Software Defined Networking

Amante Alvaran
Agenda

• Business Consideration
• A Customer Journey
• Quick Overview of SDN and NFV
• Software Networking Solutions
  – SDN Controller
  – vRouter
  – Virtual Application Delivery Controller (vADC)
• Network Complexity, Why the Need for SDN
• SDN A New Solution
• OpenFlow Basics
• SDN Solution Demo
Executive Management

Profit, Revenue

Time to Market

Business Growth

Handle Spikes in Customer Requests

CAPEX/OPEX

Customer SLAs
Industry Shifts

How do I achieve...

• Rapid Deployment of New Services
• Economies of Scale
• Better ROI
Data Center Journey over the Past Decade!

Business Dynamics Act as a Catalyst for Network Change

Consolidation  Virtualization  Convergence  Cloud  New IP

Customers Journey Towards Success

BEFORE

- Decreased CAPEX and OpEx
- 95% provisioning time eliminated

AFTER

- High performance
- Executive Management
- Rapid setup / Speed
  Delivered new services to 100 markets in under 6 months
- Save costs / Decreased CapEx and OpEx

Timeline

Revenue

Bandwidth use

Capex spend

Cost ($$)
Overview
Software-Defined Networking (SDN)
Network Function Virtualization (NFV)
Components of a Data Center

Application and Services
- OSS/BSS, ERP, Databases, Enterprise Apps, Consulting, Professional Services, Systems Integrations

Management (Infrastructure, Enterprise)
- Infrastructure (Configuration, HA, DR, Workload)
- Enterprise (Monitoring, Analytics, Chargeback)

Physical and Virtual Infrastructure
Software Defined Networking (SDN)
A Programmable Network—Design, Build, Manage

Advantages
- Logically centralized view of the overall network
- Improved interoperability and manageability
- Enable users to develop and deploy unique capabilities for their network needs
Network Functions Virtualization (NFV)

Advantages

- Virtualize the network devices to create multiple virtual functions for various network appliances
- Enable service innovations and meet service SLAs
- Reduce CAPEX/OPEX
Software Networking Solutions
Stepping stones to Next Gen Connectivity

Software Defined Networking (SDN)
SDN Controller

Network Function Virtualization (NFV)
vRouter
vADC

Greater Agility
- Deliver industry leading performance for any given resource assignment
- On-demand licensing and flexible utility-based pricing brings agility to capacity acquisition
- Streamline network processes

Open with a Purpose
- Delivers an open modular architecture that enables customers to program and control their network of choice
- Strong investments in NFV management and orchestration through open source frameworks like OpenStack
- First pure OpenDaylight SDN Controller distribution

Improved Economics
- Economies of scale
- Investment protection for heterogeneous infrastructure
- Lower CAPEX/OPEX
SDN Controller
Open source SDN Controller

Designed for an open, multivendor world
- Each layer can be selected independently
- No platform or northbound dependencies

Simple on-ramp to SDN
- Low-risk investment protection
- Smooth installation and maintenance

Collaborative innovation
- Joint and custom app development
- Bridge to OpenDayLight community

Open-source SDN Controller built directly from OpenDayLight.
vRouter

Increase networking agility and connectivity with improved ROI

vRouter is the first virtual router capable of providing advanced networking and security in software with scale, reliability, and performance.

High ROI and efficiency
- Proven high VNF density of vRouters in single server
- Low CPU consumption provides headroom for other VNFs

Delivers breakthrough performance
- 80Gbps line rate on single vRouter
- Advanced enhancements with Intel DPDK

High versatility
- Single product, multiple functions for Cloud & SP
- Cloud network segmentation, Secure gateway services and Workload and Subscriber firewall

Proven High Density with Breakthrough Performance
Virtual Application Delivery Controller (vADC)

Fast, secure, available applications

Purpose–built for the Cloud
- Deploy in any Hybrid Cloud architectures
- On-demand application delivery—ADCaaS
- Established CSP partnerships—Azure, AWS, Rackspace, Joyent, etc.

