PICA8®
OPEN NETWORKING

WHITEPAPER

Bare Metal Networking—Leveraging "White Box" Thinking

Bare Metal Networking—Leveraging "White Box" Thinking

Switch commoditization has created new approaches to lower CapEx while improving functionality

Conversations are swirling throughout the tech industry about whether white box switches are disrupting the networking industry in the same way that white box manufacturers helped commoditize the server industry.

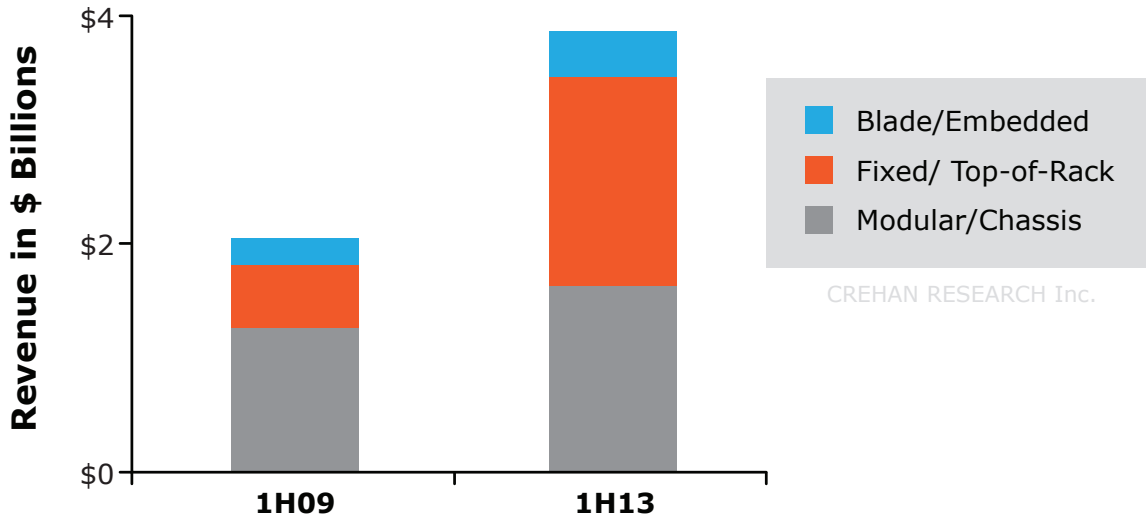


Figure 1. Data Center Ethernet Switch Revenue

If this InfoWorld article where the authors predict a sizeable uptick in white box sales over the next five years is not enough to persuade you—see Figure 1, consider that Cisco CEO John Chambers himself has recently discussed the threat of white boxes eroding his company's margins.

Conceptually, white box switching helps leverage the idea of communization of mass-produced hardware, while creating an operational abstraction between the "metal" (in our case, white box switches from original device manufacturers, or ODMs) and the network operating system (OS).

Bare Metal, White Box Switches Defined

The "white box" phrase has been used to describe generic PCs built by non-branded manufacturers for many years. "Bare metal" refers to the hardware only, and is similar conceptually to a server with no loaded software. Regardless of server hardware differences, all servers get our choice of a server OS that is mostly unified across a specific operation. In recent years, the networking industry has embraced "bare metal" and "white box" terminology. For example, the recognized network stack is considered to include the hardware or "white box" network OS and applications that run on the OS. So the bare metal switch is the substrate or base of the stack. Similar to the server side, "bare metal" means you just get the metal and no OS is installed. You need to have the expertise to work with that or have a solution integrator install the OS.

Both the terms ODM and OEM (original equipment manufacturer) are related to the manufacturing industry. An OEM builds a product according to its own specification, while an ODM assembles a product according to another company's specification.

In the past 10 years, the main networking OEM vendors (Cisco, Juniper, etc.) have specified switches built by ODMs and load their own software on top of them in order to sell them as a bundle. These switches look just like any other switches and are assembled by the same companies that build white-box servers. Accton, Quanta Computer, and Foxconn all make white box switches (standard 1-RU, 48-port Ethernet switches). As a collective, these companies have more than $200 billion of manufacturing muscle annually.

