THE LIPPIS REPORT

The Resource for Network/IT Business Decision Makers

Nexus 9000 Programmable Network Environment

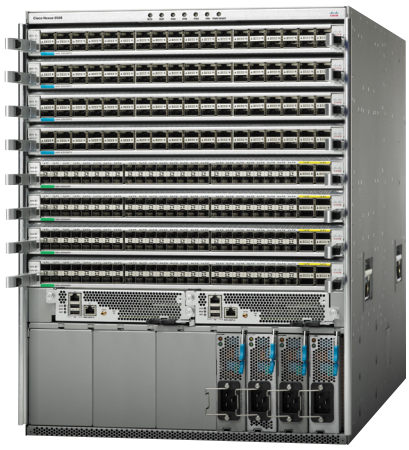
Lippis Report Research Note
October 2013

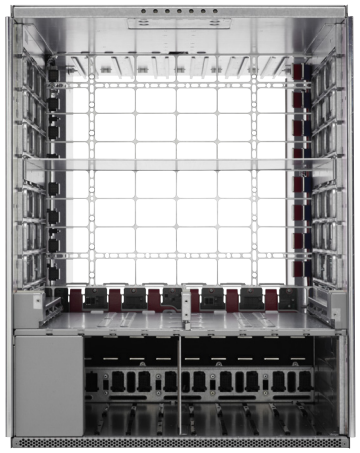
Cisco Delivers a Programmable Networking Environment with New Nexus Platform, and the World Changed

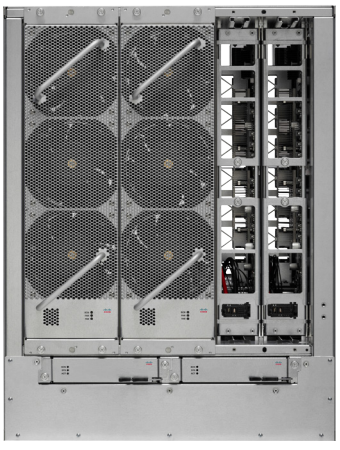
Software-Defined Networking or SDN is entering its second stage or SDN 2.0. The definition of SDN 1.0 being the separation of data and control plane offered little to DevOps professionals or network engineers seeking a programming environment to customize their network and automate operational tasks. SDN 2.0 adds a programming environment to networks that starts to deliver on the Software-Defined aspect of modern day computer networking. It's not an understatement to say that programmable networking will change not only networking, but also the Information Technology industry broadly. As such, interest is keen to understand how networking is moving toward a programmable platform for developers and network engineers. Cisco is offering the most comprehensive set of network programming options, with its new Nexus product line through its acquisition of Insieme Networks, that delivers programming tools through an enhanced version of the Nexus Operating System (or NX-OS), which ranges from direct switch programming via its built-in Linux BASH environment, open RESTful APIs with JSON and XML support, direct integration with Python, etc. Most network engineers will opt to enroll in a Linux programming course versus obtaining another CCIE certification after reading this Lippis Report Research Note. In this research note, we review Cisco's new programming environment detailing its options, use cases and industry impact.

Cisco recently announced its new Nexus 9000 family of data center switches, which comprises the Nexus 9300 series of fixed switches, and the Nexus 9500 series modular switches. While this line up is impressive in terms of performance, power efficiency and scale, its programming environment sets a new industry standard and direction. Today's networks are restricted to configuration management via Command Line Interface, or CLI, but what if applications can call upon network resources automatically or if application developers are provided access to network state, topology, VM port group and performance information? How might applications change and user experience improve? How may a programmable networking environment enable automated provisioning and orchestration of network resources triggered by new or modified workload deployment? Could a programmable network enable a new era of IT and an industry of network aware applications just when the Internet of Things is starting to emerge?

The entire concept of how we think about networking changes in SDN 2.0. Networking equipment was vertically integrated with a single vendor providing hardware, operating system and protocols. When new features were required, the vendor community would deliver a broad industry solution via their product development phased review process, which usually takes two plus years. Some features are so important to be standardized and would progress through the IETF, IEEE, ITU or other such industry organizations, taking another 18 to 24 months to be ratified. SDN 2.0 offers a way to accelerate innovation by opening up network resources to DevOps and network engineers through a set of programming interfaces to customize networks in ways which were previously difficult to accomplish or just outright impossible.