Process automation
- Easy, consistent deployment of Service templates (LBaaS) and service isolation with centralized mgmt.
- Usage visibility—metering and billing capabilities for chargeback
- Integration with key orchestration tools—OpenStack, VMware NSX and Brocade SDN Controller

Hyperscale and Performance on Demand
- Highest Layer 7 throughput for software (50GB/node)
- Unique web application security solution
Software Networking Solutions
Agile, Open, Economics

Branch Cloud

Data Center
Network Complexity and Why the need SDN
Network adaption to demands

- Last 20 years > radically shifting network demands
  - Massive increase in scale (end points, switches, bytes, flows etc.)
  - Static endpoints (weeks-months) > dynamic endpoints (hours-days)
  - Mostly north-south traffic > mostly east-west traffic
- By contrast, network haven’t changed much
  - Link speeds have gone up, but...
  - Still largely managed networks devices-by-device via CLI
  - If you’re lucky, orchestration at the granularity of a few device
Things are changing

• Device-to-device > Network wide
• Open Standards > Open Standards + Open Source
• Proprietary Software > Open Source
• Networking, Storage, Compute > Converge IT
• Hardware > Software (Virtualized)

• To large extend, moving towards open source networking
Solution: disaggregation and open software

• Device-by-device operations

• Proprietary, vendor-specific verticals stacks for control, management and orchestration
  – Limited innovation in individual silos

• Network-wide operation

• Open control, management and orchestration using open control protocols/modeling language
  – Independent innovation at each layer of the stack
SDN A New Solution
Who is behind Software Defined networking?

Open Networking Foundation (ONF)

- ONF launched publicly in March 2011
- Support from more than 70 major companies
- The ONF define OpenFlow and API specifications
- Founding member of ONF
  - Deutsche Telecom
  - Microsoft
  - Facebook
  - Verizon Wireless
  - Google
  - Yahoo
So, SDN is all about…

- Network Programmability
  - API interaction with network elements
- Separation of Control Plane and Forwarding Plane
  - Forwarding Plane can be software or hardware
  - Control Plane – agnostic to the underlying hardware
- Integration with higher-order Orchestration platforms
  - OpenStack, CloudStack, vCloud Director
- Network topology and orchestration derived from the application tenant. This is how SDN is different from switched networks.
- Vendor independence – open source
What is Driving SDN?

- SDN is not about Agility
- It is about new services
- It is about the economics of networking
  - Capital Costs an the ‘Tax”
  - Energy costs
  - Real-Estate
  - Operating Costs
SDN Use Cases

• Custom Analytics and Compliance
• Big Data
• Security
• QoS and Traffic Management
• Network Virtualization
• Service Configuration Policy
• Research and New Protocols
• Fault and Disaster Recovery
• WAN Optimization
SDN Controller

Open, Modular

- Designed for open, multivendor world
  - Each layer can be selected independently
  - No platform or northbound dependencies
- Simple on-ramp to SDN
  - Low risk investment protection
  - Smooth installation and maintenance
- Collaborative Innovation
  - Joint and custom app development
  - Bridge to OpenDayLight community
Yang

• Data modelling language for the NETCONF network configuration protocol. Specifically designed to support service activation and provisioning.

• Is a modular language representing data structures in an XML tree format. XML content transported over SSH+TCP.

• XML Namespaces make it possible to add e.g. new RPC types or new table columns without breaking existing applications.

• Configuration changes happen all-or-nothing and all-at-once which simplifies network management applications.

• Can address multiple network elements in parallel to implement network-wide transactions.
Yang Models

Network Topology
List of Nodes
List of Links
Links and Nodes can be “augmented” later

Can have multiple topologies
Overlay/underlay
Disjoint
Peered

```yaml
container network-topology {
  ...

  list node {
    description "...";
    key "node-id";
    uses node-attributes;
  }

  list link {
    description "...";
    key "link-id";
    uses link-attributes;
  }
}
```
YANG Submodules

Each submodule belongs to one specific main module

```yang
module acme-module {
    namespace "...";
    prefix acme;

    import "ietf-yang-types" {
        prefix yang;
    }

    include "acme-system";

    organization "ACME Inc.";
    contact joe@acme.example.com;
    description "Module describing the ACME products";
    revision 2007-06-09 {
        description "Initial revision.";
    }
}

submodule acme-system {
    belongs-to acme-module {
        prefix acme;
    }

    import "ietf-yang-types" {
        prefix yang;
    }

    container system {
        ...
    }
}
```
Postman to test API interaction

• Is a language for developers to communicate and test APIs. This led to the build, test, and document APIs faster.