The commoditization that has resulted from this business model has caused networking hardware specifications to become almost identical, and with that, a subsequent explosion of OEM vendors have arrived on the scene. The Open Compute Project (OCP) created by Facebook has also driven hardware design.

As a result of all of these developments, the industry is moving toward an open networking model that has already created opportunities for new players to become OEMs (Dell) and ODMs (Accton) in the switch market. Both will offer standard bare metal switches.

Better CapEx or Better Operational Idea?

Open networking can help reduce costs far beyond offering customers some cheaper hardware. In the data center network, in terms of devices, switches are mostly fixed configuration. The OS is not open and is tied to the switch. When the hardware is changed (especially if you are bold enough to change vendors), many of the tools and applications need to be painstakingly modified, accommodating new application programming interfaces (APIs) and MIBs or other proprietary interfaces. Because this environment is so closed, proprietary and inflexible, changing any layer of the network ecosystem is a costly exercise. If you want to change your management tools, it impacts the OS below. If you want to modify the OS (if the vendor allows this), you may be required to change the switch hardware or the tools used to deploy and manage the switch.

What if the OS was open and truly separated from the hardware? What if you could swap out the hardware without having to change the OS or any of the changes you made to this open OS? What if you could change the management tools or provisioning system without having to worry about the complexity it may create with your proprietary or closed programming interfaces? Having a network that is closer to the server environment (open and separated) may give customers the chance to really apply lifecycle management thinking to the network. Your data center stack is built on racks, where each rack is a building block.

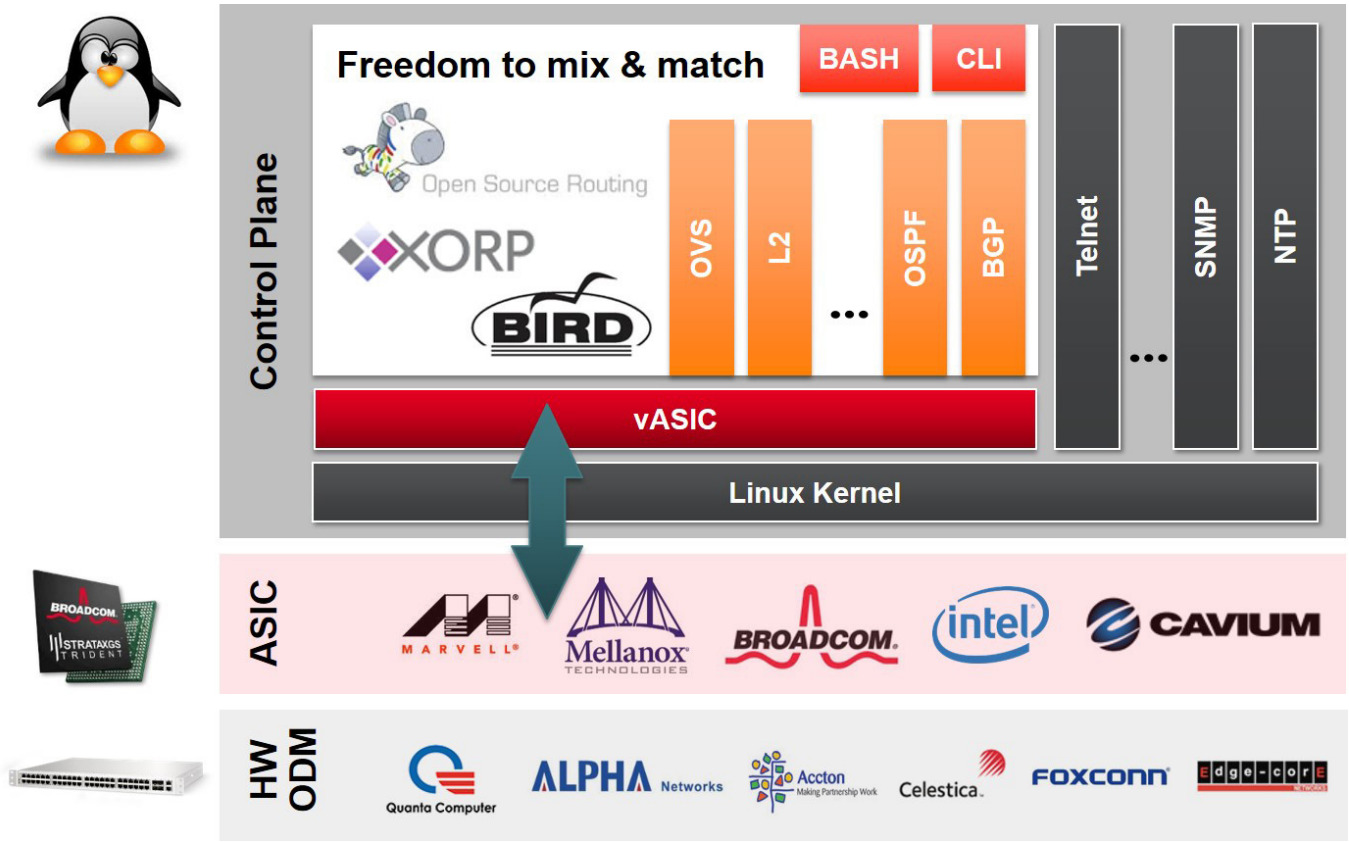
Why is Pica8 Offering a Customized Network OS Available Through a Software Model and Through Pre-loaded Systems?