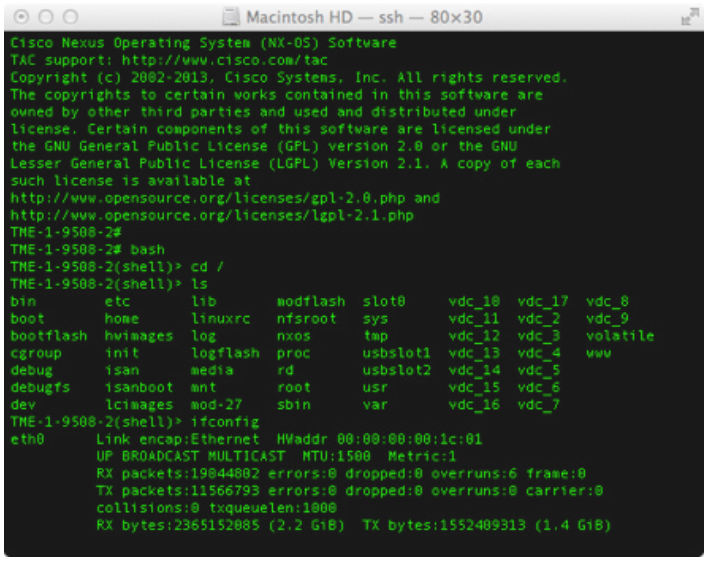




Cisco's new Nexus product line acquired from Insieme Networks has the industry's broadest networking programming environment to date. Insieme engineers started with the existing Nexus Operating System (NX-OS) that powers the Cisco Nexus platforms, such as the Nexus 7000 and 5000. In partnership with customers, Cisco cataloged requirements of a next generation data center network platform, and what was loud and clear was that the market demanded programmability, configuration automation and much more network visibility. These set of requirements was also made loud and clear by 180 IT executives who attended the last Open Networking User Group conference hosted by Fidelity Investments in Boston on February 2013. The result is that the enhanced NX-OS that runs on the new Nexus 9000 series of data center switches is equipped with a suite of open programming tools and functions that can either be leveraged independently, or put to work in unison with other platform capabilities.

Open Systems Approach to Exposing Switch Data

Cisco's programming environment is based on an "Open Systems" approach where unconstrained access to network switch data in the platform is now more readily available. This includes things such as Interface Counters, Resource utilization information (CPU and forwarding), and generic SNMP-OIDs or Simple Network Management Protocol-Object IDentifiers. Cisco has also provided direct access to the complete Linux BASH environment, allowing full access to functions and processes in the user space and switch control plane that programmers can leverage to gain additional device visibility and performance information. Direct switch ASIC/chip-level counters access is now available via the Broadcom Shell SDK and Cisco's custom ASIC as the Nexus 9000 is based on a custom and merchant silicon hybrid architecture. In addition, resource monitoring, including detailed switch buffer usage information is also accessible and much more available for the administrator to extract and leverage in whatever tools or scripts they desire.

