Some Considerations on Security and Scalability Issues of Software Defined Networking

Yasuo Okabe
Kyoto University
okabe@i.Kyoto-u.ac.jp

Global Future Internet Summit 2013

Nov. 20 2013

Overview of my talk (1)

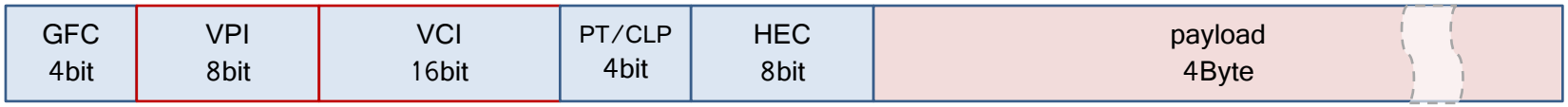
- Future Internet researches aim for an innovative Internet architecture designed with a clean slate, and Software Defined Networking (SDN) is one approach of it.
- OpenFlow, a typical implementation of SDN, separates the control plane and the data plane, and simple forwarding process at the data plane is done by relatively low-cost commodity switches.
- As the deployment of SDN goes, there have been indicated some serious issues to be solved on security and scalability of SDN.

Overview of my talk (2)

- As is often said "deja-vu", the architecture of OpenFlow has strong similarity with ATM LAN emulation in 1990s in some points.
- In this presentation, we first take a look on the history of flow-based networking like ATM and MPLS, and give some thoughts on what are new and what are not among the issues of SDN.
- Then we introduce some recent research works including ours on security and scalability of OpenFlow.

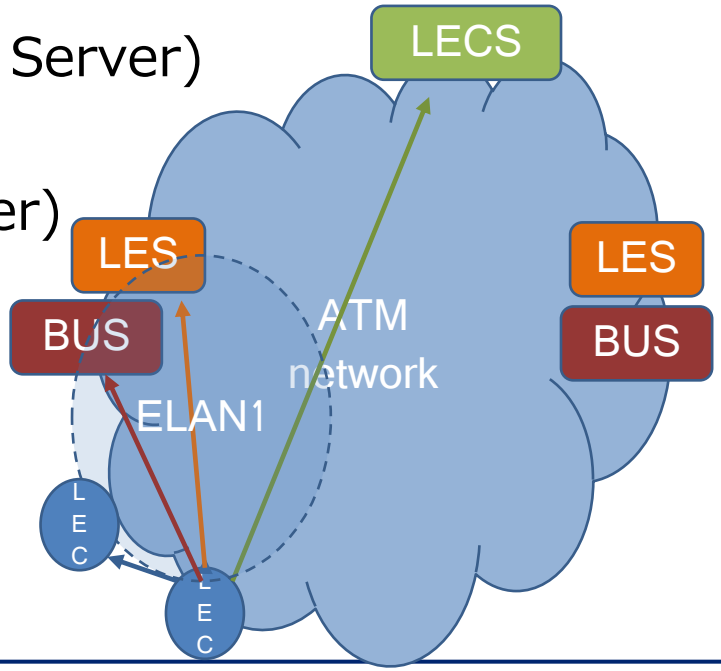
Back to 1990s: What was ATM?

- ATM (Asynchronous Transfer Mode) is a carrier-oriented networking technology
  - Developed and deployed in 1990s, as a “next-generation” technology of telecom networks
  - Called B-ISDN (Broadband ISDN), an enhancement of ISDN
- Features of ATM:
  - Connection-oriented
    - Virtual circuit (or “flow” in the terminology of IP networking)
  - End-to-end signaling
    - 20Byte (160bit) global hierarchical addressing
  - Short fixed-length packet (cell)
    - 53Byte (5+48)



Back to 1990s: What was ATM LAN Emulation (LANE)?

- Emulating Ethernet LAN by ATM
  - Connectionless L2 (Ethernet) over connection-oriented L3 (ATM)
  - Emulating Ethernet broadcast by NBMA (Non-broadcast multiple access) ATM network
- ATM LANE components:
  - LECS (LAN Emulation Configuration Server)
  - LES (LAN Emulation Server)
  - BUS (Broadcast and Unknown Server)
  - LEC (LAN Emulation Client)
  - ELAN (Emulated LAN)
    - Virtual LAN
- Configuration is stored at LECS
  - Centralized control



Back to 1990s: What had happened at Kyoto U ATM LAN?

- KUINS-II/ATM (1996-2002):
  The world largest private ATM network at Kyoto University
  - > 300 ATM switches
    - 4-layer P-NNI
  - > 100 LES/BUS, > 200 ELAN, > 1000 LEC
- Centralized control scheme reduced the daily operational cost
  - It had been working fine when it had no trouble, but…
- Once a trouble happened, it might cause a campus-wide melt down of the whole network
  - Troubles might happen on either physical layer, ATM layer, LANE layer, IP layer or upper layers, but ATM LES/BUS was one of the weakest point against overloads and DoS attacks.
  - It is not easy to detect the cause of a trouble on fully virtualized networks
    - Conventional trouble-shooting techniques may not work.

MPLS, as the successor of ATM

- MPoA (Multiprotocol over ATM)
  - proposed as the State-of-Art technology for ATM Intranet, integrating RSVP, NHRP, LANE, flow switching, WWW, …
- MPLS (Multi-Protocol Label Switching)
  - Motivated by ATM networking, *Tag Switching*, a flow-based IP networking technology is proposed and standardized as MPLS
  - MPLS inherits the label switching technology of ATM
    - Originally designed for the convergence of connection-oriented ATM network and connectionless IP networking
  - MPLS is not used as originally intended but is utilized in Traffic Engineering and L2/L3 VPN at backbone networks operated by carriers.
    - MPLS-TE
- GMPLS (Generalized MPLS)
  - As the next-generation networking

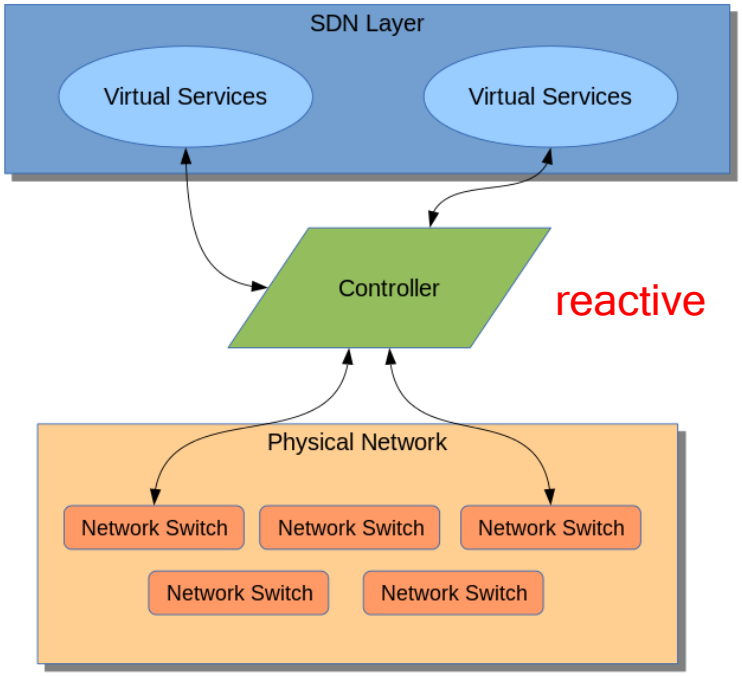
Telecom people love centralized stateful technologies.

