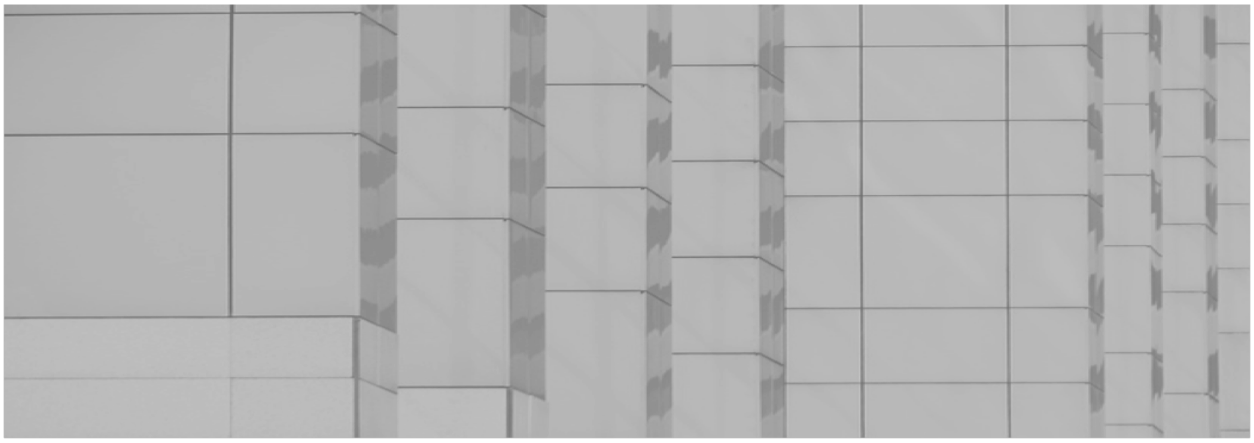
WHITE PAPER



SDN/NFV

Central Office Re-architected as Datacenter (CORD)

This white paper describes an architecture for the Telco Central Office with SDN, NFV, Cloud and commodity hardware to build cost-effective, agile networks that significantly lower CAPEX/OPEX and enable rapid service creation and monetization. CORD is a collaborative effort of AT&T and Open Networking Lab.

June 3, 2015

Changing Landscape

Service Providers are in the throes of unprecedented challenges. The proliferation of video traffic, mobile devices, and OTT services has pushed Service Provider networks into uncharted territory. At AT&T, for example, data traffic has increased 100,000 percent in the last eight years, and looking forward, plans are now underway to roll out ultra-fast fiber and access to 100 cities across the US.

Service Providers want to make their networks efficient, programmable, elastic and agile to meet the challenges of user bandwidth demands, as well as to create new revenue streams with innovative services and new business models. They want to benefit from both the economies of scale (infrastructure constructed from a few commodity building blocks) and the agility (the ability to rapidly deploy and elastically scale services) that cloud providers like Amazon, Google and Facebook enjoy today.

This cloud-inspired cost-effectiveness and agility is especially needed at the edge of Service Provider network—in the **Telco Central Office (CO)**. These facilities contain a diverse collection of purpose-built devices, assembled over fifty years, with little coherent or unifying architecture, making them both a source of significant CAPEX and OPEX and a barrier to rapid innovation. Moreover, large service providers operate thousands of such central offices. For example, AT&T currently operates thousands of central offices, each of which serves from thousands to millions of residential, mobile, and enterprise customers. By virtue of this scale, cost reduction in each of these Central Offices potentially translates into significant overall CAPEX and OPEX savings for the providers.

This white paper describes **CORD**—an architecture for the Telco Central Office that builds from existing work on SDN, NFV and commodity hardware in order to build cost-effective, agile networks with significantly lower CAPEX/OPEX and to enable rapid service creation and monetization. CORD is a collaborative effort between **AT&T** and **ON.Lab**.

Introducing CORD

CORD re-architects the Central Office as a datacenter. The basic approach centers on unifying the following three related but distinct threads:

- The first is **SDN**, which is about separating the network's control and data planes. This makes the control plane open and programmable, and that can lead to increased innovation. It also allows for simplification of forwarding devices that can be built using merchant silicon, resulting in less expensive white-box switches.

