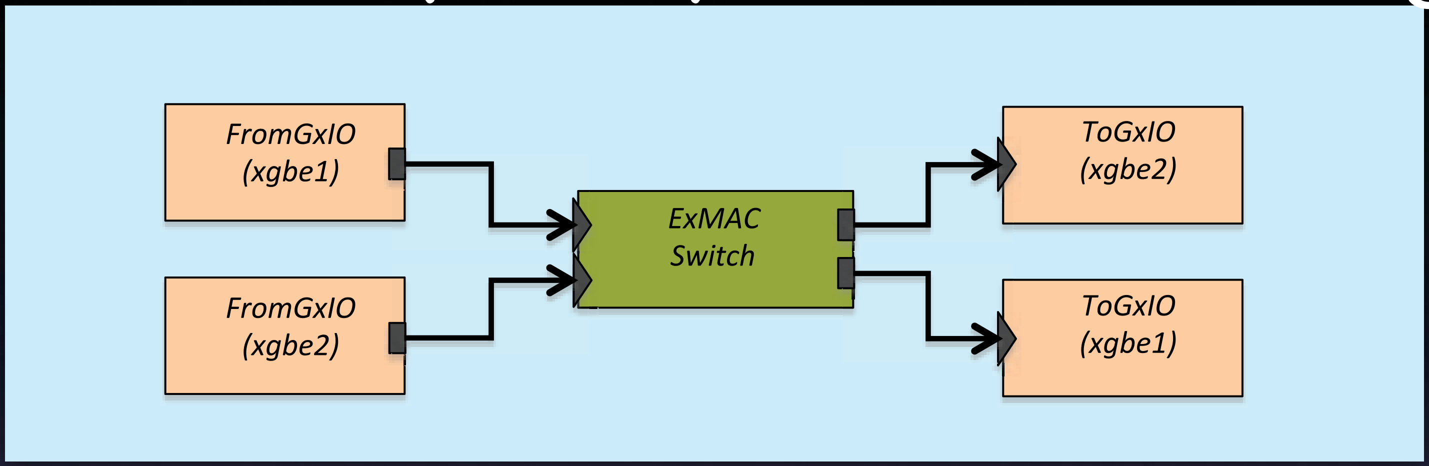
Extended Ethernet Frame with Extended MAC:

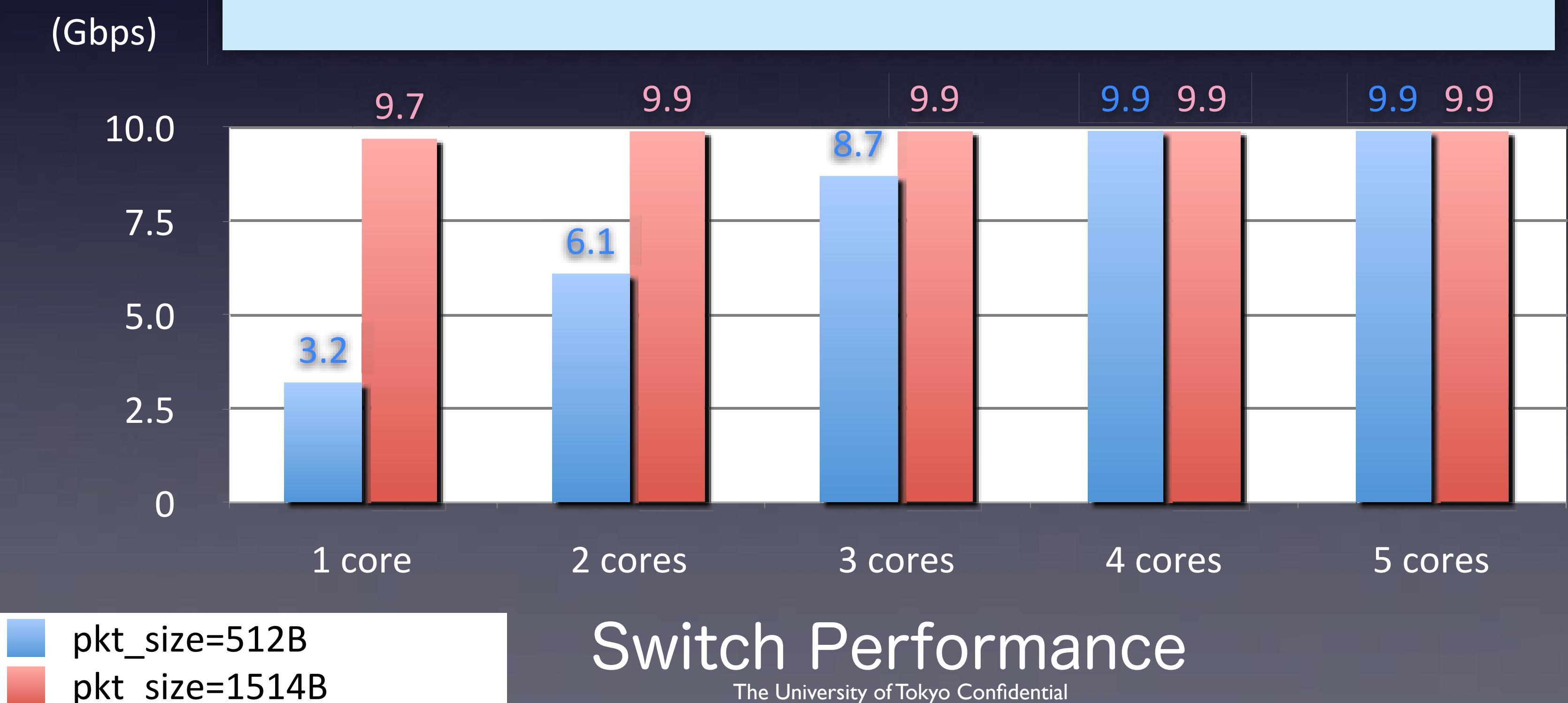
DMAC (96bits)	SMAC (96bits)	Type	IP PayLoad