Software-defined Networking

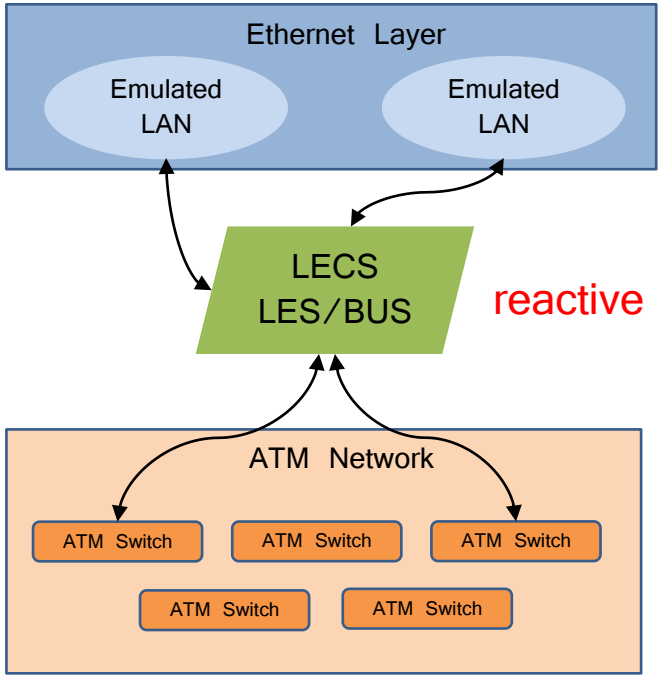
- Software-defined networking (SDN) allows network administrators to manage network services through abstraction of lower level functionality.
- This is done by decoupling the system that makes decisions about where traffic is sent (the control plane) from the underlying systems that forwards traffic to the selected destination (the data plane).

SDN and ATM LANE

- SDN



- ATM LANE

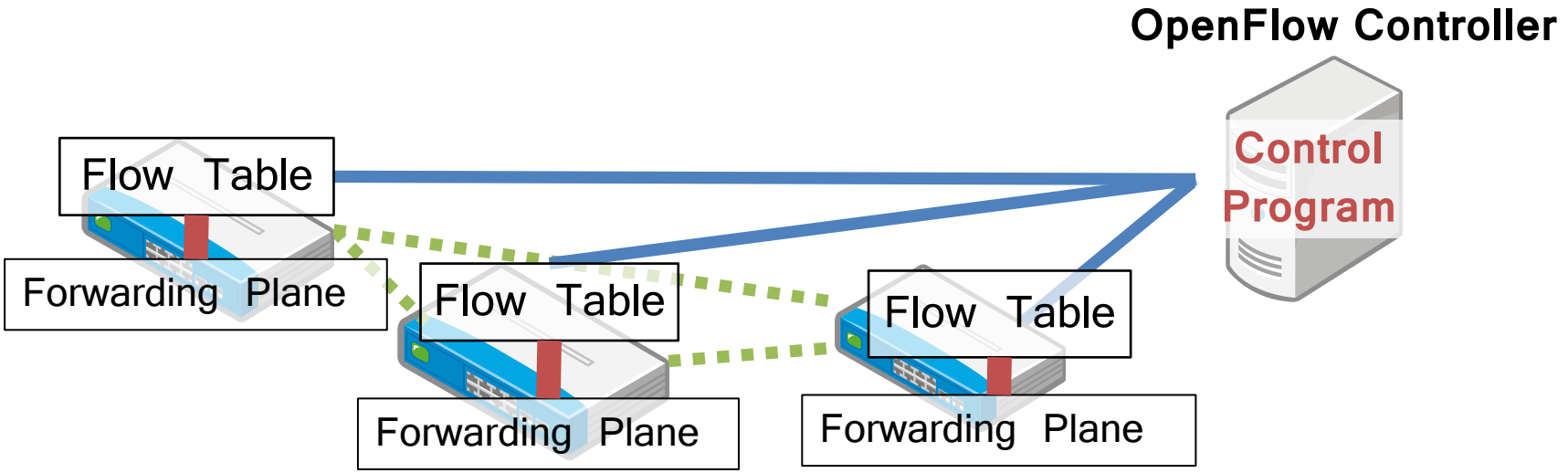


The controller acts as an interface between the physical network and the SDN layer (Wikipedia "Software-defined Networking")

De-ja vu

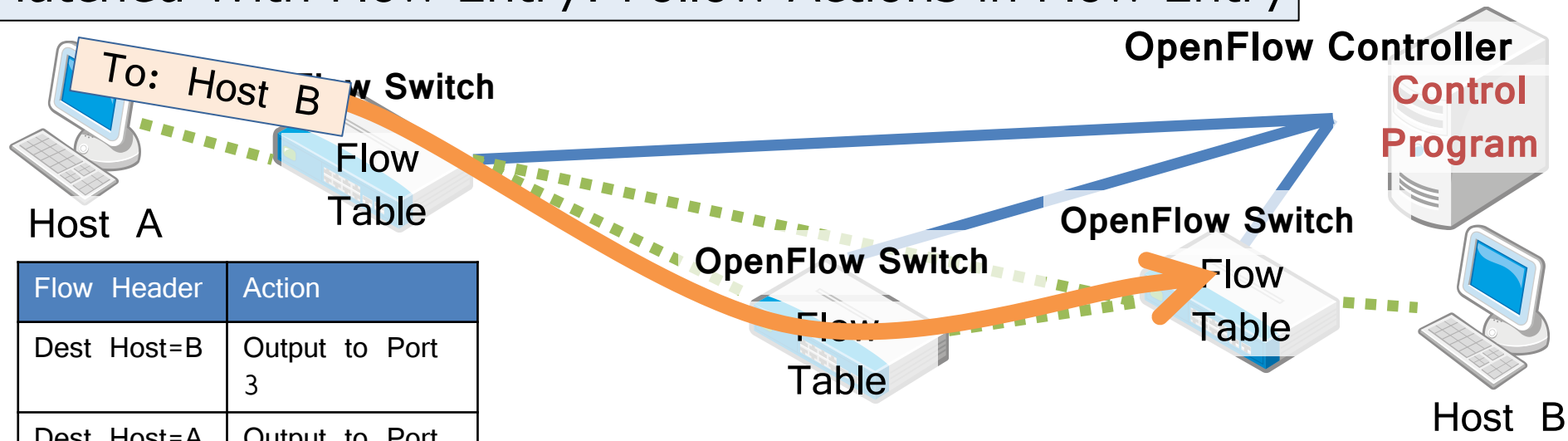
OpenFlow

- OpenFlow:
  An implementation of Software-Defined Networking
  - Flow Tables allow us to define how to process packets in switches
  - Centralized Management of Flow Tables in a Controller



Packet Forwarding in OpenFlow

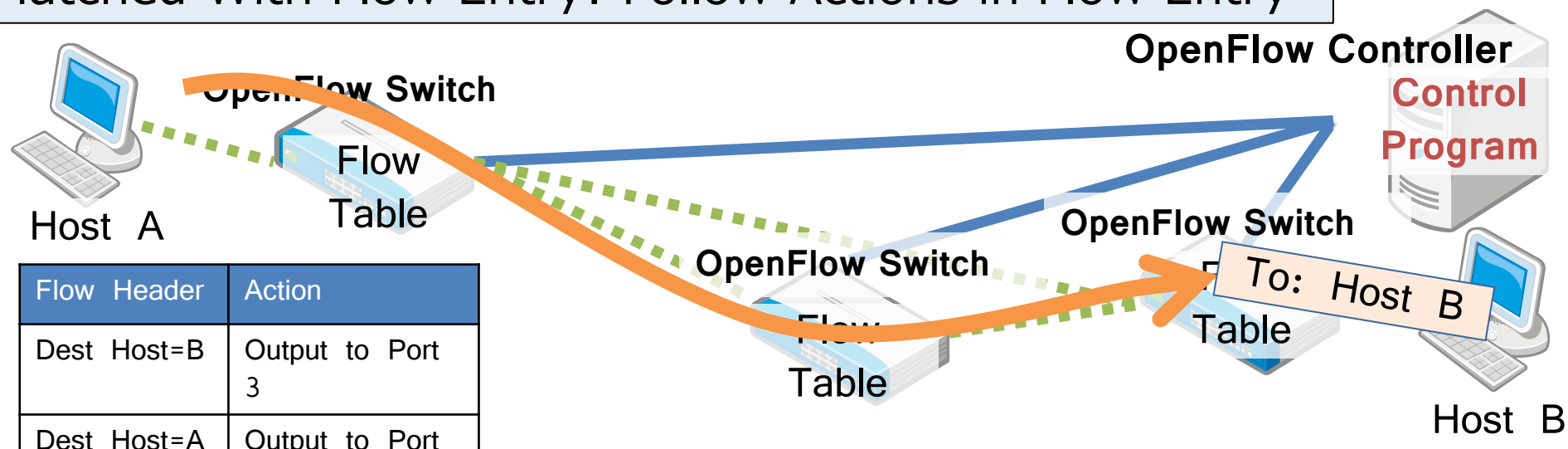
Matched With Flow Entry: Follow Actions in Flow Entry