• Postman features include:
  – History of sent requests
  – Create requests quickly
  – Replay and organize
  – Switch context quickly
  – Built-in authentication helpers
  – Customize with scripts
  – Automate collections
RESTCONF – Northbound API

• RESTCONF describes how to map a YANG specification to a REST interface.

• RESTCONF provides a simplified interface that follows REST-like principles and is compatible with a resource-oriented device abstraction.

• SDN Controller implements a memory data store to store data that is related to the YANG models. Using the RESTCONF protocol, you can perform a CRUD (create, read, update or delete) operation on the data that is stored in the memory data store.
NETCONF

- Network management protocol specifically designed to manipulate configuration, support service activation and provisioning.
  - XML content transported over SSH+TCP.
  - XML Namespaces make it possible to add e.g. new RPC types or new table columns without breaking existing applications.
  - Can address multiple network elements in parallel to implement network-wide transactions.
### Comparison between SNMP and NETCONF

<table>
<thead>
<tr>
<th></th>
<th>SNMP</th>
<th>NETCONF</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data models</strong></td>
<td>MIB</td>
<td>YANG</td>
</tr>
<tr>
<td><strong>Data Modeling Language</strong></td>
<td>SMI</td>
<td>YANG</td>
</tr>
<tr>
<td><strong>Management Operations</strong></td>
<td>SNMP</td>
<td>NETCONF</td>
</tr>
<tr>
<td><strong>Transport Protocol</strong></td>
<td>UDP</td>
<td>SSH</td>
</tr>
</tbody>
</table>
SDN Controller using NETCONF

Feature Supported

- NETCONF Protocol Version 1.1, RFC 6241
  - Read-config(get/get-config)
  - Edit-config
  - Copy-config
  - Delete-config

- Notifications Lock/unlock
  - Lock the candidate configuration, which prevents other users or applications from changing it until the application releases the lock
OpenFlow Basics
OpenFlow Introduction

- Consists of a controller, an OpenFlow-enabled switch, and the OpenFlow communications protocol.
- The switch and the controller communicate by exchanging OpenFlow protocol messages, which the controller uses to add, modify, and delete flows in the switch.
- Control various aspects of the network, such as traffic forwarding, topology discovery, Quality of Service.
- OpenFlow-enabled router supports an OpenFlow Client (control plane software)
- OpenFlow Client communicates with an OpenFlow Controller using the OpenFlow protocol
  - OpenFlow Controller runs on a server
- OpenFlow-enabled routers support the abstraction of a Flow Table, which is manipulated by the OpenFlow Controller
OpenFlow-Enabled Router Operation

- Flow Table contains Flow Entries
  - Each Flow Entry represents a Flow, e.g., packets with a given destination IP address

- The flow table is sorted by flow priority, which is defined by the controller
  - Highest priority flows are at the top of the Flow Table

- Incoming packets are matched against the flow entries (in order)
  - Matching means: Does the packet belong to this Flow?

- If there is match, flow matching stops, and the set of actions for that flow entry are performed
  - Packets that don’t match any flow entry are typically dropped
Flow Table Entry

Open Flow 1.0

Each flow table entry contains a set of rules to match (e.g., IP src) and an action list to be executed in case of a match (e.g., forward to port list).

- Forward packet to a port list
- Add/remove/modify VLAN Tag
- Drop packet
- Send packet to the controller
OpenFlow Matching Operation

Set Input Port
Ether Src
Ether Dst
Ether Type
Set all others to zero

EtherType = 0x8100?
Y
N

EtherType = 0x806?
Y
N

EtherType = 0x0800?
Y
N

Set VLAN ID
Set VLAN Priority
Use EtherType in VLAN tag for next EtherType Check

Set IP Src, IP Dst
IP Proto, IP ToS from within ARP

Not IP Fragment?
Y
N

IP Proto = 6 or 7?
Y
N

IP Proto = 1?
Y
N

Packet lookup using assigned header fields

Match Table 0?
Y
N

Match Table n?
Y
N

Apply Actions
Set Src Port, Dst Port for L4 fields
Use ICMP Type and code for L4 Fields