- The second is **NFV**, which is about moving the data plane from hardware appliances to virtual machines. This reduces CAPEX costs (through server consolidation and replacing high-margin devices with commodity hardware) and OPEX costs (through software-based orchestration). It also has the potential to improve operator agility and increases the opportunity for innovation.

- The third is the **Cloud**, which defines the state-of-the-art in building scalable services—leveraging software-based solutions, microservice architecture, virtualized commodity platforms, elastic scaling, and service composition, to enable network operators to rapidly innovate.

While it is easy to see that all three threads (SDN, NFV, Cloud) play a role in reducing costs, it is just as important to recognize that all three are also sources of innovative services that Telcos can offer subscribers:

- Control Plane Services (e.g., content-centric networking, virtual networks on demand, cloud-network binding)
- Data Plane Services (e.g., Parental Control, NAT, WAN Acceleration)
- Global Cloud Services (e.g., CDN, NoSQL DB, Analytics, Internet-of-Things)

In short, the goal of CORD is not only to replace today's purpose-built hardware devices with their more agile software-based counterparts, but also to make the central office an integral part of every Telco's larger cloud strategy and to enable them to support more attractive and meaningful networking services.

Commodity Hardware

The target hardware for CORD consists of a collection of commodity servers and storage, interconnected by a leaf-spine fabric constructed from white-box switches. An illustrative example is shown in Figure 1.

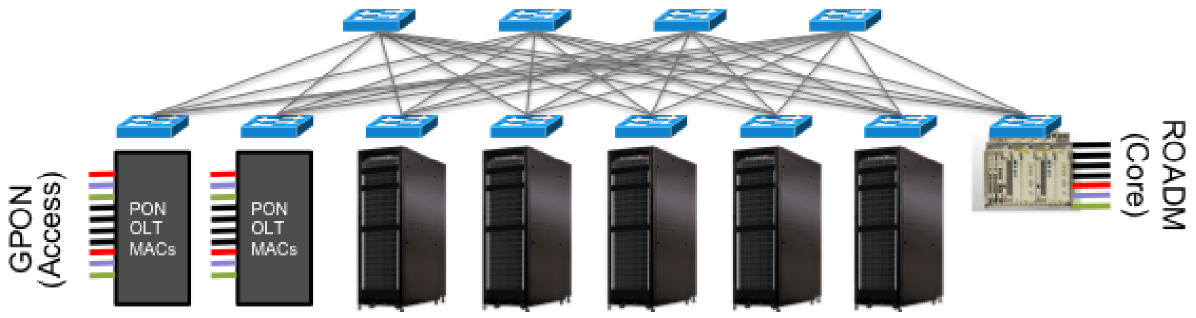


Figure 1. Target hardware, constructed from commodity servers, storage, and switches.

Although similar in design (albeit on a smaller scale) to a conventional datacenter, there are two unique aspects to this hardware configuration:

- The switching fabric is optimized for traffic flowing east-to-west—between the access network that connects customers to the central office and the upstream links that connects the central office to the operator’s backbone. There is no north-south traffic in the conventional sense.
- The racks of GPON OLT MACS commoditize connectivity to the access network. They replace proprietary and closed hardware that connects millions of subscribers to the Internet with an open, software-defined solution. (GPON is the example access technology in the reference implementation of CORD, but the same argument applies to other access technologies as well.)

Software Building Blocks

With respect to software, our reference implementation of CORD exploits three open source projects, as depicted in Figure 2:

- **OpenStack** is the cluster management suite that provides the core IaaS capability, and is responsible for creating and provisioning virtual machines (VMs) and virtual networks (VNs).
- **ONOS** is the network operating system that manages the underlying white-box switching fabric. It also hosts a collection of control applications that implement services on behalf of Telco subscribers. An ONOS subsystem (OVX), is responsible for embedding virtual networks in the underlying fabric, which is in turn accessed via OpenStack’s Neutron API.
- **XOS** is a service orchestration layer that unifies infrastructure services (provided by OpenStack), control plane services (provided by ONOS), and any data plane or cloud services (running in OpenStack-provided virtual machines).