Flow Header	Action
Dest Host=B	Output to Port 3
Dest Host=A	Output to Port 1

Packet Forwarding in OpenFlow

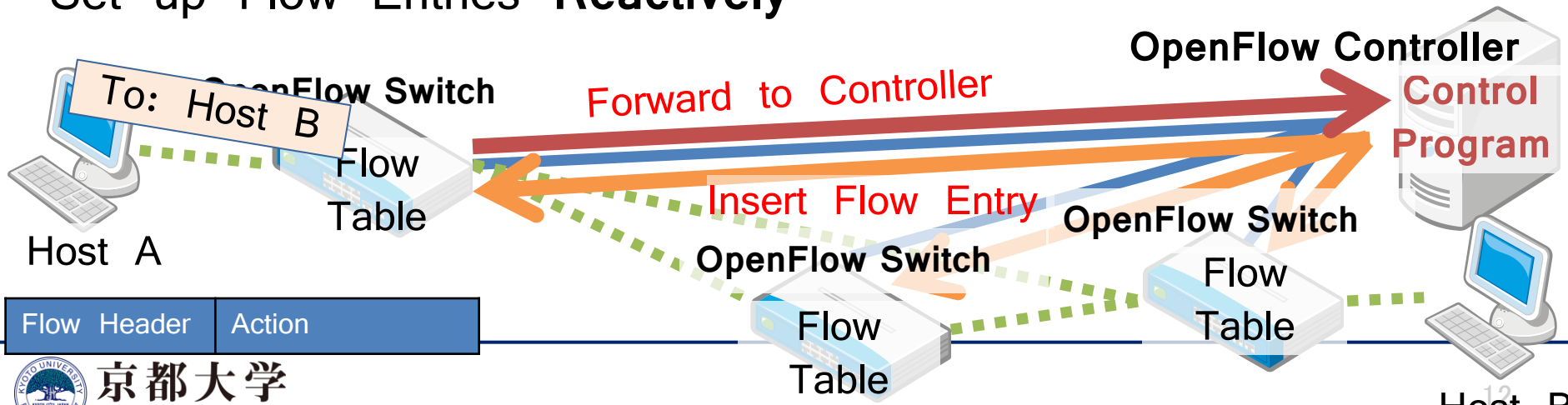
Matched With Flow Entry: Follow Actions in Flow Entry



Flow Header	Action
Dest Host=B	Output to Port 3
Dest Host=A	Output to Port 1

Missing Flow Entry: Forward to Controller as Packet-In message

Set up Flow Entries **Reactively**



Flow Header	Action

“The Top Five Security Considerations for Software Defined Networking”

http://www.sdncentral.com/technology/the-top-five-security-considerations-for-software-defined-networks-sdns/2012/02/

By Sarah Sorensen Posted: Feb. 27, 2012

1. Secure the **controller**
2. Protect the **controller**
3. Establish trust between the **controller** and the applications and devices
4. Create Robust Policy Framework
5. Forensics and Remediation

Recent Works on Security and Scalability Issues of OpenFlow

- Kevin Benton, L. Jean Camp, Chris Small
  "OpenFlow Vulnerability Assessment"
  HotSDN'13, August 16, 2013, Hong Kong
  - Widespread failure to adopt TLS for the OpenFlow control channel by both controller and switch vendors, leaving OpenFlow vulnerable to man-in-the-middle attacks
  - Highlight the classes of vulnerabilities that emerge from the separation and centralization of the control plane in OpenFlow
- Seungwon Shin, Guofei Gu,
  "Attacking Software-Defined Networks: A First Feasibility Study"
  HotSDN'13, August 16, 2013, Hong Kong
  - Demonstrate effective and efficient resource consumption attacks to OpenFlow controllers

Our previous work on this issue

- Daisuke Kotani, Yasuo Okabe,
  "Packet-In Message Control for Reducing CPU Load and Control Traffic in OpenFlow Switches"
  European Workshop on Software Defined Networks (EWSDN2012) pp.42-47, October, 2012.
  - A Problem by many Packet-In messages of a ***heavy* flow**
    - Many Packet-Ins are sent to a controller before a flow entry is set (consume CPU/Memory/bandwidth for management)
    - Limiting the overall bandwidth of Packet-Ins is not a good idea
      - That drops many Packet-Ins of ordinary flows
  - Categorize Packet-In messages based on the processing in controllers
    - State Change, Flow Setup (Important), Forward (Less Important)
      - Most of Packet-Ins by "Heavy Flows" would be in "Forward"
  - Propose a method to limit Packet-Ins in "Forward" group
    - Switches record flows whose packets are sent to the controller

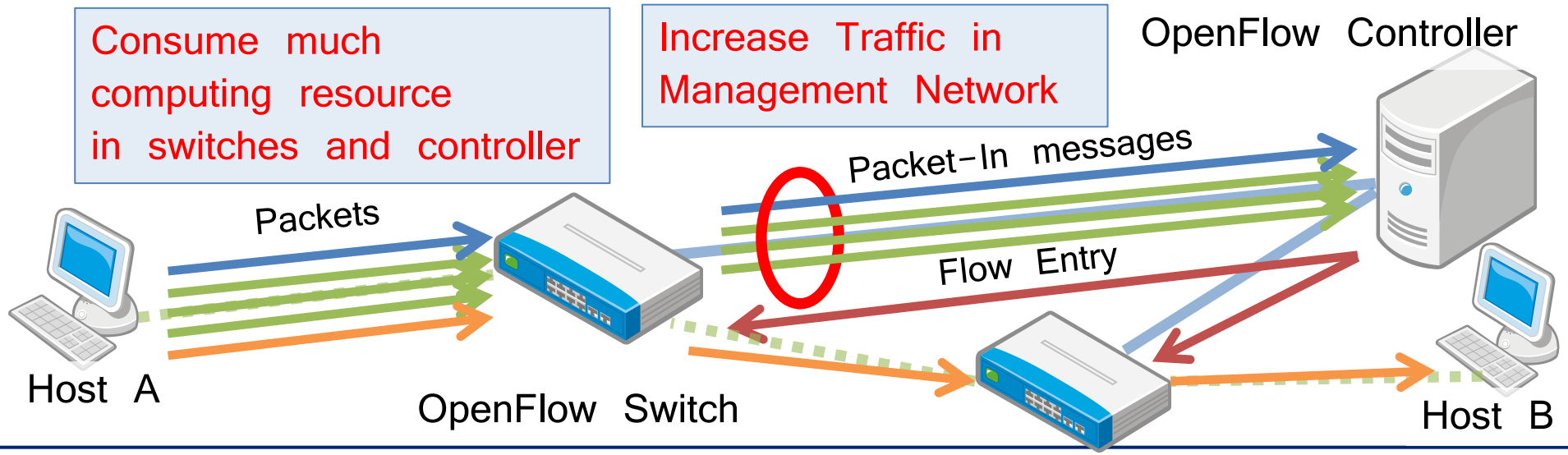
A Problem caused by many Packet-In messages associated with a heavy flow

