Introduction to Software Defined Networking (SDN)

Raj Jain
Washington University in Saint Louis
Saint Louis, MO 63130
Jain@cse.wustl.edu

These slides and audio/video recordings of this class lecture are at:
http://www.cse.wustl.edu/~jain/cse570-18/
Overview

1. What is SDN?
2. SDN Controllers
3. Alternative APIs: XMPP, PCE, ForCES, ALTO
4. RESTful APIs and OSGi Framework

Note: This is the second module of three modules on OpenFlow, SDN and NFV in this course.
Origins of SDN

- SDN originated from OpenFlow
- Centralized Controller
  - Easy to program
  - Change routing policies on the fly
  - Software Defined Network (SDN)

- Initially, SDN=
  - Separation of Control and Data Plane
  - Centralization of Control
  - OpenFlow to talk to the data plane

- Now the definition has changed significantly.
Three Features that Define SDN

1. **Abstract the Hardware**: No dependence on physical infrastructure. Software API.
2. **Programmable**: Shift away from static manual operation to fully configurable and dynamic.
3. **Centralized Control of Policies**: Policy delegation and management.
What = Why We need SDN?

1. **Virtualization**: Use network resource without worrying about where it is physically located, how much it is, how it is organized, etc. Abstraction ⇒ Virtualization.

2. **Orchestration**: Should be able to control and manage thousands of devices with one command.

3. **Programmable**: Should be able to change behavior on the fly.

4. **Dynamic Scaling**: Should be able to change size, quantity
   Virtualization ⇒ Scaling

5. **Automation**: To lower OpEx minimize manual involvement
   - Troubleshooting
   - Reduce downtime
   - Policy enforcement
   - Provisioning/Re-provisioning/Segmentation of resources
   - Add new workloads, sites, devices, and resources
Why We need SDN? (Cont)

6. **Visibility**: Monitor resources, connectivity

7. **Performance**: Optimize network device utilization
   - Traffic engineering/Bandwidth management
   - Capacity optimization
   - Load balancing
   - High utilization
   - Fast failure handling

8. **Multi-tenancy**: Tenants need complete control over their addresses, topology, and routing, security

9. **Service Integration**: Load balancers, firewalls, Intrusion Detection Systems (IDS), provisioned on demand and placed appropriately on the traffic path
Why We need SDN? (Cont)

10. **Openness**: Full choice of “How” mechanisms
    ⇒ Modular plug-ins
    ⇒ Abstraction:
      - Abstract = Summary = Essence = General Idea
        ⇒ Hide the details.
      - Also, abstract is opposite of concrete
        ⇒ Define tasks by APIs and **not by how** it should be done.
          E.g., send from A to B. Not OSPF.

Ref: Open Data Center Alliance Usage Model: Software Defined Networking Rev 1.0,”
Software Defined Anything (SDx)

- Tsunami of software defined things
  - Software Defined Networking (SDN)
  - Software Defined Datacenter (SDDC)
  - Software Defined Storage (SDS)
  - Software Defined Compute (SDC)
  - Software Defined Infrastructure (SDI)
Centralized vs. Distributed

- Fast Response to changes
- Fast Consistency
- Less overhead $\Rightarrow$ Scalable
- Single Point of Failure

- Time to converge
- Slow consistency
- Not scalable
- Fault Tolerant

©2018 Raj Jain
What SDN is Not?

- All of these are mechanisms.
- SDN is *not* about a mechanism.
- It is a framework ⇒ Many solutions
Four Confusions About SDN

1. **Policies vs. Control:**
   Control = All bits and messages not sent by the user
   In IP, control includes all header bits and all routing messages.

2. **Separation of Control Plane:**
   Elements have only data plane and have no brains

3. **SDN vs. OpenFlow:**
   OpenFlow is the father of SDN but not SDN.

4. **Need OpenFlow:**
   - OpenFlow is micro-management.
   - It is not scalable.
   - For large infrastructure, need scalable solutions.
Separation vs. Centralization

Separation of Control Plane

Centralization of Policies

Micromanagement is not scalable
Current SDN Debate: What vs. How?

1. SDN is easy if control is centralized but not necessary. Distributed/hierarchical solutions may be required for fail-safe operation.