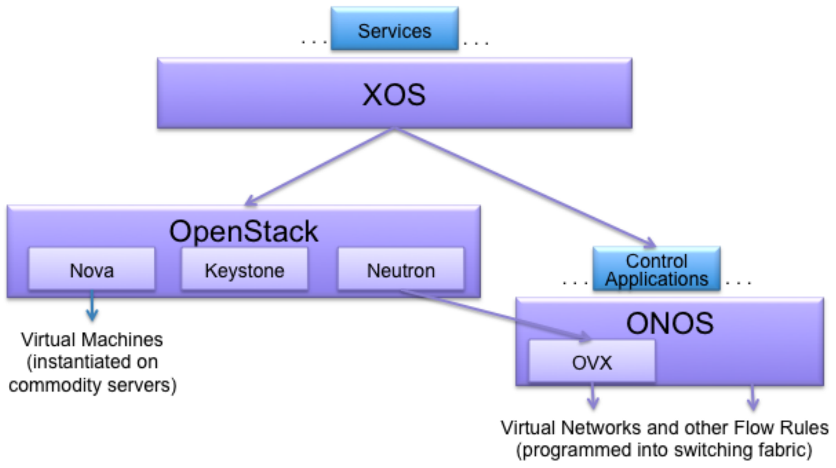

Services
XOS
OpenStack
Nova
Keystone
Neutron
Virtual Machines (instantiated on commodity servers)
Control Applications
ONOS
OVX
Virtual Networks and other Flow Rules (programmed into switching fabric)

Figure 2. Overview of the software architecture.

Transformation Process

Given this hardware/software framework, transforming today's central office into CORD can be viewed as a two-step process. The first step is to virtualize the devices, that is, turn each purpose-built hardware device into its software counterpart running on commodity hardware. The second step is to provide a framework that these virtualized software elements—along with any cloud services that the operator wants to run in the central office—can be plugged into, producing a coherent end-to-end system. The following two sections describe these steps to re-architect the central office in greater detail.

Virtualizing Legacy Devices

The first step in re-architecting the central office as a datacenter involves virtualizing the existing hardware devices, transforming each device into its software *service* counterpart and running it on commodity hardware. In the process, functionality is likely disaggregated and re-packaged in new ways.

This section walks through the process for the example set of devices highlighted in red in Figure 3, which includes Optical Line Termination (OLT), Customer Premises Equipment (CPE), and Broadband Network Gateways (BNG). We don't virtualize the Ethernet switch, per se, but it is effectively replaced by the switching fabric in Figure 1 (under control of the virtualized BNG).

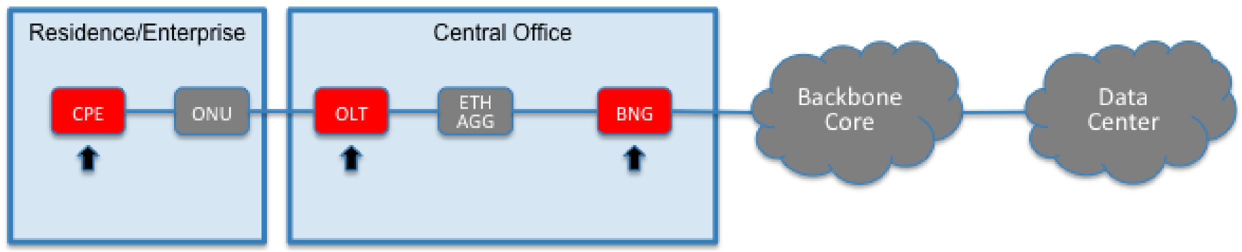


Figure 3. Legacy central office, including three physical devices to be virtualized.

Note that this white paper focuses on GPON technology and virtualizing the relevant pieces such as OLT. However the underlying principles are equally applicable to other access technologies including copper-based G.Fast networks and cable DOCSIS networks.

Benefits and Challenges

OLT is a large capital investment, involving racks of closed and proprietary hardware that terminate access for tens of thousands of subscribers per CO. Virtualizing OLT is especially challenging because, unlike many network appliances that are implemented by software running on vendor-branded commodity servers, OLT is implemented primarily in hardware. CPEs are currently distributed to tens of thousands of customer sites per CO, making them a significant operational burden. This is especially true when a service upgrade requires a hardware upgrade. BNGs are expensive and complex routers that have historically aggregated much of the functionality provided by a Central Office, making them difficult to evolve in an agile and cost-effective way.