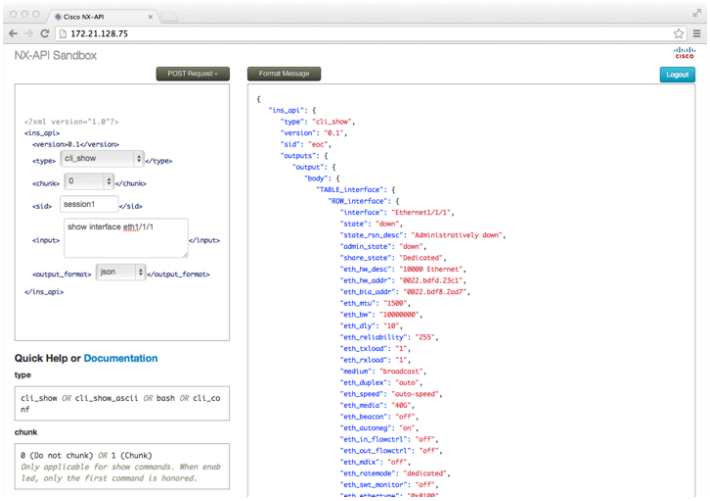


On-Switch Programming Environment

The above are a sample of the type of data that's now available on the new Nexus 9000 product line. But data needs to be contextualized for it to be useful information. In order to accommodate this, the Insieme engineers added an "On Device" or on-switch programming environment to perform a certain amount of processing or to contextualize the data into something that's useful. For example, LXC, or LinuX Containers, is an operating system-level virtualization method for running multiple isolated Linux systems (containers) on a single host. LXC can be used to install custom applications on the switch that is interesting for a given customer. As applications run in different containers, there is inherent isolation so that no single container impacts the performance or stability of another or the underlying switch operations. In addition to LXC, a Python Shell is supported for on-switch scripting, where operations teams can choose to install Python scripts on the switch to automate different functions, or access these scripts remotely as well. Cisco also plans to offer the complete Cisco onePK development environment on the Nexus 9000 series platforms, providing a comprehensive environment for developers to write applications that can be deployed across a variety of Cisco IOS and NX-OS platforms.

Integrating Network Information into Applications

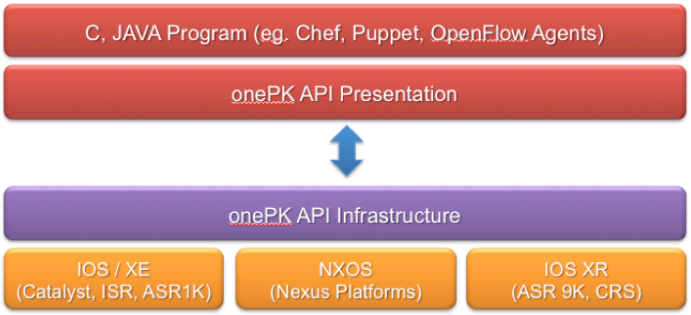
With network data and information exposed, DevOps and network engineers can now integrate network device information into their backend systems and applications plus develop streamlined network management and monitoring tools. One of the keys to opening up access to the power of networking is found in a new RESTful NeXus Application Programming Interface, or NX-API, which provides easy-to-use, web-based APIs for network engineers and DevOps integration. With this, DevOps teams can enhance and develop network aware applications easily through web-based tools that are so prevalent in the industry today. In addition to the NX-API, NetConf, an XML-based IETF network management protocol that enables install, manipulate and delete network device



configuration data, is supported as well as traditional SNMP-MIBs and CLI.

While SDN 1.0 focused on programming network routing tables through the OpenFlow protocol, SDN 2.0 expands the definition of programmable networking significantly and provides multiple approaches to forwarding table manipulation. For example, the new Cisco Nexus 9000 product line offers two modes to program network-forwarding tables – through LXC directly on a switch or through Cisco's onePK development environment.

The advantage of programming applications directly through in the LXC is that a greater amount of control over forwarding constructs can be provided. OnePK, on the other hand, provides a well-defined environment where applications written over onePK can run across any Cisco platform that supports onePK, which will eventually include most of Cisco's routing and switching product lines. An OpenFlow agent is an example of an application that can be instantiated within the Cisco onePK container that can also directly control the forwarding logic of the switch. This allows a Northbound network controller, such as Cisco's eXtensible Network Controller (XNC) or the OpenDayLight (ODL) Controller, to manage and program multiple platforms. These controllers will be able to program any network devices, including the Nexus 9000 series platforms, that present either OpenFlow agents or directly through Cisco onePK. Therefore, developers will be able to command control over the switch's forwarding logic, either directly via LXC or through Cisco onePK or its OpenFlow extensions.



Network Configuration Automation

On top of the above programming options rest a set of automation tools that run natively on the Nexus 9000 series platform. These include DevOps tools like Puppet or Opscode Chef, and an OpenStack Neutron plugin so that large-scale network resources and services can be provisioned at the time of workload creation and movement. In addition, Cisco's existing Embedded Event Manager, or EEM, for real-time network event detection and onboard automation is supported as well as Power on Auto Provisioning, or PoAP, which allows the operations team to specify the switch's software image to load at initial configuration upon power up sequence from a central point, much like PXE-booting a server.