Set up flow entries *reactively* & a host sends a *heavy flow*

↓

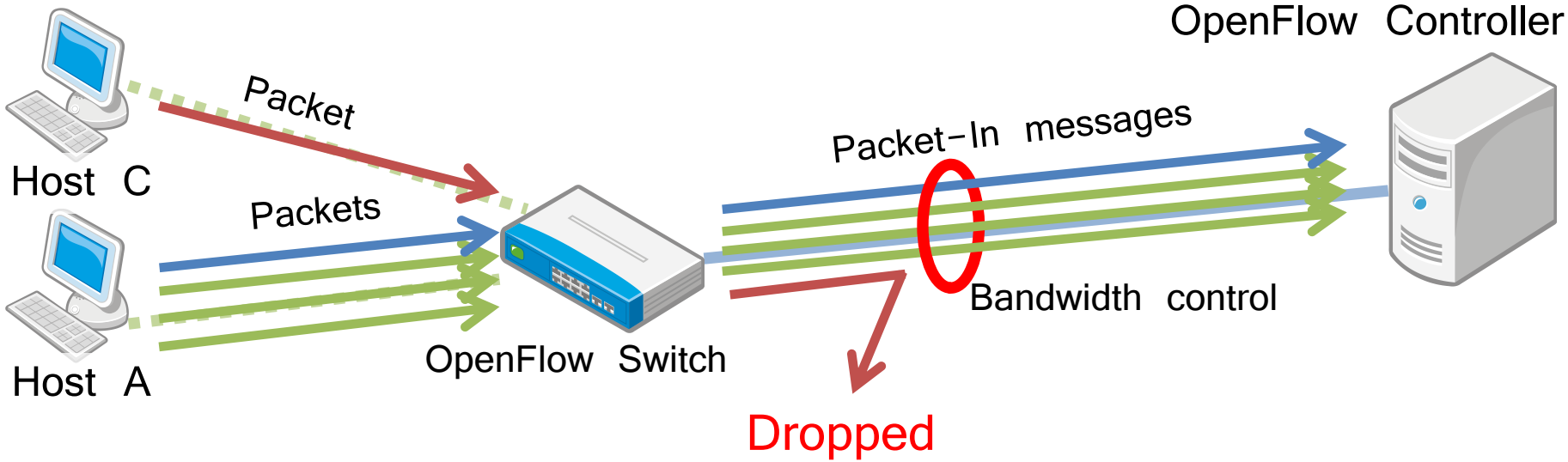
Many Packet-In messages are sent to the controller

- Heavy flow: like video streams over RTP
  - A host suddenly starts to send many packets in the same flow without any negotiation in advance



A Problem caused by many Packet-In messages

- Ordinary Flows: like TCP sessions
  - Establish a session between hosts
- Limiting the overall bandwidth of Packet-In messages drops Packet-In messages of ordinary flows
  - Delay for insertion of flow entries of such flows



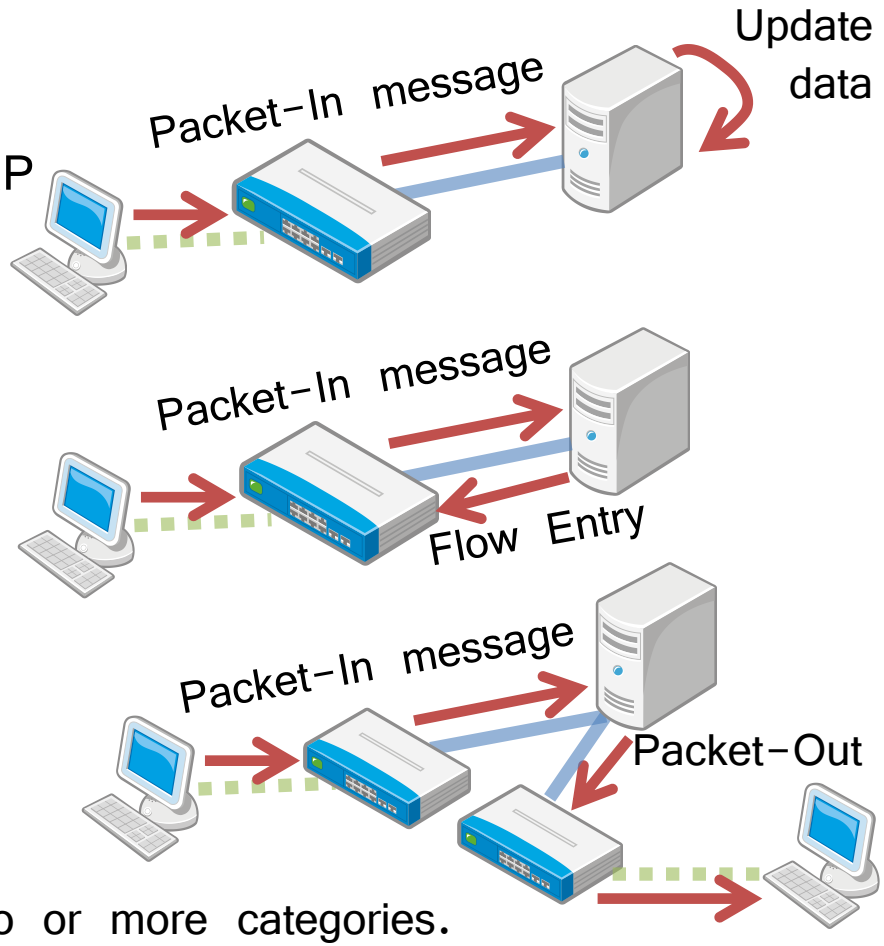
We need a method to limit only Packet-In messages from "Heavy Flows" in switches

Categorization and Discrimination of Packet-In messages

① **State Change:** **Important** for control
Change data in controllers
Ex: Obtain location of hosts using ARP
Set Flow Entries manually
to send to controllers in advance

② **Flow Setup:** **Important** for control
Trigger insertion of flow entries
The first packet of a flow

③ **Forward:** **Less Important** for control
Forward packets to other switches
(Most of packets in Heavy Flows)
Second or later packets of a flow



Note: A message would be categorized to two or more categories.

Propose a method to limit only "③ Forward" in switches

Related proposals for the issue

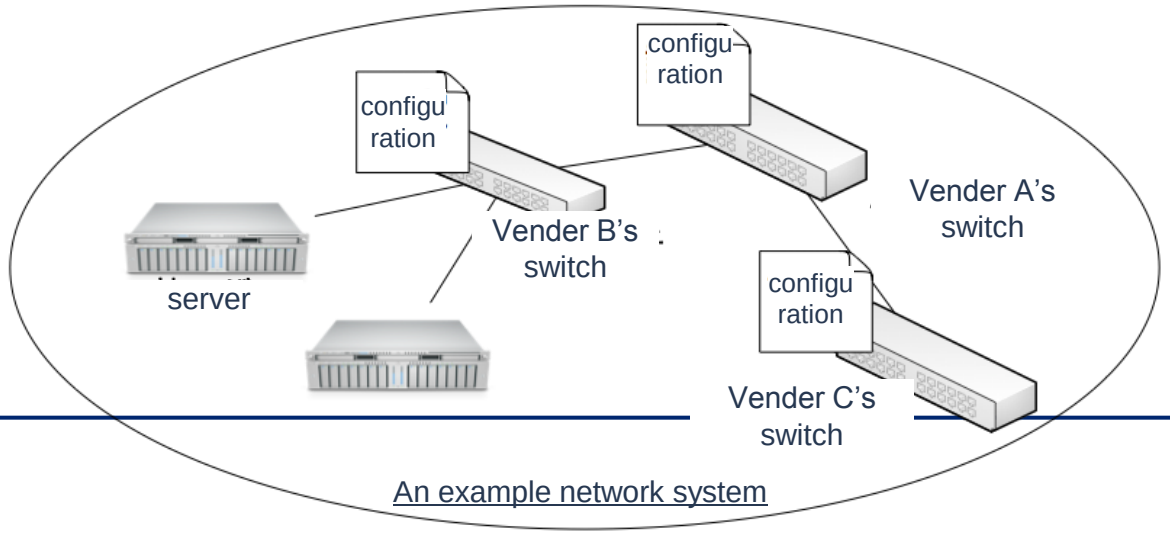
- Improve performance and stability of controllers
  - ONIX [Koponen 2010], HyperFlow [Tootoonchian 2010]
  - Loads of switches and management networks would not be reduced
- Enable more complex operations in the forwarding plane
  - DIFANE [Yu 2010], DevoFlow [Curtis 2011]
  - Packet-In messages would be reduced

- Meter (introduced in OpenFlow 1.3)
  - Bandwidth / Rate limit mechanism in OpenFlow
  - Can control the rate/bandwidth of Packet-In messages per pre-defined group of flows

Reconsideration on the Background and Motivation of SDN

Background
- Increase of the numbers of devices which one administrator manages, and Increase of the complexity and variety of features of them as well
- Most systems are designed as large-scale distributed systems.

