

Project Crossbow

Network Virtualization and Resource Management

White Paper
May 2009

Abstract

OpenSolaris™ Project Crossbow brings capabilities for network virtualization and network resource management to OpenSolaris 2009.06. It promotes more effective sharing of networking resources, enhances the ability to consolidate workloads on a single server, and allows organizations to meet quality-of-service goals for networking.



Table of Contents

Executive Summary.....	1
Introduction.....	1
Architecture.....	3
Performance.....	6
Virtual Networking Scenarios	6
Summary.....	9
Other Resources.....	9

Executive Summary

The OpenSolaris Crossbow project implements a wide-reaching re-architecture within the OpenSolaris networking stack to bring a variety of advantages:

- Providing a fully virtualizable network environment for more effective sharing of networking resources and increasing the scope for server consolidation projects.
- Providing networking resource management capabilities to allow organizations to meet quality-of-service goals for networking.
- Decreasing latency and increasing throughput, particularly as network load increases.

The combination of open-source software and industry-standard hardware opens a variety of new avenues for OpenSolaris in embedded applications and in consolidation projects, where the value of network virtualization and resource management offers capabilities other general purpose operating systems cannot offer.

Introduction

The Crossbow Project is an ambitious effort to rearchitect OpenSolaris networking to accomplish the three goals listed above. In more detail:

1. Crossbow network virtualization enhances the ability to consolidate server workloads. In the networking equipment market, the term 'network virtualization' tends to have a somewhat limited span, focusing on specific issues like Virtual LANs and aggregation techniques. Network virtualization in the context of this paper is a far more general abstraction — the notion of virtualizing all aspects of a network topology within a server virtualization framework. There are many facets to the Crossbow network virtualization story:

- By virtualizing the concept of the hardware Network Interface Controller (NIC) into Virtual NICs (VNICs), Crossbow promotes more effective sharing of networking resources. The VNIC construct allows dividing a physical NIC port into multiple virtual ones to create kernel-enforced isolated and dedicated network stacks from physical interface to application.
- Crossbow virtualizes a physical LAN switch through the “Etherstub” construct, allowing switching to take place within a virtual environment.
- Virtualization is not only appropriate for dividing a physical NIC, but also for aggregation purposes. Aggregating two physical NICs to a single virtual one and dividing the result into (for example) three VNICs allows better sharing of the network resources and provides redundancy should one of the physical links fail.
- The industry-standard Virtual LAN (VLAN) construct is supported, allowing NICs and/or VNICs to be assigned to a VLAN. In an environment with switches and routers that also support VLANs, this allows end-to-end traffic isolation even though the traffic may be running on a shared physical link.

Additionally, other operating system networking elements can be brought into play, particularly the router and firewall included with the OpenSolaris OS. Taken together, these elements enable building an entire networking topology within one computer system — useful for architecting/prototyping, testing,

and even deployments. There are diagrams later in this paper that illustrate examples.

2. Crossbow network resource management allows organizations to meet quality-of-service goals for networking by specifying guaranteed resource levels as well as resource maximums to system processes. Traditionally, this has meant giving administrators control over CPU and memory resources to ensure that certain applications can get minimum resource levels no matter what other demand there is for those resources. Resource management can also be used to prevent applications from taking too much of CPU or memory resources. Crossbow extends resource management to networking in three ways:

- By allowing specification of the CPU resources assigned to a NIC port or VNIC. This could be used to limit CPU resources or to ensure that enough resources are allocated for high priority, high bandwidth traffic.
- By allowing the specification of bandwidth limits for a NIC port or VNIC. Again, bandwidth can be used to limit the resources or to ensure a minimum level. Bandwidth limits must be assigned intelligently, as the sum of the NIC bandwidths should not exceed the physical bandwidth. I
- By allowing the specification of priorities for types of traffic on a NIC/VNIC basis based on source or destination IP address, MAC address, port, or protocol.

These networking resource management capabilities enable enforceable organizational network sharing policies.

3. Crossbow can increase network throughput by more efficiently scheduling and handling packets. The best performance gains typically come with the latest-generation intelligent NICs with packet classification and multiple receive and transmit ring buffers that Crossbow can manage. There are many aspects of the design of Crossbow that facilitate increased efficiency, but one of the major ones is dealing more efficiently with inbound packets. See the Architecture section below for details.

Virtualization Technologies

Crossbow also enhances the value of Sun's virtualization technologies for server consolidation using OpenSolaris as either the delivery OS or the underlying host for virtual machine guest operating systems.

Operating System Virtualization

SolarisS Containers is a virtualization technology that allows one operating system instance to offer multiple virtual, isolated OS environments. The key advantage of this approach is that while applications see an environment that looks like a dedicated OS, in reality these Container virtual-OS environments are running on one operating system. Solaris Containers, when compared to hypervisor virtualization technologies, can make much more efficient use of system resources because one OS oversees the CPU, memory, and network resource allocation. Containers also have excellent scaling properties because of the extremely small system overhead they place on the operating system. And finally, creation and destruction of Solaris Containers is a light-weight task that facilitates their use in dynamic environments.