2. Complete removal of control plane may be harmful. Exact division of control plane between centralized controller and distributed forwarders is yet to be worked out.
Current SDN Debate: What vs. How? (Cont)

3. SDN is easy with a standard southbound protocol like OpenFlow but one protocol may not work/scale in all cases
   1. Diversity of protocols is a fact of life.
   2. There are no standard operating systems, processors, routers, or Ethernet switches.
4. If industry finds an easier way to solve the same problems by another method, that method may win. E.g., ATM vs. MPLS.
Flavors of SDN

1. **OpenDaylight**: Multi-Protocol Southbound
2. **Bare Metal Switches + Network Operating System**
   a. Switches from Dell, Edgecore, HP, Penguin, QCT, Agema, Supermicro
   b. Open Network Install Environment (ONIE) on a set of programmable switch
   c. Network operating system: Alcatel-Lucent, Arista, Big Switch, Broadcom, Brocade, Cisco, Cumulus, Dell, Ericsson, Extreme, HP, Juniper, OCP, Pica8, Pluribus
3. **Network Virtualization/Overlay**: VMWare’s NSX
4. **ONF SDN**: OpenFlow southbound

All provide: Abstraction, Programmability, and Centralization


Source: Alan J Weissberger
Washington University in St. Louis

http://www.cse.wustl.edu/~jain/cse570-18/
Floodlight

- **Java** based OpenFlow controller based on Beacon runs within a JVM. Developers from Big Switch Networks
- **Indigo**: Software to make switch hardware OpenFlow compatible
- **Floodlight** is the core of Big Switch Controller from Big Switch Networks

Washington University in St. Louis

http://www.cse.wustl.edu/~jain/cse570-18/
Floodlight (Cont)

- A number of real-world networking applications
  - **Neutron plug-in** for OpenStack cloud management system
  - **Static Flow Pusher**: Allows users to manually insert flows
  - **Circuit Pusher**: Creates permanent entries on all switches along the path
  - **Firewall**: Enforces access control list (ACL) rules on packets
  - **Big Virtual Switch**: Automates network provisioning for a large scale data centers. Includes provisioning, multi-tenant partitioning

Ref: http://www.projectfloodlight.org/floodlight/
- ONOS

- Open Network Operating System: Distributed OpenFlow OS for a large WAN
- Initially OpenFlow-only. Now multi-protocol southbound.

Ref: ONOS Architecture, https://docs.google.com/presentation/d/1Y4S82YZyskqnKAzW4kKm-6lIO04h_nnrRQyfrsCdt-I/edit?usp=sharing
Washington University in St. Louis
http://www.cse.wustl.edu/~jain/cse570-18/
©2018 Raj Jain
OpenDaylight: Multi-Protocol SDN

Northbound APIs
- DLUX GUI
- REST/RESTCONF/NETCONF/AMQP APIs

OSGi Framework

AAA Authentication Filter

Enhanced Network Services
- AAA
- Centinel – Streaming Data Hdlr
- Controller Shield
- Dev Discovery, Id & Dvr Mgmt
- DOCSIS Abstraction
- LACP
- LISP Svc
- Messaging 4Transport
- NetIDE
- Neutron Northbound
- OVSDB Neutron
- SDN Integration
- Aggregator
- Svc Fn Chaining
- Virtual Net Mgr

Network Abstractions
- ALTO Protocol Mgr
- Fabric as a Service
- Group Based Policy Svcs
- NEMO
- Network Intent Composition

Service Abstraction Layer (SAL)/Core
- Plugin Mgr, Capability Abstractions, Flow Programming, Inventory, etc.

Southbound Protocol Plugins
- USC
- LACP
- CAPWEB
- CoAP
- HTTP
- CAPWAP
- IoT HTTP/CoAP
- PCMM/COPS
- OpenFlow
- OVSDB
- NETCONF
- PCEP
- LISP
- OPFLEX
- SNBI
- SNMP
- SXP

Network Elements
- Network Element
- Network Element
- Network Element
- Overlay Tunnels (VxLAN, NVGRE, …)

Washington University in St. Louis

http://www.cse.wustl.edu/~jain/cse570-18/

©2018 Raj Jain
OpenDaylight SDN Controller Platform (OSCP)