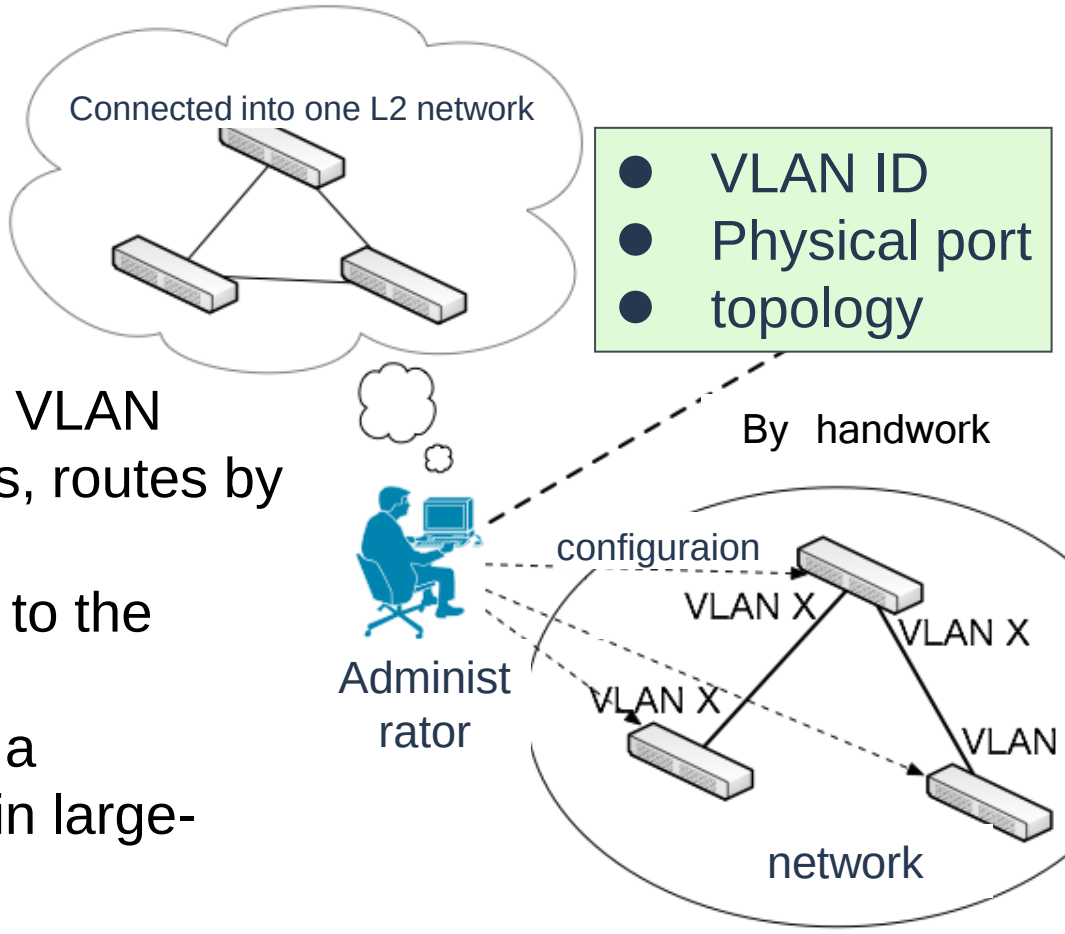
Issues
- The administrator of a system must be conscious of integrity of the configuration of each device at any modification or reconfiguration of it.
- The style of configuration and the features each device has differ.



An example network system

Issues in managing conventional large-scale campus/enterprise network systems

- Configuration is distributed in devices on a LAN
  - The administrator has a responsibility to keep the integrity and the consistency of the distributed configuration



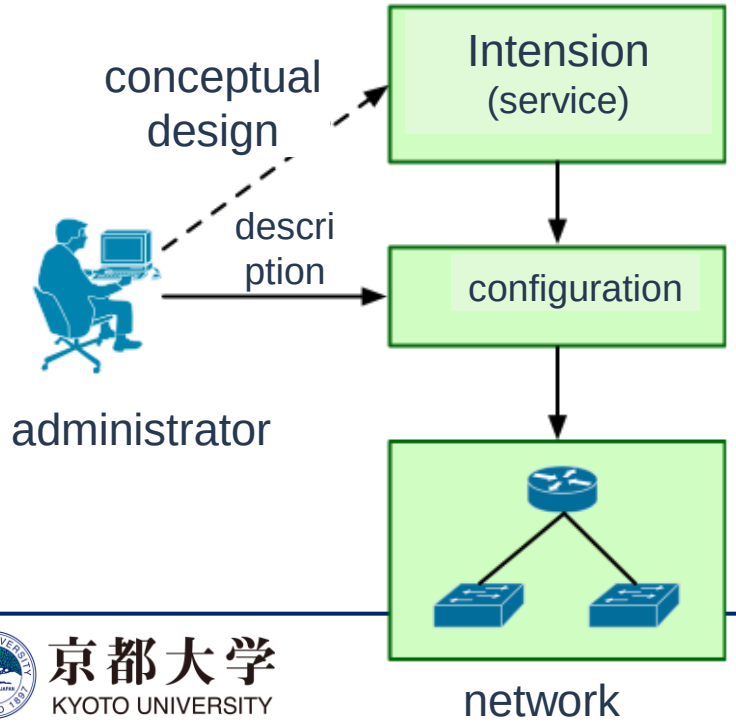
- Example：Configuration of a VLAN
  - Assign a VLAN ID, ports, routes by handwork
  - Set the configuration up to the devices each
- Integrity and consistency of a configuration is hardly kept in large-scale networks.

Proposal of Service Defined Network (SvDN)

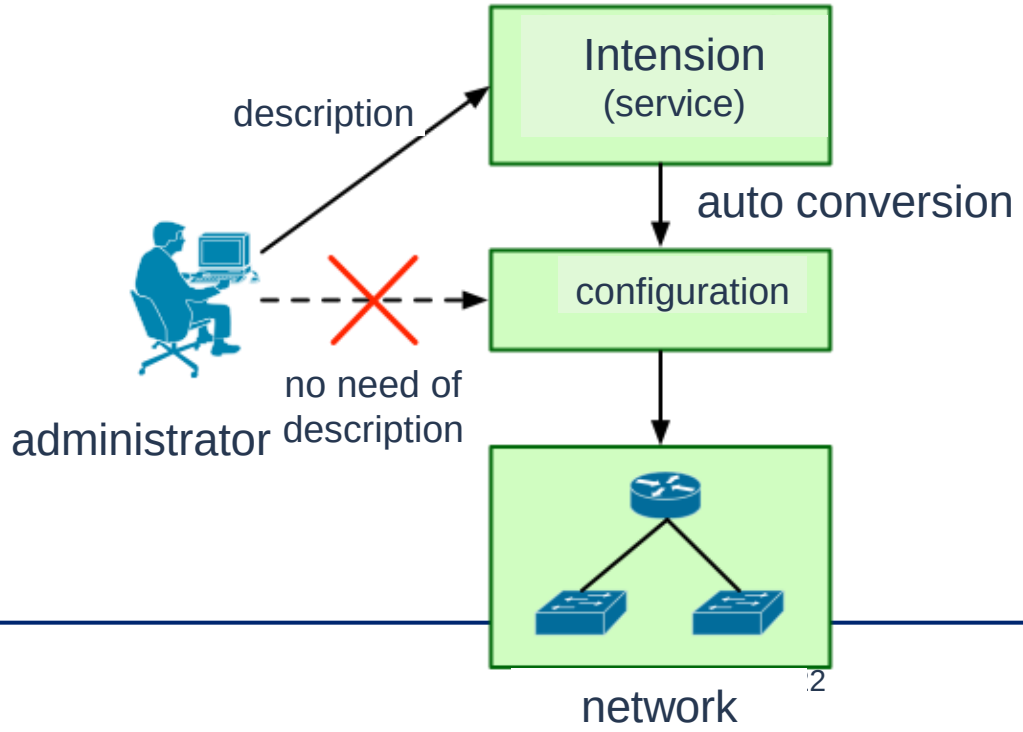
A framework for managing large-scale network as a conceptual extension of SDN

Yasuhiro Teramoto, Rei Atarashi, Yoshifumi Atarashi, Yasuo Okabe,
“Managing Networks Independently of the Physical Topology by Service Defined Network”
ADMNET2013, in Proc. IEEE COMPSAC2013 Workshops）
July 2013.