The challenge is how to systematically transform such a diverse collection of devices into software running on commodity hardware. Our main insight is that there is a simple template for how each physical device is virtualized. It includes a combination of three elements: (1) merchant silicon, including both commodity servers and white-box switches; (2) a control plane function, which we refer to as the SDN element; and (3) a data plane function, which we refer to as the NFV element. While SDN and NFV are over-loaded terms, for our purposes, both are implemented by software running on commodity servers, where it is considered an NFV element if packet processing is entirely in software, and it is considered an SDN element if that software also controls a commodity switch through an open interface like OpenFlow. Said another way, NFV elements run *on* commodity hardware, while SDN elements run on commodity hardware (servers) but also *control* commodity hardware (switches).

The rest of this section shows how this pattern is applied to OLT, CPE, and BNG, resulting in virtual incarnations of each physical device. It is not the goal to preserve a one-to-one mapping between physical and virtual devices, and in fact, the opposite it true. Virtualizing legacy hardware provides an opportunity to disaggregate and refactor their functionality. Examples of such refactoring are presented throughout this section.

Virtual OLT (vOLT)

OLT terminates the optical link in the Central Office, with each physical termination point aggregating a set of subscriber connections. Given the number and cost of OLT devices in a CO, virtualizing the OLT has the potential to yield significant CAPEX and OPEX savings.

The first challenge is to create an I/O Blade with the PON OLT MAC. AT&T has started work at the Open Compute Project to develop an open specification for a GPON MAC 1RU “pizza box”, as shown in Figure 4. This board includes the essential GPON Media Access Control (MAC) chip under control of a remote control program via OpenFlow. This replaces an existing closed and proprietary OLT chassis (not shown) that integrates this GPON MAC chip with GPON protocol management, 802.1ad-compliant VLAN bridge, and Ethernet MAC functions. These other functions from the legacy device are being unbundled and implemented in software.

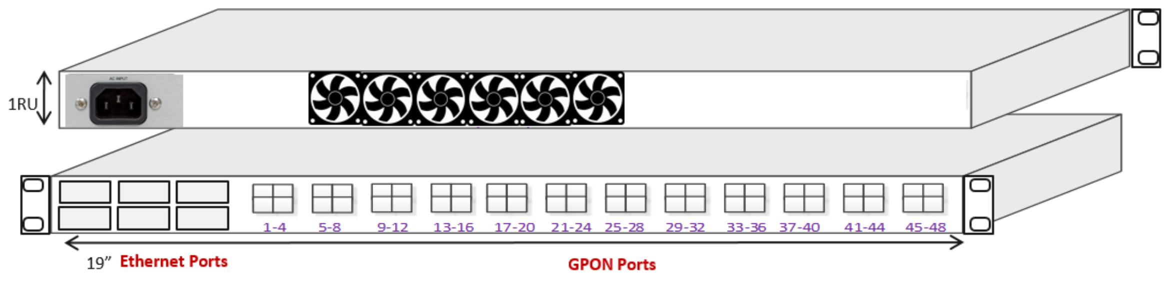


Figure 4. GPON OLT IO Blade.

The end result is that the access network interface is brought under the same SDN-based control paradigm as the white-box based switching fabric. In other words, virtual OLT (vOLT), is implemented as a combination of:

- Merchant Silicon: Commodity GPON interface cards in the central office.
- SDN: A control function that sets up and manages control plane functions of an OLT, like 802.1X, IGMP Snooping, and OAM.
- NFV: Hosts software functions (e.g., management and SDN agent for the merchant silicon) extracted out of the existing OLT.

The control function, implemented as an application running on top of ONOS, facilitates attachment and its authentication (AAA), establishes and manages VLANs connecting consumer devices (see next section) and the central office switching fabric on a per-subscriber basis, and manages other control plane functions of the OLT. The NFV data plane function for virtualized OLT simply sets up an SDN agent on the IO blade.