The Nexus programming environment exposes and provides access to ASIC level device and network data. It also provides a set of onboard programming tools to process this data into information and various APIs to integrate device and network information into backend systems and applications. The ability to automate configuration, management and monitoring is provided by a set of Linux and network tools that leverage the underlining-programming environment. In essence, the Insieme engineers have delivered a programming stack for open networking. So how does the Nexus programming environment benefit data center network operations, IT application delivery and user experience?

First and foremost, the Nexus programming environment, like all programming environments, unleash creativity and innovation and usually take on a life of their own. The same will be true of the Nexus programming environment, but the first wave of innovation will be operationally focused, such that networks will be provisioned faster, thanks to the above automation tools. There will be fewer configuration errors as manual switch-by-switch configuration is replaced

with defined and consistent automation models, thanks to embedded tools, such as Puppet/Chef, Structured APIs with XML/JSON; Northbound cloud orchestration tools such as OpenStack; Programmable SDN Controllers, such as Cisco XNC/ODL; and even embedded multi-node management tools, such as an integrated XMPP Pub-Sub Bus.

One of the biggest immediate benefits will be a common skill set that's leveraged across Linux-based server teams and networking teams. In short, these two teams will now have a common language and skill set enabling a range of collaboration that was previously not possible. Application visibility will increase, enabling DevOps and NetOps to increase efficiency across the IT infrastructure to improve application performance and user experience. In addition, NetOps will be able to automate network data visibility access through simple Open APIs to ease troubleshooting and reduce overall time-to-recovery versus typing switch-by-switch CLI commands. It's the Nexus programming environment architecture of being open, flexible and simple, thanks to new open web-based interfaces, that will enable the rapid development of automation and management platforms.

There will be a new, automated process in which networks are provisioned and managed, thanks to the Nexus programming environment. At day 0, NetOps and DevOps will automate network configuration and provisioning leveraging PoAP and Puppet/Chef. At day 1, when provisioning shifts toward management, NetOps will leverage global environmental provisioning via Puppet/Chef to tweak and optimize their network; if and when additional tuning is needed, the administrator can leverage Open APIs, or even directly through CLI commands, leveraging the XMPP Pub-sub bus to control a group of switches. Puppet and Chef's main value in network provisioning is to deliver a consistent intent driven framework to a large number of networking devices, such as globally provisioning ACL configuration, define VLANs, VxLANs, mappings, routing, etc. In addition, NX-OS adds support for image patching so that incremental enhancements or bug fixes can be introduced onto the platform without a complete upgrade of the NX-OS image. Furthermore, these patches can be automatically rolled out across the infrastructure in conjunction with Puppet/Chef.

Greater Network Visibility

Network visibility will be much more granular, thanks to the Dynamic Buffer Monitoring capability of the Nexus 9000. This allows information, such as real-time buffer utilization statistics, to be exposed via the CLI or directly through the NX-API. Insieme is also providing a capability called vTracker, which gathers data from VMWare's vCenter to provide information, such as VM location, VM port group, etc., to provide virtualized infrastructure visibility which can also be returned through structured APIs. Even routine NetOps troubleshooting tasks, such as measuring ping return time across a range of switches, can now be automated with a simple Python script. In fact, accessing the NX-API via the web interface allows NetOps to input a CLI command to a switch and receive its response in structured XML or JSON code. This can help the operations team build automated troubleshooting tools or sent to a program like Gephi to generate a visual display across the entire network, enhancing troubleshooting and saving NetOps precious time.

From the above, there are two groups that will gain the most from the programmatic capabilities engineered into NX-OS on the Nexus 9000 series of switches – those groups are organizations with an established DevOps environment, as well as those with more traditional NetOps groups.