Conventional network management:
An administrator describe configurations by handwork

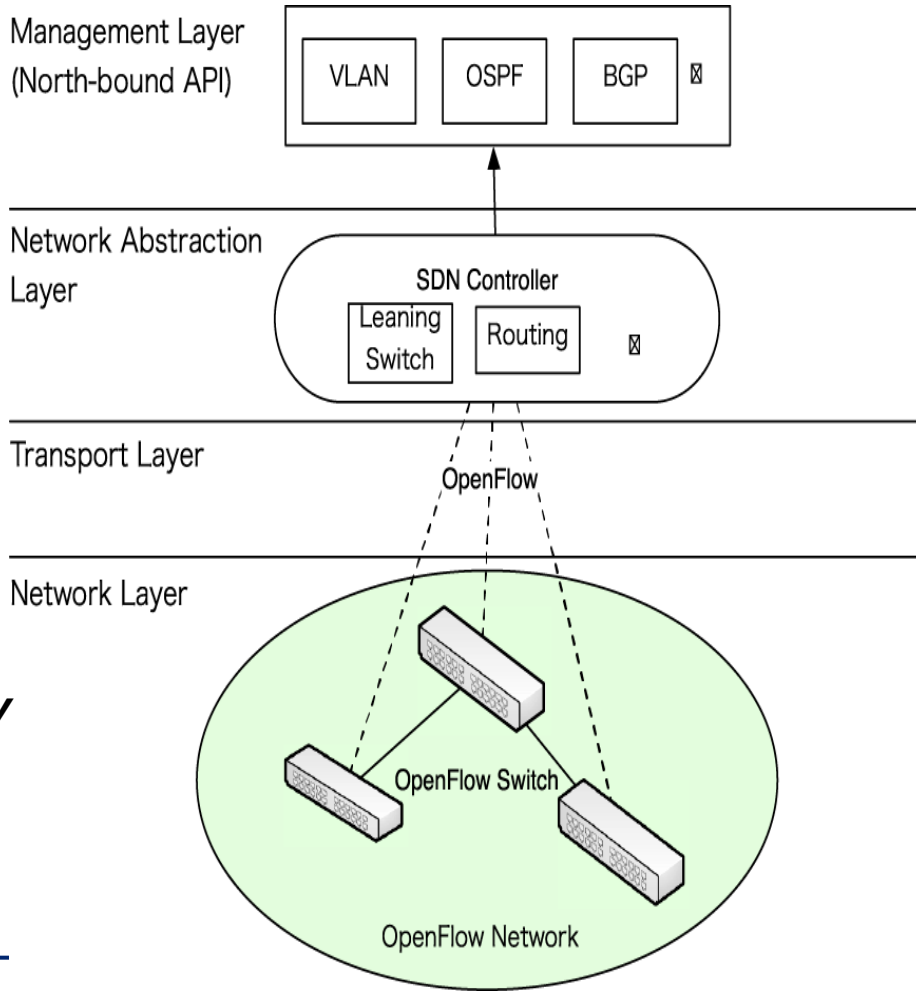


SvDN:
Intension of an administrator is specified in service description, and configurations are automatically derived.



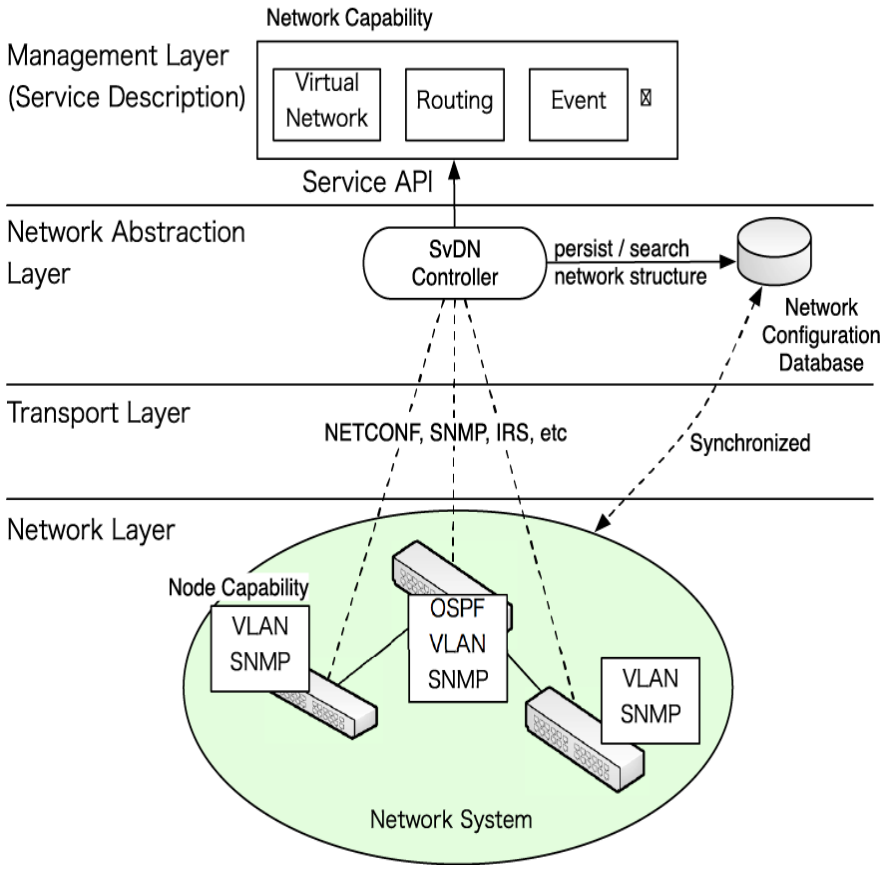
Comparison with Software-defined networking (SDN)

- Network is defined by software
  - In OpenFlow, features of conventional network devices are implemented as a software module on a server, called a controller.
  - Specification of the module is standardized and opened as an API (North-bound API)
- Issues
  - Controller is the bottleneck
    - Set-up of a flow on switches is triggered by the first packet which matches predefined rules ***reactively***



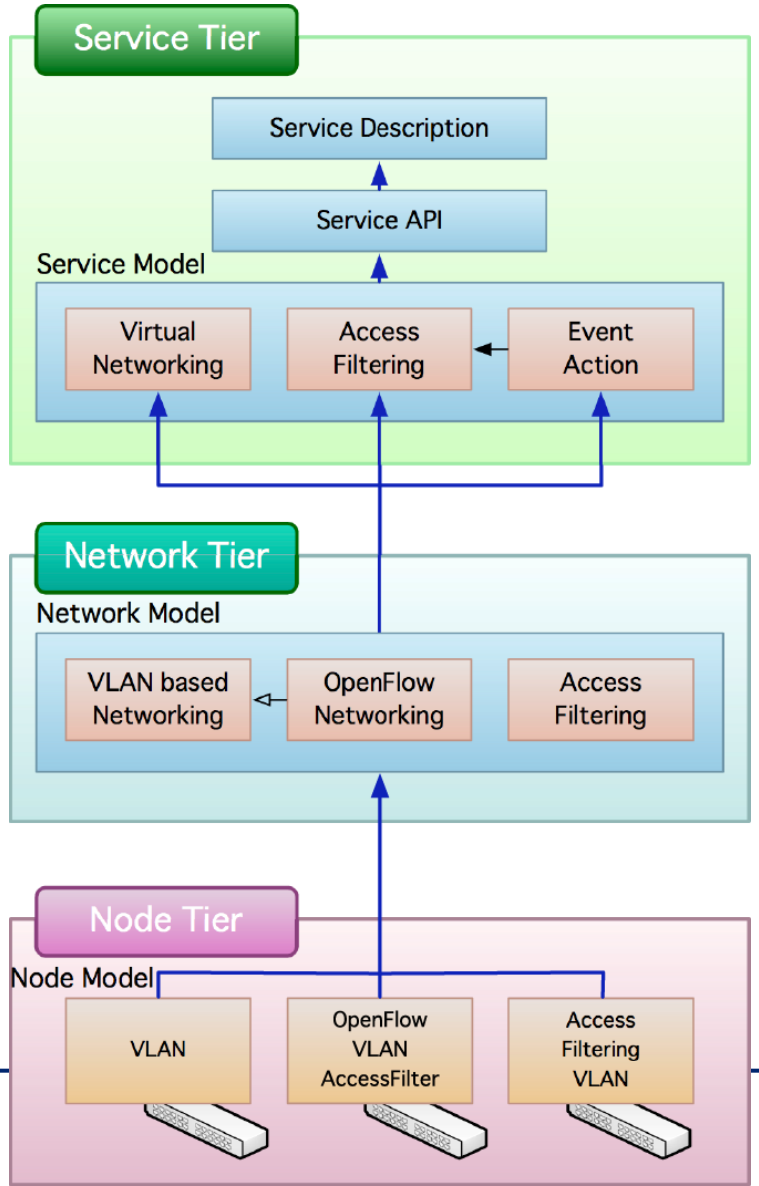
Proposed SvDN Architecture

- SvDN architecture for a LAN system composed of managed network switches
  - Highly abstract service definition provides compatibility and interoperability to other architectures
- Configuration Database which stores network configuration and topology
  - While configuration of devices cannot extract from the physical network topology, we serve a database that stores such information in advance.