Virtual CPE (vCPE)

A CPE, sometimes called a “home router” or “residential gateway,” is installed in the customer’s premises. Because of their numbers, they are a significant source of CAPEX and OPEX costs, as well as a barrier to introducing new services. They often run a collection of essential functions (e.g., DHCP, NAT) and optional services (e.g., Firewall, Parental Control, VoIP) on behalf of residential subscribers. More sophisticated enterprise functions are also common (e.g., WAN Acceleration, IDS), but this whitepaper will only focus on residential functions. By extending the capabilities of CPE in the cloud, new value-add services as well as customer care capabilities can be provided where they could not before because of limitations in the hardware.

Our virtualized version of CPE, called vCPE, also runs a bundle of subscriber-selected functions, but it will do so on commodity hardware located in the central office rather than on the customer’s premises. That is, vCPE is implemented as a combination of:

- Merchant Silicon: Bare-metal switch located on the customer premises.
- SDN: Allows the NFV functions to control the merchant silicon.
- NFV: A data plane function that implements a per-subscriber service bundle.

In this case, a more complex set of per-subscriber services is provided than could be supported on the original CPE device, with the enhanced CPE implemented by a network-located VM that runs a bundle of functions on behalf of the subscriber. In other words, the “customer LAN” includes a remote VM that resides in the central office, effectively providing every subscriber with a direct ingress into the Telco’s cloud.

There is a wide range of implementation choices for subscriber bundles, including a full VM, a light-weight container, or a chain of light-weight containers. Our approach is to treat the bundle as a whole (roughly corresponding to a VM image or a container configuration file) as the standard representation, and leave the means by which subscribers select the set of functions to be included in their bundle as an implementation choice. For example, our proof-of-concept gives subscribers the ability to select from a small collection of functions (e.g., DHCP, NAT, firewall, parental filtering), but implements each through the proper configuration of a container (as defined by a corresponding Dockerfile).

Virtual BNG (vBNG)

A BNG is one of the more complex and expensive devices in a central office, providing the means through which subscribers connect to the public Internet. It minimally manages a routable IP address on behalf of each subscriber, and provides that subscriber with some type of network connectivity. In practice, however, the BNG also bundles a large collection of value-added features and functions, including VPNs, GRE tunneling, MPLS tunneling, 802.1ad termination, and so on. CORD's virtualized BNG, denoted vBNG, is implemented as a combination of:

- Merchant Silicon: Leaf-Spine fabric of white-box switches in the central office.
- SDN: A control function that routes flows between the CO and the Internet.
- NFV: N/A

In other words, vBNG is implemented as an ONOS-hosted control program that manages flows through the switching fabric on behalf of subscribers. No attempt is made to reproduce many of the auxiliary functions historically bundled into a BNG device, although in some cases (e.g., authenticating subscribers), that functionality is provided by another service (e.g., vOLT, in our case).

In general, it is more accurate to think of vBNG as providing each subscriber with their own "private virtual router," where the underlying fabric can be thought of as distributed router with "line cards" and "backplanes" instantiated by bare-metal switches. The vBNG control program then creates an IP network that routes between the attached, per-subscriber subnets.

Finally, our approach to vBNG highlights an example of refactoring. Historically, BNG is also responsible for authenticating the user, but that capability has been unbundled and moved to vOLT. This is because subscribers have to be authenticated *before* accessing vCPE, which use to reside in the home but has now moved into the Central Office.

Service Orchestration

This section focuses on the second step in re-architecting the central office as a datacenter: orchestrating the software elements resulting from the first step (plus any additional cloud services that the operator wants to run there) into a functioning end-to-end system.

Benefits and Challenges

Replacing hardware devices with software running in virtual machines is a necessary first step, but is not by itself sufficient. Just as all the devices in a hardware-based Central Office must be wired together in a meaningful way, their software counterparts must also be managed as a collective. This process is often called *service orchestration*, but if network operators are to enjoy the same agility as cloud providers, the abstractions that underlie the orchestration framework must fully embrace (1) the elastic scale-out of the resulting virtualized functionality, and (2) the composition of the resulting disaggregated (unbundled) functionality. A model that simply “chains” VMs together as though it is operating on their hardware-based counterparts will not achieve either goal.

Our approach is to adopt *Everything-as-a-Service (XaaS)* as a unifying principle. This brings the disparate functionality introduced by virtualizing the hardware devices under a single coherent model. The control functions run as scalable services (these functions run on top of ONOS, a scalable network operating system), the data plane functions run as scalable services (these functions scale across a set of VMs), the commodity infrastructure is itself managed as a service (this service is commonly known by the generic name IaaS), and various other global cloud services running in the central office are also managed as scalable services (as outlined below, CORD includes a CDN as an illustrative example).