Send to Controller
## OpenFlow Counter

<table>
<thead>
<tr>
<th>Per Table</th>
<th>Per Flow</th>
<th>Per Port</th>
<th>Per Queue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active Entries</td>
<td>Received Packets</td>
<td>Received Packets</td>
<td>Transmit Packets</td>
</tr>
<tr>
<td>Packet Lookups</td>
<td>Received Bytes</td>
<td>Transmitted Packets</td>
<td>Transmit Bytes</td>
</tr>
<tr>
<td>Packet Matches</td>
<td>Duration (Secs)</td>
<td>Received Bytes</td>
<td>Transmit overrun errors</td>
</tr>
<tr>
<td></td>
<td>Duration (nanosecs)</td>
<td>Transmitted Bytes</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive Drops</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transmit Drops</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive Errors</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transmit Errors</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive Frame</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Alignment Errors</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive Overrun errors</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive CRC Errors</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Collisions</td>
<td></td>
</tr>
</tbody>
</table>
OpenFlow Timer

• Idle timeout
  – Remove entry if no packets received from this time

• Hard timeout
  – Remove entry after this time

• If both are set, the entry is removed if either one expires
OpenFlow Action

• Forward to physical port I or to Virtual Port
  - All: to all interfaces except incoming interface
  - Controller: encapsulate and send to controller
  - Local: Send to its local forwarding table
  - Table: Perform action in the flow table
  - In_port: Send back to input port
  - Normal: Forward using traditional Ethernet
  - Flood: Send along minimum spanning tree except the incoming interface

• En queue: To a particular queue in the port => QoS

• Drop

• Modify field: E.g., add/remove VLAN tags, ToSbits, Change TTL
Open vSwitch Features

Inter-VM communication monitoring via:

- **NetFlow**: protocol for sampling and collecting traffic statistics (RFC 3954)
- **sFlow**: Similar to NetFlow by sflow.org (RFC 3176)
- **Jflow**: Juniper’s version of NetFlow
- **NetStream**: Huawei’s version of NetFlow
- **IPFIX**: IP Flow Information Export Protocol (RFC 7011) IETF standard for NetFlow
- **SPAN, RSPAN**: Remote Switch Port Analyzer–port mirroring by sending a copy of all packets to a monitor port
- **GRE-tunnelled mirrors**: Monitoring device is remotely connected to the switch via a GRE tunnel
OpenFlow Version Development

• Version 1.1
  - Perform action on a match. Ethernet/IP only. multipath (a flow can be sent over one of several path), MPLS, Q-in-Q and tunnel

• Version 1.2
  - IPv6 Support: Matching fields include IPv6 source address, destination address, protocol number, traffic class. ICMPv6 type, ICMPv6 code, IPv6 neighbor discovery header fields, and IPv6 flow labels

• Version 1.3
  - IPv6 extension headers: Can check if Hop-by-hop, Router, Fragmentation, Destination options, Authentication, Encrypted Security Payload (ESP), unknown extension headers are present MPLS Bottom-of-Stack bit matching MAC-in-MAC encapsulation Tunnel ID metadata: Support for tunnels (VxLAN, ...)
  - Meter Band: If the packet/byte rate exceeds a pre-defined Band's threshold, the meter has triggered the band. Action can be set to remark DSCP or drop
Key Difference Between OpenFlow 1.0 and Beyond

OF 1.0 model
(single lookup)

- Single Table
- Controller
- Packet IN
- Packet DROP
- Packet OUT

OF 1.1 and beyond model
(multiple lookups)

1. Find highest-priority matching flow entry
2. Apply instructions:
   i. Modify packet & update match fields (apply actions instruction)
   ii. Update action set (clear actions and/or write actions instructions)
   iii. Update metadata
3. Send match data and action set to next table

© 2015 BROCADE COMMUNICATIONS SYSTEMS, INC.
Required matched Field and Action

<table>
<thead>
<tr>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>OXM_OF_IN_PORT</td>
</tr>
<tr>
<td>OXM_OF_ETH_DST</td>
</tr>
<tr>
<td>OXM_OF_ETH_SRC</td>
</tr>
<tr>
<td>OXM_OF_ETH_TYPE</td>
</tr>
<tr>
<td>OXM_OF_IP_PROTO</td>
</tr>
<tr>
<td>OXM_OF_IPV4_SRC</td>
</tr>
<tr>
<td>OXM_OF_IPV4_DST</td>
</tr>
<tr>
<td>OXM_OF_IPV6_SRC</td>
</tr>
<tr>
<td>OXM_OF_IPV6_DST</td>
</tr>
<tr>
<td>OXM_OF_TCP_SRC</td>
</tr>
<tr>
<td>OXM_OF_TCP_DST</td>
</tr>
<tr>
<td>OXM_OF_UDP_SRC</td>
</tr>
<tr>
<td>OXM_OF_UDP_DST</td>
</tr>
</tbody>
</table>

- Output
- *Set-queue
- Drop
- Group
- *Push-Tag/Pop-Tag
- *Set-Field (example: VLAN)
- *Change-TTL
- * Optional fields
Metering

• Per-flow metering measures and controls the rate of packets for each flow entry.