Transitioning to white box switches could be a complex enough task without forcing users to find the right combination of supported white box switch and network OS. Many users will want the OS integrated with the white box switch, which is why Pica8 sells switches with software installed on them. We also sell our PicOS separately for those who want to install the OS themselves, but many of our customers want the whole package on one system so there's only one entity responsible for service and support. In either case, Pica8 supports the entire system and our relationships with ODMs include knowing who to call and we ensure your "out of the box" experience goes beyond the initial installation.

What Makes a Network OS Portable?

In a typical first meeting, we frequently get asked if they can port a version of our OS on their existing Cisco switches. At first blush, it makes sense but let's examine the three key issues that need to be addressed to truly make an OS compatible with a "bare metal" switch.

For portability, the bare metal device must provide three key component areas within the device to the OS, similar to any board BIOS: the CPU Board Support Package (BSP), the "U-Boot" (or universal boot loader), and the ASIC's APIs. All three collectively need to be matched to the network OS in order for you to port an OS onto that same piece of metal. This is why all the white box, software defined networking (SDN) vendors either sell the complete switch and OS solution or provide a list of pre-qualified devices so you can directly buy the hardware. Similar to early PC days, compatibility requires close coordination and sharing of drivers. Both paths deliver the same solution. We will mainly focus on these three component areas so you can assess the level of portability of your network OS.



The CPU and the Motherboard

Thanks to PCs and the server market, making an OS work with any CPU is the easiest part of the puzzle to solve. Any modern Linux OS will have the needed drivers and kernel support for all common CPUs. You can assume this is not a problem, but the motherboard consists of other chips than the CPU. You need to make sure you have full access to the driver code of all chips to ensure that complete functionality can be accessed through the new OS. This need is new and particularly topical for Ethernet switches because servers don't have U-Boot functionality–they have BIOS.

The obvious approach here is to use Linux. It is the standard base OS on the server side, and IT is looking for solutions that enable a single management framework between the network operations and development operations sides of the house.

The Universal Boot Loader (U-Boot)

The U-Boot is essential in both white box switches (think PXE Boot on servers). The Linux OS holds all the details on the shell of the box itself (the fans, lights and USB ports as examples) that map directly to the actual box. U-Boot loaders are hidden from users and are one of the reasons you must have pre-qualified white boxes. Every manufacturers have a different U-Boot loader on their devices; and even within product families of the same vendor, the U-Boot code could be different.

Open Source Initiatives

The Open Compute Project (OCP) is driving initiatives to help foster the idea of open networking and therefore tools that help users build an open stack, from the bare metal switch to the operating system and applications on top.

The Open Network Install Environment (ONIE) is an open source initiative driven by a community of next-generation networking vendors to define an open "install environment" for bare metal network switches. This is a comparable model to servers' PXE boot that allows a centralized repository of code to be distributed on demand to the different machines (servers or network devices), such as existing ODM switches and the upcoming OCP Network Switch design. ONIE enables a bare metal network switch ecosystem where end users have a choice among different network OS.