XaaS is discussed in more detail in a companion white paper, but the main take-away is that all services support a logically centralized interface, called a *service controller;* elastically scale across a set of *service instances* (corresponding to VMs and OpenFlow switches); and are multi-tenant with an associated *tenant abstraction*.

Scalable Services, Not Virtual Devices

While terms vOLT, vCPE, and vBNG are used to refer to the virtualized counterpart of the three physical devices, they are not identifiable components (virtual or physical) in the resulting architecture. Instead, all three are packaged as services—each includes a multi-tenant service controller, and each scales independently across a set of service instances. While we could name these services according to their legacy counterparts, the new architecture no longer requires functionality to be bundled along the same boundaries as before. For this reason, it is more intuitive to think of the virtualization process outlined above as resulting in three generic, multi-tenant services (we retain the terms vOLT, vCPE, and vBNG to name specific software modules that implement some narrow aspect of each service):

- **Access-as-a-Service (ACCaaS):** Implemented by a vOLT control application running on ONOS, where each tenant corresponds to a *Subscriber VLAN*.
- **Subscriber-as-a-Service (SUBaaS):** Implemented by a vCPE data plane function scaled across a set of containers, where each tenant corresponds to a *Subscriber Bundle*.
- **Internet-as-a-Service (INTaaS):** Implemented by a vBNG control application running on ONOS, where each tenant corresponds to a *Routable Subnet*.

If a Content Distribution Network (CDN)—itself a scalable cloud service deployed throughout the operator's network, including caches in the central offices—is added to these three new services, we have an example of the three kinds of services outlined in the Introduction: a cloud service (CDN), a data plane service (Subscriber-as-a-Service), and two control plane services (Access-as-a-Service, Internet-as-a-Service). This results in the legacy central office depicted in Figure 3 being re-architected into the datacenter version shown in Figure 5.

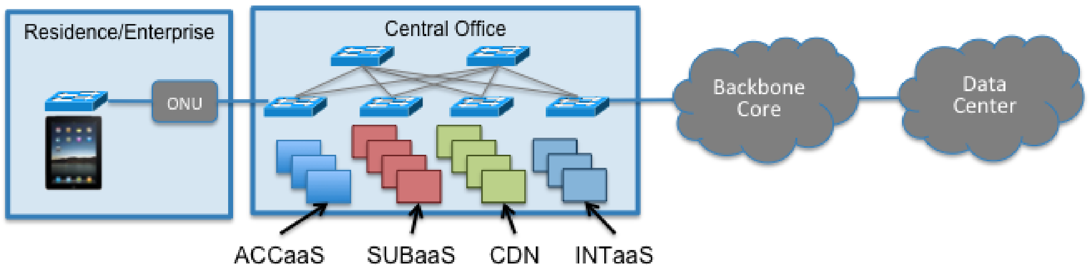


Figure 5. Four scalable services running in a central office, with a bare-metal switch located on the customer premises.

Software Architecture Revisited

Returning to the software architecture introduced in Figure 2, these four services all run in virtualized infrastructure managed by OpenStack; Access-as-a-Service and Internet-as-a-Service correspond to the vOLT and vBNG control applications, respectively, running on top of ONOS; and XOS manages the entire set of services and the relationships among them. This results in the specific software configuration shown in Figure 6.

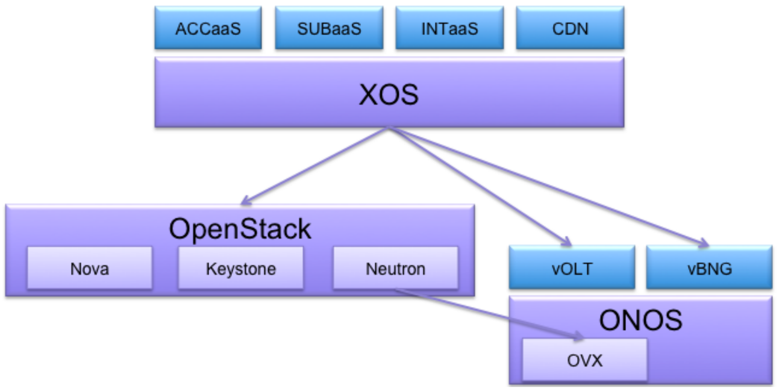


Figure 6. CORD software architecture populated with a specific set of scalable services.