Prototype with Click



L2 Programmability
Extended (96bit) MAC switching





Switch Performance

L2 Programmability
Extended (96bit) MAC switching

Purpose:

- Demonstrating "Clean-Slate" programmability even for L2 protocols
- Possibly alternative to VXLAN for mitigating MAC address exhaustion for supporting a large number of tenants in data center networks

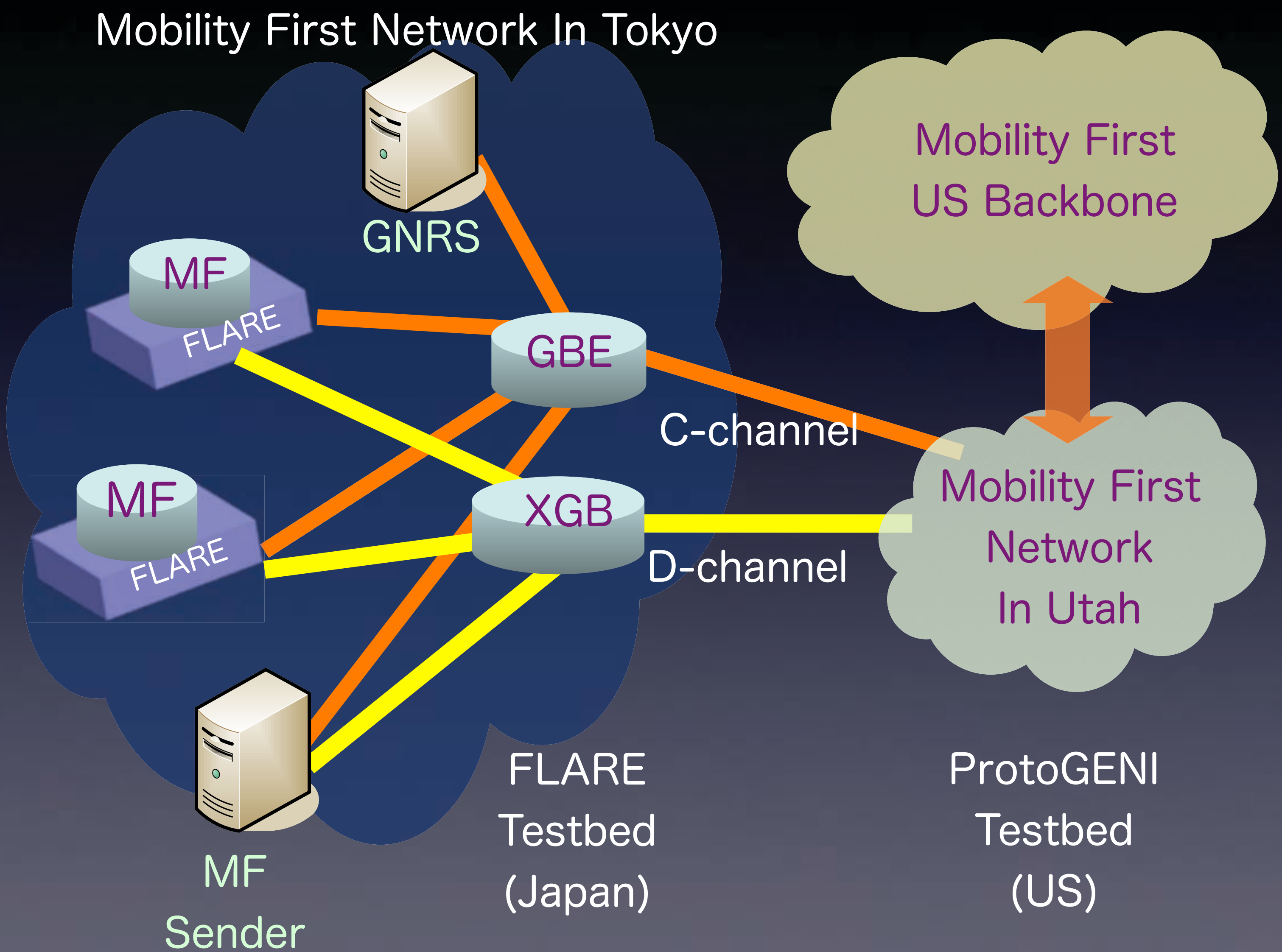
Benefit:

MAC address extension keeping transparence for IP applications

Solution:

FLARE can literally provide "deep programmability" in data-plane, even in L2.