- Multi-company collaboration under Linux foundation
- Many projects including OpenDaylight Controller
- Supports multiple southbound protocols via plug-ins including OpenFlow
- Dynamically linked in to a Service Abstraction Layer (SAL) Abstraction ⇒ SAL figures out how to fulfill the service requested by higher layers irrespective of the southbound protocol
- Modular design using OSGI framework
- A rich set of North-bound APIs via RESTful services for loosely coupled applications and OSGI services for co-located applications using the same address space

https://wiki.opendaylight.org/view/Main_Page
Washington University in St. Louis

http://www.cse.wustl.edu/~jain/cse570-18/
©2018 Raj Jain
Examples Alternative APIs

- Southbound APIs: PCEP, BGP, …
- Northbound APIs: ALTO, …
- Overlay: VxLAN, TRILL, LISP, …
- Configuration API: NETCONF, RESTCONF, …
- Controller: PCE, …

Washington University in St. Louis http://www.cse.wustl.edu/~jain/cse570-18/
©2018 Raj Jain
Path Computation Element (PCE)

- MPLS and GMPLS require originating routers to find paths that satisfy multiple constraints including not using any backup routers and having a given bandwidth etc.
- This may require more computer power or network knowledge than a router may have.
- IETF PCE working group has developed a set of protocols that allow a Path computation client (PCC), i.e., router to get the path from path computation element (PCE)
- PCE may be centralized or may be distributed in many or every router.

What is the 1 Gbps route to New York not going through Boston?

Path Computation Client (PCC)  Path Computation Element (PCE)  Traffic Engineering Database
PCE (Cont)

- PCE separates the route computation function from the forwarding function.
- Both functions may be resident in the same box or different boxes.
- 25+ RFCs documenting protocols for:
  - PCE-to-PCC communication
  - PCE-to-PCE communication (Multiple PCEs)
  - PCE discovery

Ref: http://datatracker.ietf.org/wg/pce/
Ref: http://en.wikipedia.org/wiki/Path_computation_element
Washington University in St. Louis  http://www.cse.wustl.edu/~jain/cse570-18/
Application Layer Traffic Optimization (ALTO)

- IETF working group to optimize P2P traffic ⇒ Better to get files from nearby peers
- Provide guidance in peer selection
- ALTO Server: Has knowledge of distributed resources
- ALTO Client: Requests information from servers about the appropriate peers
- Ratio Criteria: Topological distance, traffic charges, …
- ALTO Server could get information from providers or from nodes about their characteristics, e.g., flat-rate or volume based charging
- A client may get the list of potential peers and send it to the server, which can return a ordered list
- Also need a protocol for ALTO server discovery

Ref: J. Seedorf and E. Berger, “ALTO Problem Statement,” [http://datatracker.ietf.org/doc/rfc5693/?include_text=1](http://datatracker.ietf.org/doc/rfc5693/?include_text=1)
ALTO Extension

- Now being extended to locate resources in data centers
- Need to be able to express
  - resource (memory, storage, CPU, network) availability
  - Cost of these resources
  - Constraints on resources, e.g., bandwidth
  - Constraints on structure, e.g., Power consumption
- ALTO client gets the info from various providers
- Issue of privacy of resource and cost info for the provider

```
Application Orchestrator

ALTO Client

Data Center 1  Data Center 1  Data Center 1
```

Washington University in St. Louis

http://www.cse.wustl.edu/~jain/cse570-18/

©2018 Raj Jain

16-25
XMPP

- Extensible Messaging and Presence Protocol
- **Extensible** $\Rightarrow$ Using XML
- Similar to SMTP email protocol but for near real-time communication
- Each client has an ID, e.g., john@wustl.edu/mobile (John’s mobile phone)
- Client sets up a connection with the server $\Rightarrow$ Client is online
- **Presence**: Server maintains contact addresses and may let other contacts know that this client is now on-line
- **Messaging**: When a client sends a “chat” message to another clients, it is forwarded to these other clients
- Messages are “pushed” ($\Rightarrow$ real-time) as opposed to “polled” as in SMTP/POP emails.