First, DevOps have the skills to leverage the above tools and the new Open Systems access capabilities of the Nexus 9000, including Chip-level counters, BASH access, LXC, Chef/Puppet automation tools, Python to create their own custom scripts and leverage the NX-API into other tools with which they are already familiar. In short, DevOps have the resources to customize network device data and use it in the way that's important for them to either deliver competitive business value or to reduce OpEx through automation. Hyperscale web companies, cloud providers, service providers and large enterprise IT organizations fit into this group.

The second group is the network engineering teams who'll leverage the Nexus programming environment for network device automation. NetOps will leverage the same tools utilized on servers to bootstrap (PoAP) plus globally provision network configurations and automate code loads with Puppet/Chef.

This group may not necessarily have the programming skills to expertly author scripts that DevOps possess, but it's not a stretch to think that a wide range of scripts will be open sourced and made available by the DevOps community on open forums, such as GitHub or StackOverflow, and will be leveraged by NetOps teams by making modifications to adapt these scripts for their own environments.

Developers using the Nexus programming environment will first focus on automation, management and monitoring efficiency, but then will quickly move toward application enhancement and creation, thanks to network information access. This is where IT can fundamentally change as networking becomes programmable impacting server, storage and application assets. With billions of devices plugged into networks scaling to trillions over the next business cycle, thanks to the Internet of Things, programmable networking will unequivocally reshape the IT landscape and how we live our lives. Programmable networking will usher in a vast domain of opportunity by creating an environment to unleash creativity and innovation. It's the beginning of a very exciting industry and economic cycle on the scale of the internet.

One implication of this shift is that DevOps will have an increasing say over network purchases and design over time as SDN 2.0 takes hold. DevOps influence will increase significantly, especially as these Linux programming tools are used to deliver customer visible features and infrastructure cost savings. By now, network engineers may be asking themselves "where can I enroll in that Linux programming course?" Well, both network engineers and DevOps can start here.

About Nick Lippis

Nicholas J. Lippis III is a world-renowned authority on advanced IP networks, communications and their benefits to business objectives. He is the publisher of the Lippis Report, a resource for network and IT business decision makers to which over 35,000 executive IT business leaders subscribe. Its Lippis Report podcasts have been downloaded over 200,000 times; ITunes reports that listeners also download the *Wall Street Journal's* Money Matters, *Business Week's* Climbing the Ladder, *The Economist* and The *Harvard Business Review's* IdeaCast. He is also the co-founder and conference chair of the Open Networking User Group, which sponsors a bi-annual meeting of over 200 IT business leaders of large enterprises. Mr. Lippis is currently working with clients to design their private and public virtualized data center cloud computing network architectures with open networking technologies to reap maximum business value and outcome.

He has advised numerous Global 2000 firms on network architecture, design, implementation, vendor selection and budgeting, with clients including Barclays Bank, Eastman Kodak Company, Federal Deposit Insurance Corporation (FDIC), Hughes Aerospace, Liberty Mutual, Schering-Plough, Camp Dresser McKee, the state of Alaska, Microsoft, Kaiser Permanente, Sprint, Worldcom, Cisco Systems, Hewlett Packet, IBM, Avaya and many others. He works exclusively with CIOs and their direct reports. Mr. Lippis possesses a unique perspective of market forces and trends occurring within the computer networking industry derived from his experience with both supply- and demand-side clients.

Mr. Lippis received the prestigious Boston University College of Engineering Alumni award for advancing the profession. He has been named one of the top 40 most powerful and influential people in the networking industry by *Network World. TechTarget*, an industry on-line publication, has named him a network design guru while *Network Computing Magazine* has called him a star IT guru.

Mr. Lippis founded Strategic Networks Consulting, Inc., a well-respected and influential computer networking industry-consulting concern, which was purchased by Softbank/Ziff-Davis in 1996. He is a frequent keynote speaker at industry events and is widely quoted in the business and industry press. He serves on the Dean of Boston University's College of Engineering Board of Advisors as well as many start-up venture firms' advisory boards. He delivered the commencement speech to Boston University College of Engineering graduates in 2007. Mr. Lippis received his Bachelor of Science in Electrical Engineering and his Master of Science in Systems Engineering from Boston University. His Masters' thesis work included selected technical courses and advisors from Massachusetts Institute of Technology on optical communications and computing.