Non-IP protocol



Non-IP protocol

Purpose:

Develop and operate Non-IP protocols over a network

Benefit:

Enable non-IP protocol development for research community

Solution:

FLARE can be used to program data-plane as well as control-plane for Non-IP protocols (that requires data-plane programming)

FLARE at ITPro EXPO 2012

Beyond OpenFlow/SDN

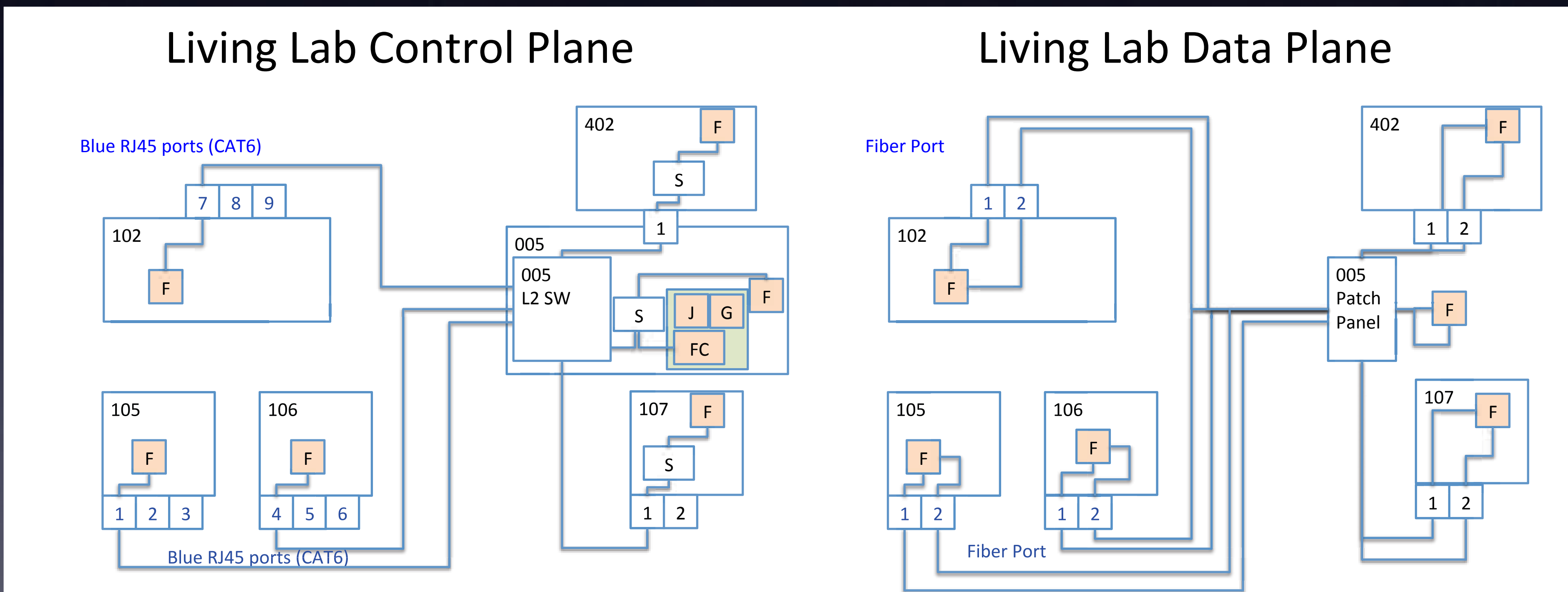
MPLS 2012
(with Cisco & Juniper)

IM2013 Keynote

iPoP 2013 Business Session
Platinum Booth

LivingLab @ NakaoLab

Living with Deeply Programmable FLARE



Conclusion

- Deep Programmability refers to the extensive programmability including Control-plane, Data-plane (including non-IP handling), (re)defining APIs in SDN, etc.
- Deeply Programmable Network research encourages "clean-slate" thinking and redesigning the network and lifts the limitation in traditional networking and even in the current SDN
- Standardization on deep programmability within the network is yet to be done

Credits

- FLARE Project Team @ UTokyo
  - Aki Nakao
  - Shu Yamamoto
  - Ryota Ozaki
  - Ping Du
  - Eiji Miyagaki
  - Haruki Denpo
- NICT & MIC for Funding the Project(s)