Washington University in St. Louis
http://www.cse.wustl.edu/~jain/cse570-18/
©2018 Raj Jain
XMPP (Cont)

- XMPP is IETF standardization of Jabber protocol
- RFC 6121 defines XMPP using TCP connections. But HTTP is often used as transport to navigate firewalls
- All messages are XML encoded
  \[\Rightarrow\] Not efficient for binary file transfers
  \[\Rightarrow\] Out-of-band binary channels are often used with XMPP.
- A number of open-source implementations are available
- Variations of it are widely used in most instant messaging programs including Google, Skype, Facebook, …, many games
- Used in IoT and data centers for management. Network devices have XMPP clients that respond to XMPP messages containing CLI management requests \[\Rightarrow\] You can manage your network using any other XMPP client, e.g., your mobile phone
- Arista switches can be managed by XMPP, Juniper uses XMPP as a southbound protocol for SDN

Ref: http://en.wikipedia.org/wiki/XMPP
Washington University in St. Louis http://www.cse.wustl.edu/~jain/cse570-18/ ©2018 Raj Jain
XMPP in Data Centers

- Everything is an XMPP entity. It has its own contact list and authorizations.

Ref: https://github.com/ArchipelProject/Archipel/wiki/Architecture-%26-Concepts

©2018 Raj Jain
OpenDaylight Tools

1. **Applications**: Provides Virtual Network Segments (VNS) for each tenant
   1. OpenDaylight Network Virtualization (ONV):
   2. OpenDaylight Virtual Tenant Network (VTN)

2. **Services**:
   1. Unified Secure Channel Manager

3. **Northbound APIs**:
   1. **REST**: Representational State Transfer (like HTTP)
   2. **RESTCONF**: RESTful Configuration
   3. **NETCONF**: Network Configuration
   4. **Dlux**: Northbound API using AngularJS, an extension of HTML by Google for dynamic views
   5. **AMQP**: Advanced Message Queuing Protocol
OpenDaylight Tools (Cont)

4. **Southbound APIs:**
   1. OpenFlow Plug-in + Protocol Library (V1.0, V1.1,…)
   2. Locator ID Separation Protocol (LISP) Mapping Service
   3. SNMP4SDN
   4. BGP Link State Path Control Element Protocol

5. **Overlay:**
   1. Open Distributed Overlay Virtual Ethernet (DOVE):
      Like VxLAN but does not use IP Multicast

6. **Configuration:**
   1. OpenDaylight YANG Tools: NETCONF
   2. Open vSwitch Database (OVSDB) Integration
Open Network Linux

- Linux distribution for “open hardware” bare metal switches
- Part of Open Compute Project
- Supports multiple switch fabric APIs:
  - OF-DPA: OpenFlow Data Plane Abstraction (API) for Broadcom chips
  - OpenNSL: Open Network Switch Layer for Broadcom switches
  - SAI: Switch Abstraction Interface (vendor independent API to control forwarding elements)
- Compatible with many open-source forwarding agents or routing protocol suites

Bare Metal Switches

- Hardware that can be used to load different network operating systems
- Open Network Linux is supported by hardware from: Accton/Edge-Core, Quanta, Dell, Mellanox, Netberg, Inventec, Celestica, HPE, DNI, Ingrasys, and Alpha Networks

Ref: ONL Hardware Support and Certification, http://www.opennetlinux.org/hcl
Washington University in St. Louis
http://www.cse.wustl.edu/~jain/cse570-18/
©2018 Raj Jain
Open Source Forwarding Agents

- **Quagga**: A popular open source routing software suite including OSPF, RIP, BGP, …
- **FRRouting**: a fork of Qagga. Linux routing protocol suite including BGP, IS-IS, LDP, OSPF, PIM, and RIP (Free Range Routing?)
- **BIRD**: Internet Routing Daemon developed as a school project at Charles University, Prague. Supports IPv4, IPv6, BGP, RIP, OSPF, …
- Facebook Open Switching System (**FBOSS**): S/w stack for controlling and managing network switches with several user-space applications
- **Azure Software for Open Networking in the Cloud (**SONiC**)
- **Google gNOS**

Open Network Install Environment (ONIE)

- Part of **Open Compute Project (OCP)** open source initiative
- Allows many different "**Network Operating Systems (NOS)**" on bare metal network switches
- Like a firmware that locates the NOS boot image and loads it
- ONIE sets the environment on the first boot and is not required subsequently