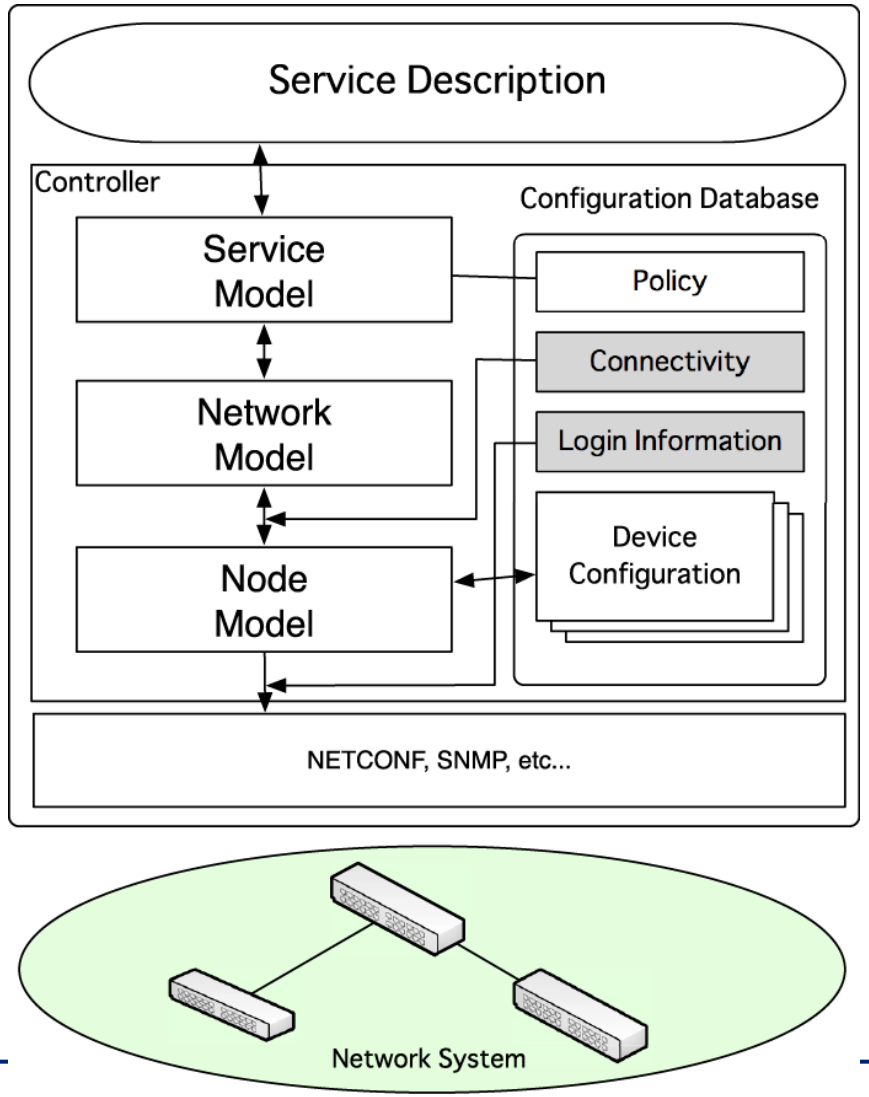
Translation from physical topology to service

- Three-layer modeling
  - Node tier
    - conceals the difference among features and configuration style of devices.
  - Network tier
    - abstracts a network as a whole, e.g. the physical topology of it.
  - Service tier
    - extracts topology-independent services from a network model.



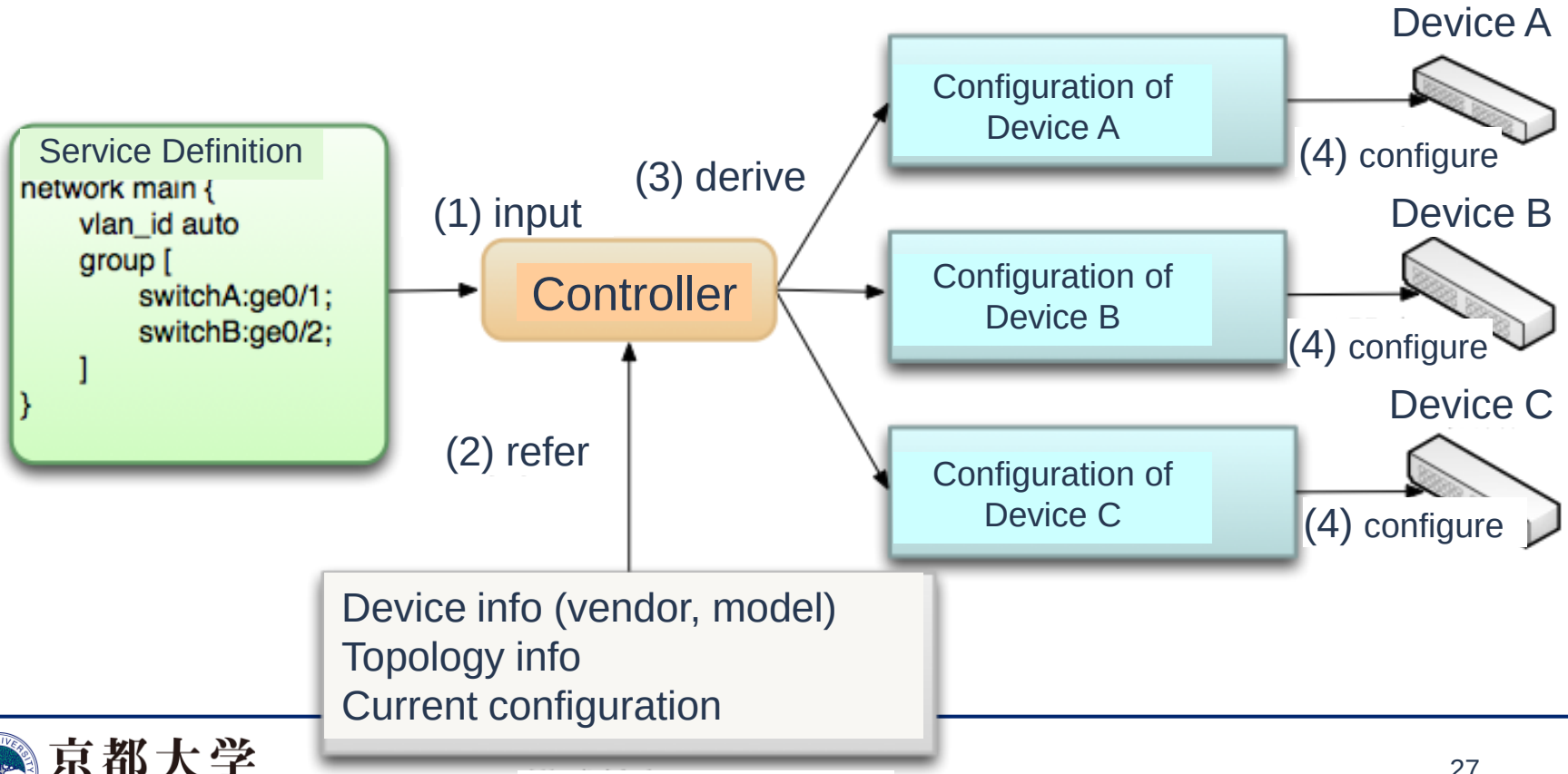
Translation of Models Based on Configuration DB

- Login information like user/passwd, IP address, etc., are stored in DB.
  - Allow to access administration interfaces
- Physical connectivity among devices are stored in DB.
  - Physical topology is strictly manages, so that it works in environments where connection information cannot be derived.