• Per-flow meters enable OpenFlow to implement simple QoS operations, such as rate-liming, and can be combined with per-port queues to implement complex QoS frameworks, such as DiffServ.

• Meters are attached directly to flow entries. Each meter can have one or more meter bands. Each meter band specifies the rate of the band applies and the way packets are processed (DROP or DIFFSERV). OpenFlow metering operation is similar to ingress rate limiting in a QoS operation.
OpenFlow Applications
What can we do with OpenFlow

• OpenFlow itself does not define or mandate any specific application
• OpenFlow enables a large set of applications due to its flexibility
• Supported applications should increase over time as new functionality is added to the OpenFlow specification
  – E.g., flow policing/rate limiting
• Ideal for automating common operations
  – E.g. security via ACLs, isolation via VLANs or VRFs etc.
OpenFlow Applications (cont..)

What can we do with OpenFlow

• Hyper-scale Data Centers
  – Address the Virtual Machine (VM) scale challenge with OpenFlow and facilitating VM mobility
  – Facilitating multi-tenancy

• Network Virtualization
  – Simplifying multi-tenancy (also part of any hyper-scale DC solution)

• Flow Management in metro/WAN
  – Traffic engineering, load balancing, security management, and etc

• Enterprise/campus data center applications
  – Reduce operational complexity and facilitating deployment of new features
Hybrid Switch versus Hybrid Port Modes

• Hybrid Switch Mode
  - OpenFlow is enabled on a per-port basis
  - OpenFlow ports do not run existing features
  - Non-OpenFlow-enabled ports run existing features, e.g., MPLS/IP routing

• Hybrid Port Mode
  - OpenFlow is enabled on a per-port basis
  - OpenFlow-enabled ports can run existing features, i.e., MPLS/IP routing, concurrently with OpenFlow on the same port
  - Supports “protected” VLANs (next slide)
Hybrid Port Mode with Protected VLANs

- Protected VLANs
- A set of VLANs can be defined as protected
- OpenFlow rules cannot affect the traffic of protected VLANs
- Ingress frames on protected VLANs are not subject to OpenFlow rules
- Protection is supported in hardware
- Other VLANs (i.e., unprotected VLANs) are subject to OpenFlow rules
Volumetric Attack Mitigation

Out of the box solution

1. Data Center Devices
   - Send sFlow samples to the collector

2. sFlowAnalyzer
   - Analyze and report volumetric flow trigger

3. Traffic steering application with policy-based UI and REST APIs
   - Instruct controller to redirect volumetric flows

4. Brocade SDN Controller
   - Program OpenFlow1.3 rules in Brocade MLX and Brocade ICX
SDN Solutions Demo
How to install and configure a virtual router

• Configure the following:
  – Router interfaces
  – Redundancy
    • VRRP
  – Routing Protocol
    • OSPF, BGP
  – Firewall Configuration
How to install and configure a SDN Controller

• Configure the following:
  – Install SDN Controller and GUI interface
  – Verify Configuration
    • Connect to RESTCONF
    • Access API Docs
  – Install Controller Apps
    • Flow Manager
OpenFlow Demo

• Deploying the Controller in an OpenFlow Scenario
• Testing Host Tracker Function
  - Tracking Mode
  - Flood Mode
  - No Food Mode
  - Viewing Hosts and Topology
• How to create path between two hosts
RESTCONF Tools

- Retrieving the installed flow
- Deleting installed flow
- Getting the inventory using Postman
- Getting Operational Flow
SDN with Flow Optimizer Application

DDOS, redirect and mirror
Sample Python script interaction with SDN

- Add add view virtual router configuration
- Create and Remove Firewall
- Display Firewall Configuration
- Applying Firewall Rules
- Deleting virtual router from Controller
Thank you