The value provided by OpenStack is well understood—it provides the means to acquire and provision virtual machines and virtual networks. ONOS and XOS augment this basic capability by providing two useful high-level abstractions—ONOS manages the physical switching fabric and provides a *Network Graph* abstraction that is more convenient for control applications to program, and XOS enforces policies on the virtual infrastructure and provides a *Service* abstraction that is more convenient for network operators and service developers to program.

Also of note, XOS provides explicit support for service tenancy, which is to a collection of elastic cloud services in a datacenter what "service chaining" is to a collection of hardware devices in a traditional Central Office. It defines the system's security architecture (how VMs and VNs are interconnected) and it defines the system's composite functionality (how existing building block services are composed to create new services).

CORD implements the tenancy graph given in Figure 7. In this particular configuration, the home device is a tenant of ACCaaS (connecting the device to the subscriber VLAN); ACCaaS is a tenant of SUBaaS (connecting the subscriber VLAN to a container that implements a subscriber bundle); both SUBaaS and CDN are tenants of INTaaS (which provides a means for these services to reach the Internet); all four of these services are tenants of XOS (which represents these services as first-class objects); and XOS is a tenant of OpenStack (which provides building block VMs and VNs).

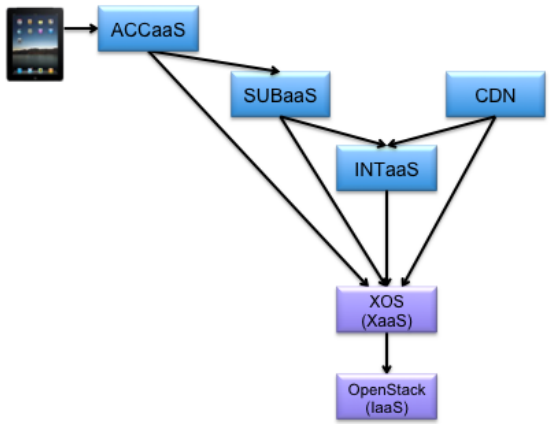


Figure 7. XOS tenancy graph among the set of services running in CORD.

Path to Real-World Deployment

CORD is a significant milestone in bringing cost effectiveness and agility to Telco Central Offices. It provides a holistic approach that is more effective and comprehensive than current approaches simply focused on moving functions from a dedicated piece of hardware to a virtual machine. In short, CORD enables service providers to take advantage of SDN, NFV and Cloud's full range of possibilities.

The plan is to build a fully functional reference implementation, starting with an initial proof-of-concept demonstration, and followed by validating the architecture through lab trials and eventually field trials, and hardening the system along the way based on trial data. Taken as a whole, the goal is to bring CORD close to readiness for commercial deployments in Service Provider networks.

Proof-of-Concept (June 2015)

Two complementary proofs-of-concept are being showcased at the Open Networking Summit in June.

The first proof-of-concept demonstrates the end-to-end CORD vision. It consists of the software architecture shown in Figure 6 running on a minimal server cluster with a Top-of-Rack white-box switch (Pica8). It also includes a bare-metal NetGear switch in the home and a prototype GPON interface card that uses the PMC Sierra PON OLT MAC chip in the cluster.

Because the switching fabric is minimal, the Internet-as-a-Service (vBNG) component is also minimal—it simply provides each tenant with a routable IP address. The Subscriber-as-a-Service (vCPE) includes a collection of residential services, such as parental control over web access, and a CDN caches Internet content on behalf of third-party content providers.

The second proof-of-concept demonstrates a high performance switching fabric and includes a more complete set of BNG functionality. As shown in Figure 8, it is configured as a leaf-spine topology with four (top-of-rack) leaf switches (48X10G ports and 6X40G uplinks) and four spine switches (32x40G ports). This fabric can be thought of as a distributed router, with the leaf switches corresponding to “line cards” and the spine switches corresponding to the router “backplane.”