**First Boot**

- Network Operating System (from NOS vendor)
- NOS Installer (from NOS vendor)
- ONIE (from h/w vendor)
- Boot Loader (from h/w vendor)
- Bare Metal Switch Hardware

**Subsequent Boots**

- Boot Loader
- ONIE
- Installer
- NOS
- Hardware

**Configurations**

- Configures Switching ASIC
- Runs network protocols
- Provides CLI
- Linux Executable on USB or Network
- Installs NOS in mass storage
- Thin Linux OS
- Configures Management Ethernet Interface
- Locates and executes NOS Installer
- Loads ONIE from flash

Ref: ONIE Overview, [https://opencomputeproject.github.io/onie/overview/index.html#onie-overview](https://opencomputeproject.github.io/onie/overview/index.html#onie-overview)
Washington University in St. Louis [http://www.cse.wustl.edu/~jain/cse570-18/](http://www.cse.wustl.edu/~jain/cse570-18/)

©2018 Raj Jain
Mininet

- Widely used open source network emulation environment.
- Can simulate a number of end-hosts, switches, routers, links on a Linux
- Used for rapid prototyping of software define networks
- Built-in Open vSwitch, and a OpenFlow capable switch
- Command line launcher and Python API for creating networks of varying sizes, e.g., `mn -topo tree,depth=2,fanout=3`
- Useful diagnostic commands like iperf, ping, and other commands in a host, e.g., `mininet> h11 ifconfig -a`
- Mininet code for several popular commercial switches are available.

RESTful APIs

- Software architecture style developed by W3C.
- Introduced by Roy Fielding in his PhD thesis.
- WWW uses this style. Very popular in other applications.
- Goals: Scalability, Generality, Independence, and allow intermediate components
- Client-Server Model: Clients and servers can be developed undependably.
- Server is stateless
- Responses can be cached for the specified time
- Intermediate Servers (Proxies) can respond. End point is not critical.
REST (Cont)

- Create, Read, Update, Delete (CRUD) Operations
- Uniform Interface: GET (Read), POST (Insert), PUT (write), DELETE
- Resources identified by global identifiers, e.g., URI in Web.
  E.g., GET http://odcp.org/rest/v1/model/controller-node
- Data Types: Controller node, Firewall rule, Topology configuration, Switch, Port, link, flow entry, VLAN, …
- Data types can include commercial entities, such as, Big Virtual Switch from Big Switch Networks, vCenter from VMware, …
- If optional-id and query parameters are omitted, the returned text includes all of the items of the given data type.

OSGi Framework

- Initially, Open Services Gateway initiative
- A set of specifications for dynamic application composition using reusable Java components called bundles
- Bundles publish their services with OSGi services registry and can find/use services of other bundles

Ref: http://www.osgi.org/Technology/WhatIsOSGi

Washington University in St. Louis http://www.cse.wustl.edu/~jain/cse570-18/
OSGi (Cont)

- Bundles can be installed, started, stopped, updated or uninstalled using a lifecycle API
- Modules defines how a bundle can import/export code
- Security layer handles security
- Execution environment defines what methods and classes are available in a specific platform
- A bundle can get a service or it can listen for a service to appear or disappear.
- Each service has properties that allow others to select among multiple bundles offering the same service
- Services are dynamic. A bundle can decide to withdraw its service. Other bundles should stop using it
  \[ \Rightarrow \text{Bundles can be installed and uninstalled on the fly.} \]
Summary

1. SDN = Abstraction + Programmability + Centralization
   SDN = Disaggregation of h/w and s/w
   = Bare metal switches + ONIE + ONL

2. OpenFlow originated SDN but now many different southbound and northbound APIs, intermediate services and tools are being discussed and implemented by the industry, e.g., XMPP, PCE, ALTO

3. OpenDaylight and ONOS are SDN Controllers.
   Differ on how much open.

4. Mininet for network simulation

5. REST=HTTP APIs
   OSGI framework for modularity
Reading List


Wikipedia Links

- [http://en.wikipedia.org/wiki/Path_computation_element](http://en.wikipedia.org/wiki/Path_computation_element)
References

- J. Seedorf and E. Berger, “ALTO Problem Statement,” http://datatracker.ietf.org/doc/rfc5693/?include_text=1
- http://www.osgi.org/Technology/WhatIsOSGi
- http://www.sdncentral.com/sdn-use-cases/
- http://datatracker.ietf.org/wg/pce/
- https://wiki.opendaylight.org/view/Main_Page
References (Cont)