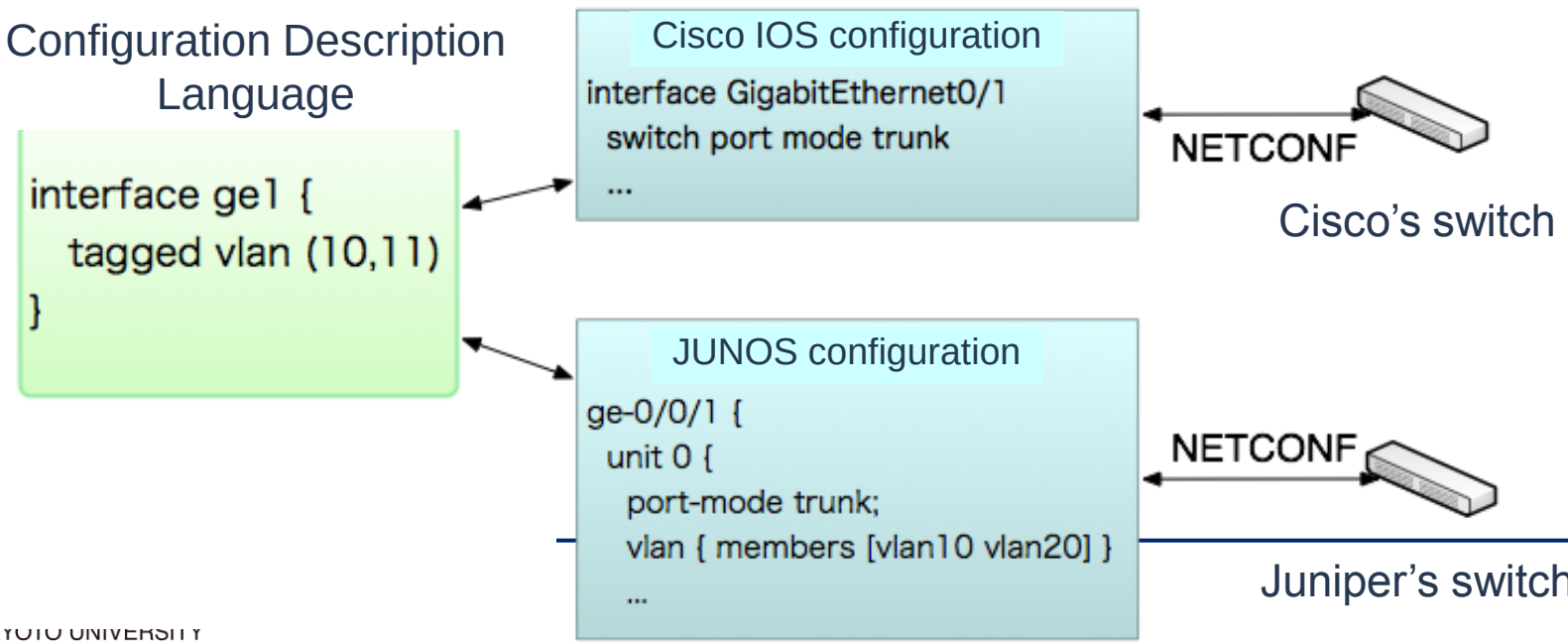
Compiling a service definition into configurations

- Referring configuration DB, a service definition is compiled into configurations of devices each.
- Configuration of each device is set up

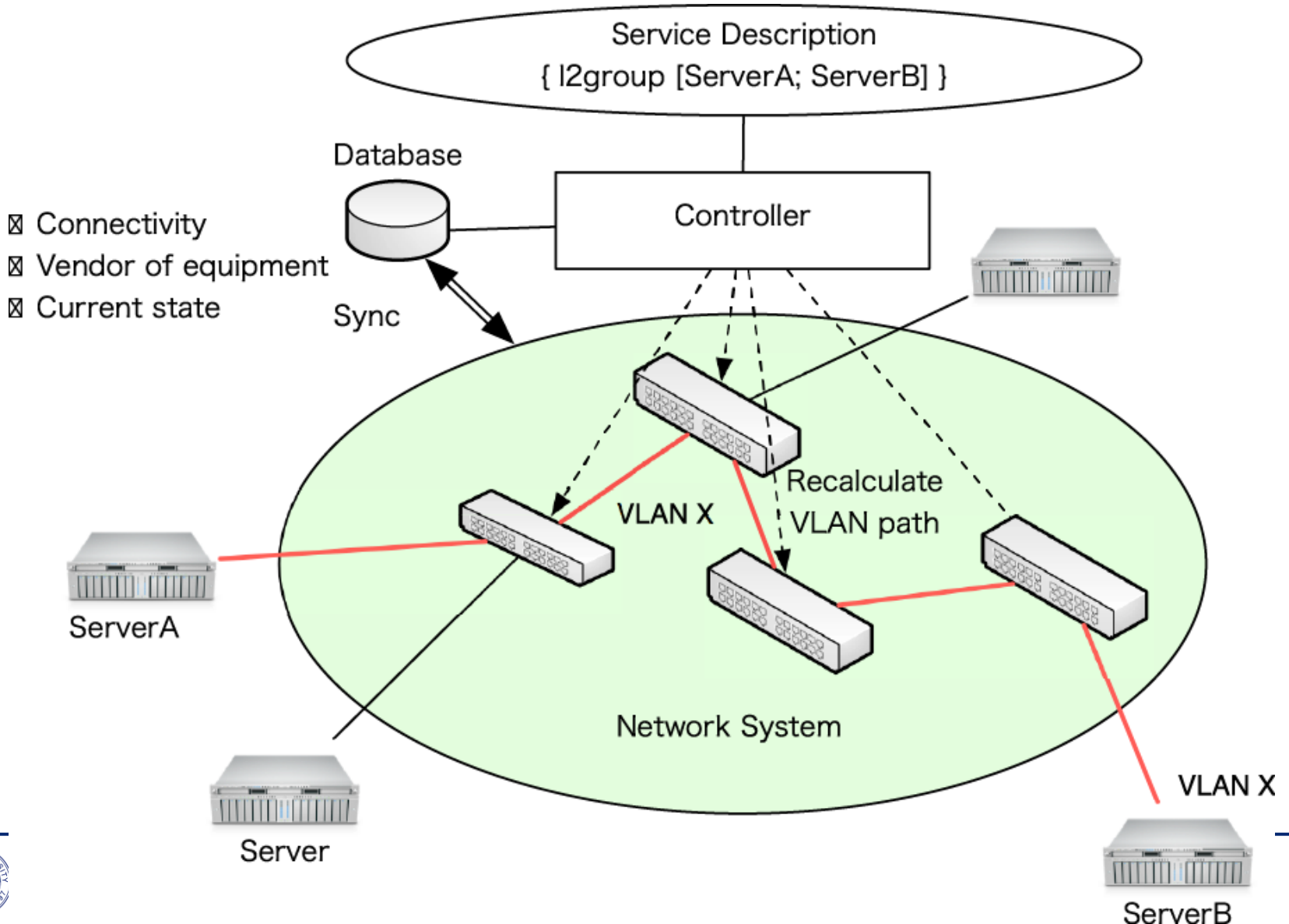


Design of device-independent configuration description language

- Designed a device-independent configuration description language (Scala DSL), which conceals differences among devices
- Configuring devices via the NETCONF over SSH protocol
  - Big difference of implementation status and syntax for configuration



Automatic configuration by service definition



Concluding Remarks

- There have long been discussed on
  - Centralized vs Distributed
  - Reactive vs Proactive
  - Vertical integration vs horizontal division
- Software-Defined Networking is not necessarily a new concept, but the background is changing, especially in global deployment of data-center networking for cloud service.
- Developing new ideas by learning from the past

温故知新

Thank you for your attention!