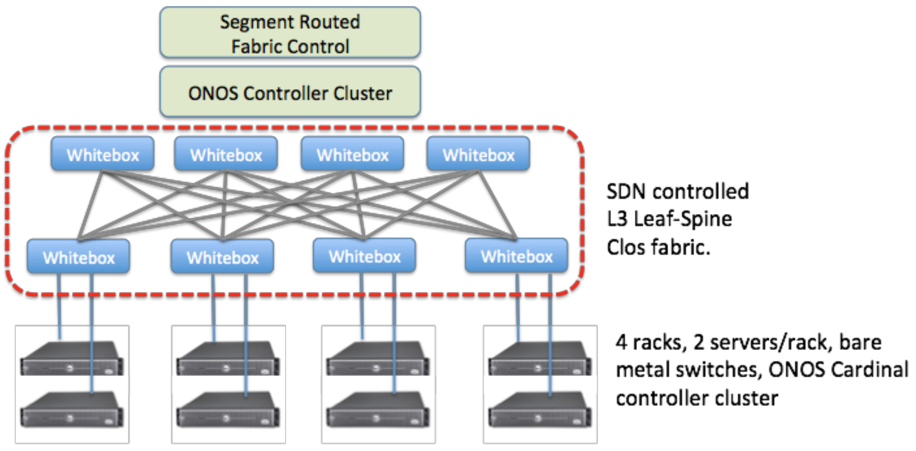

Segment Routed Fabric Control
ONOS Controller Cluster
Whitebox
SDN controlled L3 Leaf-Spine Clos fabric.
4 racks, 2 servers/rack, bare metal switches, ONOS Cardinal controller cluster

Figure 8. CORD Fabric Demo at ONS.

In addition, the fabric can automatically perform traffic-engineering. For example, it allows external analytics to detect elephant flows that cause congestion in the uplinks, and take immediate, direct action by moving the flows to under-utilized links via the fabric control application running on the controller cluster.

CORD Trial POD (December 2015)

During a second stage, with a targeted delivery in December 2015, the above two PoCs will be integrated into a single comprehensive reference implementation. This will be followed by scaling, hardening, and enhancing the constituent services to offer a more carrier-grade collection of functionality. This system, called “CORD Trial POD”, will be available for evaluation throughout 2016, with continual upgrades and hardening based on that experience.

Summary

CORD is a revolutionary effort to transform legacy Central Offices in the Telco network. In the new Central Office re-architected as a datacenter, closed and proprietary hardware is replaced with software running on commodity servers and switches. This software, in turn, is managed and orchestrated as a collection of scalable services. In doing so, CORD’s goal is to demonstrate the feasibility of a Central Office that enjoys both the CAPEX and OPEX benefits of commodity infrastructure, and the agility of modern cloud providers.

Open source software plays a critical role in CORD, which leverages three projects: OpenStack to provision the virtual infrastructure, ONOS to manage the switching fabric and host control applications, and XOS to support and manage services. Open source hardware, based on Open Compute and similar projects, is a great fit for CORD as well.

CORD is a game changer not only by virtue of its architecture and related benefits, but also because of its ambitious goal to provide real world proof points in support of trials in service provider networks in 2016.

ONOS
Software Defined Transformation of Service Provider Networks
Join the journey @ onosproject.org

About ONOS

ONOS is the open source SDN network operating system for service provider and mission critical networks, architected to provide a resilient, high performance SDN control plane featuring rich northbound/southbound abstractions and interfaces to support a diversity of management, control, service applications and network devices. ONOS(Avocet release) was open sourced on December 5th, 2014. Blackbird, the second ONOS release in Feb 2015, demonstrated SDN control plane performance, scale and HA leadership.

ONOS ecosystem comprises ON.Lab, organizations who are funding and contributing to the ONOS initiative including Tier 1 Service Providers - AT&T, NTT Communications, SK Telecom, and leading vendors including Ciena, Cisco, Ericsson, Fujitsu, Huawei, Intel, NEC; members who are collaborating and contributing to ONOS include ONF, Infoblox, SRI, Internet2, Happiest Minds, CNIT, Criterion networks, Black Duck, Create-Net, KISTI, KAIST, NAIM, Kreonet, and the broader ONOS community. Learn how you can get involved with ONOS at onosproject.org.