References (Cont)

Acronyms

- ACI  Application Policy Infrastructure
- ACL  Access Control List
- AEX  Application Information Exposure
- ALG  Application Level Gateway
- ALTO Application Layer Traffic Optimization
- AMQP Advanced Message Queueing Protocol
- ANDSF Access Network Discovery and Selection Function
- API  Application Programming Interface
- APIC Application Policy Infrastructure Controller
- ARP  Address Resolution Protocol
- ATIS Association for Telecom Industry Solutions
- ATM  Asynchronous Transfer Mode
- AVNP  Active Virtual Network Management Protocol
- BGP  Border Gateway Protocol
- BNC  Big Switch Network Controller
- BSD  Berkeley Software Distribution
Acronyms (Cont)

- BUM: Broadcast, Unknown, and Multicast
- CDN: Content Distribution Network
- CDNI: Content Distribution Network Interconnection
- CE: Control Element
- CLI: Command Line Interface
- CMS: Content Management System
- CPU: Central Processing Unit
- CRUD: Create, Read, Update, Delete
- CSP: Cloud Service Provider
- DHCP: Dynamic Host Control Protocol
- DNS: Domain Name System
- DOCSIS: Data over Cable Service Interface Specification
- DOVE: Distributed Overlay Virtual Ethernet
- DVS: Distributed Virtual Switch
- EID: Endpoint Identifier
- ETSI: European Telecommunications Standards Institute
Acronyms (Cont)

- FCAPS: Faults, configuration, accounting, performance, and security
- FE: Forwarding Element
- FE: Forwarding Element
- ForCES: Forwarding and Control Element Separation
- GMPLS: Generalized Multi-Protocol Label Switching
- GUI: Graphical User Interface
- HTML: Hypertext Markup Language
- HTTP: Hypertext Transfer Protocol
- I2AEX: Infrastructure to Application Information Exposure
- IaaS: Infrastructure as a Service
- ID: Identifier
- IDS: Intrusion Detection System
- IEEE: Institution of Electrical and Electronic Engineers
- IETF: Internet Engineering Task Force
- IGP: Interior Gateway Protocol
- IoT: Internet of Things
Acronyms (Cont)

- IP: Internet Protocol
- IPv4: Internet Protocol version 4
- IPv6: Internet Protocol version 6
- IRTF: Internet Research Taskforce
- IS-IS: Intermediate System to Intermediate System
- ISO: International Standards Organization
- L2: Layer 2
- LACP: Link Aggregation Control Protocol
- LAN: Local Area Network
- LISP: Locator-ID Separation Protocol
- LS: Link State
- MAC: Media Access Control
- MPLS: Multi-protocol Label Switching
- NAT: Network Address Translation
- NetIDE: Network Interactive Development Environment
- NEMO: File Manager for Linux Distribution
### Acronyms (Cont)

- **NFV**  
  Network Function Virtualization
- **NTP**  
  Network Time Protocol
- **NVGRE**  
  Network Virtualization using Generic Routing Encapsulation
- **NVO3**  
  Network Virtualization over L3
- **NVP**  
  Network Virtualization Platform
- **OF**  
  OpenFlow
- **OnePK**  
  Open Network Environment Platform Kit
- **ONF**  
  Open Networking Forum
- **ONV**  
  OpenDaylight Network Virtualization
- **OpEx**  
  Operational Expenses
- **OS**  
  Operating System
- **OSCP**  
  OpenDaylight SDN Controller Platform
- **OSGi**  
  Open Services Gateway Initiative
- **OSPF**  
  Open Shortest Path First
- **OVS**  
  Open Virtual Switch
- **OVSDB**  
  Open Virtual Switch Database
Acronyms (Cont)

- **PCC**  Path Computation Client
- **PCE**  Path Computation Element
- **PCEP** Path Computation Element Protocol
- **POP**  Post Office Protocol
- **PWE3** Pseudowire Emulation Edge to Edge
- **QoS**  Quality of Service
- **REST** Representational State Transfer
- **RFC**  Request for Comments
- **RLOC** Routing Locator
- **RLOC** Routing Locator
- **RS**  Routing System
- **SAL**  Service Abstraction Layer
- **SDN**  Software Defined Networking
- **SMTP** Simple Mail Transfer Protocol
- **SNMP** Simple Network Management Protocol
Acronyms (Cont)