Open Network Linux (ONL) is a Linux distribution for "bare metal" switches. ONL uses the Open Network Install Environment (ONIE) to install Linux onto on-board flash memory. Open Network Linux is a part of the OCP and is a component in a growing collection of open source and commercial projects.

The ASIC

The ASIC represents the heart of the switching engine. The ASIC provides high packet throughput and is key to how switching helps build a high-bandwidth network. Whether you leverage merchant silicon (Broadcom, Cavium, Intel, Marvell, Mellanox), or proprietary ASICs, you need to have a means to connect the chip to your software stack. That is through the ASIC API or software development kit (SDK). The purpose of the OS and the CPU is to program the ASIC to provide the required functionality based on the configuration and ASIC. Every ASIC has a different approach to packet handling, memory management and feature set.

The API is a collection of functions interfacing for control and setup of ASIC chips. With an API, low-level registers are invisible to users who just need to use corresponding API functions to achieve specific functions on the devices. The API is the minimum software delivery from chip vendors and it is usually OS independent. Users could port an API to any OS and hardware and integrate it into their own platform as an application. Examples of API functions include setting up a VLAN or an ACL entry, or adding a static MAC entry.

The Software Development Kit (SDK) usually contains BSP, API and applications. The SDK integrates and uses an API for controlling ASIC chips. Applications include control modules, protocols and management software. With an SDK, the user could build it out and have it run over specific boards (e.g. vendors' reference or validation boards). The SDK provides a platform for end users' further development since low-level controls, the kernel and some applications are already available and could be used as a foundation upon which to add their own applications if the underlying boards are compatible.

Additional Tools Borrowed from the Server Side

As is with servers today, networking devices also can be thought in a similar lifecycle manner, from the initial loading of the OS, to provisioning the device for a specific application, and then finally decommissioning the device. Leveraging the experience and wealth of tools from Linux and the generic server industry. Part of the lifecycle is provisioning the device for the first time, which goes beyond loading the OS on the bare metal switch. The idea is to get the switch into production.

Zero Touch Provisioning (ZTP) is a generic name that implies full deployment without human intervention. Normally, ZTP will enable you to load, provision and automate any post-install process of new switches in your network automatically, without manual intervention. When you physically connect a switch to the network and boot it with a default configuration, it attempts to upgrade the OS software automatically and auto-install a configuration file from the network.

The switch uses information that you configure on a Dynamic Host Control Protocol (DHCP) server to determine whether to perform these actions and to locate the necessary software image and configuration files on the network. If you do not configure the DHCP server to provide this information, the switch boots with the pre-installed software and default configuration.

ZTP Applications:

- **Using templates with Puppet/Chef to configure switches in a compute cluster:** Extending the model of cluster server provisioning, these tools create recipes or templates to quickly deploy additional racks of servers in the cluster. Along with the server templates, the manager adds a template for the Top of Rack (ToR) switches as well. Now as racks of servers are added to the cluster, both network and server platforms are configured and integrated into the existing cluster.

- **Using scripting to install Linux extensions to PicOS:** A service provider writes a shell script that is downloaded to a switch with ZTP at boot time. The shell script downloads an OpenVPN RPM, installs and configures it. Its last function is to send a message to the NOC with its logs and configuration information. After the switch reboots, managers can securely access the switch from the remote NOC for provisioning.

Summary

Open networking offers the opportunity to deploy your own OS on a bare metal switch as long as you have compatibility with the CPU, U-boot and ASICs. Leveraging a modern Linux distribution will take care of the CPU; however, you have to confirm that your OS works with the U-Boot code on your switch and the ASICs in the switch. Open source initiatives are taking hold that will streamline these processes further. Also best practices from the server side such as ZTP will ensure your network devices can be thought of in a lifecycle just like your server assets.

Pica8, Inc.
Corporate Headquarters

1032 Elwell Court, Suite 105
Palo Alto, California 94303 USA
650-614-5838 | www.pica8.com

Pica8 and PicOS are trademarks of Pica8, Inc.