- SNMP4SDN: SNMP for SDN
- SSH: Secure Socket Host
- STT: Stateless TCP-like Transport
- TCP: Transmission Control Protocol
- TE: Traffic Engineering
- TIA: Telecom Industry Association
- TRILL: Transparent Interconnection of Lots of Links
- URI: Uniform Resource Identifier
- vBridge: Virtual Bridge
- VIRL: Virtual Internet Routing Lab
- VLAN: Virtual Local Area Network
- VM: Virtual Machine
- VNS: Virtual Network Segment
- VPN: Virtual Private Network
- vTep: Virtual Tunnel End Point
- VTN: Virtual Tenant Network
Acronyms (Cont)

- VxLAN Virtual Extensible Local Area Network
- WAN Wide Area Network
- XML Extensible Markup Language
- XMPP Extensible Messaging and Presence Protocol
# Style Guide

<table>
<thead>
<tr>
<th>Correct</th>
<th>Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACL</td>
<td>Acl or acl</td>
</tr>
<tr>
<td>API</td>
<td>api</td>
</tr>
<tr>
<td>ARP</td>
<td>Arp or arp</td>
</tr>
<tr>
<td>IPsec</td>
<td>IPSEC or ipsec</td>
</tr>
<tr>
<td>IPv4 or IPv6</td>
<td>Ipv4, Ipv6, ipv4, ipv6, IPV4, or IPV6</td>
</tr>
<tr>
<td>Karaf</td>
<td>karaf</td>
</tr>
<tr>
<td>Linux</td>
<td>LINUX or linux</td>
</tr>
<tr>
<td>NETCONF</td>
<td>Netconf or netconf</td>
</tr>
<tr>
<td>Neutron</td>
<td>neutron</td>
</tr>
<tr>
<td>OSGi</td>
<td>osgi or OSGI</td>
</tr>
<tr>
<td>Open vSwitch</td>
<td>OpenvSwitch, OpenVSwitch, or Open V Switch.</td>
</tr>
<tr>
<td>OpenDaylight</td>
<td>Opendaylight, Open Daylight, or OpenDayLight.</td>
</tr>
<tr>
<td>OpenFlow</td>
<td>Openflow, Open Flow, or openflow.</td>
</tr>
<tr>
<td>OpenStack</td>
<td>Open Stack or Openstack</td>
</tr>
<tr>
<td>QoS</td>
<td>Qos, QOS, or qos</td>
</tr>
<tr>
<td>RESTCONF</td>
<td>Restconf or restconf</td>
</tr>
<tr>
<td>RPC</td>
<td>Rpc or rpc</td>
</tr>
<tr>
<td>URL</td>
<td>Url or url</td>
</tr>
</tbody>
</table>

SDN Related Organizations and Projects

- Open Networking Foundation (ONF): www.opennetworking.org
- Telecom Industry Association (TIA): www.tiaonline.org
- Internet Engineering Task Force (IETF): www.ietf.org
- OpenStack Quantum: https://wiki.openstack.org/wiki/Quantum
- OpenDaylight: www.opendaylight.org
SDN Web Sites

- SDN Central, http://www.sdncentral.com
Related Modules

CSE567M: Computer Systems Analysis (Spring 2013),
https://www.youtube.com/playlist?list=PLjGG94etKypJEKjNAa1n_1X0bWWNyZcof

CSE473S: Introduction to Computer Networks (Fall 2011),
https://www.youtube.com/playlist?list=PLjGG94etKypJWOSPMh8Azcgy5e_10TiDw

Wireless and Mobile Networking (Spring 2016),
https://www.youtube.com/playlist?list=PLjGG94etKypKeb0nzyN9tSs_HCd5c4wXF

CSE571S: Network Security (Fall 2011),
https://www.youtube.com/playlist?list=PLjGG94etKypKvzfVtutHcPFJXumyyg93u

Video Podcasts of Prof. Raj Jain's Lectures,
https://www.youtube.com/channel/UCN4-5wzNP9-ruOzQMs-8NUw