Prior to Crossbow, applications running on Solaris Containers could access network interfaces but, if a dedicated network stack for the Container's network interface was desired, a dedicated physical NIC or dedicated VLAN was required. With Crossbow, a Container can be assigned as many VNICs as needed and each will have its own dedicated stack whose bandwidth, priority, and CPU allocation can be managed.

With Solaris Containers and network virtualization, multiple servers (and services) can be consolidated on to one instance of OpenSolaris.

Hypervisor Virtualization

- xVM Hypervisor is an open-source, Xen-based virtualization technology that allows running Windows, Linux, and Solaris guest operating systems on an OpenSolaris hypervisor. This capability will typically be provisioned through SunTM xVM Ops Center, which is scheduled to be released after OpenSolaris 2009.06.
- Logical Domains are a virtualization technology embedded in SPARC[®] systems and based on a hypervisor architecture. Each guest domain has a virtual network device that will be used to access the underlying network interface(s) and to access other domains. Associating networking properties (bandwidth, priorities, CPU resources) to LDOM guests is not supported in OpenSolaris 2009.06.
- VirtualBox is a desktop virtualization technology that allows running a wide variety of x86 operating systems as an application on a variety of hosts, including OpenSolaris. VirtualBox is an attractive virtualization option, particularly for development and testing. Typically, a single server would be set up with multiple guest OS's to mirror a deployment scenario; for example, a three-tier Web/application/database server architecture. The developer or tester would interact with those guests through VNC or RDP screen-sharing sessions or by simply logging in to the guests. If VirtualBox is run on OpenSolaris 2009.06, all the advantages that Crossbow brings to network virtualization and network resource management can be passed to the VNICs used by the Solaris, Linux and Microsoft guest OS's. Using Crossbow virtualization elements like VNICs and virtual switches it is possible to create an entire network topology within the server.

From the preceding, it should be clear the impact the Crossbow architecture has on network virtualization and resource management. We will now turn to details about the architecture to understand more about how key features are delivered and to understand why Crossbow is about more than just virtualization. Crossbow is a redesign that impacts the entire network stack and particularly the data link layer to provide the foundation for the next-generation network stack.

Architecture

The fundamental building blocks of this new architecture are Virtual NICs or VNICs- a construct for dividing a physical NIC into multiple, virtual ones. A VNIC device is accessed, from the applications viewpoint, exactly like a physical NIC. The characteristics of a VNIC — the bandwidth, what CPU resources are assigned to handle it, and priorities — can be dynamically controlled.

Crossbow is designed as a fully parallelized network stack structure. If you think of a physical network link as a road, then Crossbow allows dividing that road into multiple lanes. Each lane represents a flow of packets, and the flows are architected to be independent of each other — no common queues, no common threads, no common locks, no common counters. Tying this architecture to a modern NIC is illustrated in the block diagram below.

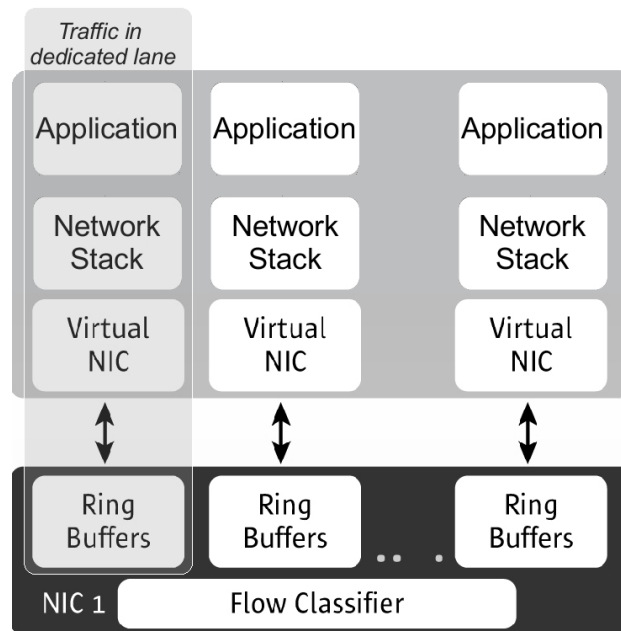


Diagram 1 – Traffic Lanes with Classification by Intelligent NIC

A key element of the design is flow classification and, in the case of diagram 1, the flow classification is done in the NIC. Once the incoming flow has been classified, it enters its own private lane indicated by the shaded area. The NIC has a Transmit (Tx) and Receive (Rx) ring buffer dedicated to each lane. The data is structured as a ring because Rx data that is not transferred into the operating system will eventually get overwritten by new incoming data. Packets are not supposed to be dropped, but we'll see below that if the load is so great that packets must be dropped, it's much better to drop them in the NIC rather than having the OS expend CPU resources to decide to drop them.

A hardware classifier is not a requirement as we will see below, but it does offer the best performance. The classifier can be programmed to classify on a range of OSI Layer 2, Layer 3, or Layer 4 attributes:

- MAC address
- Source or Destination IP address
- Protocol
- Port

Operationally, this means that HTTPS traffic can be handled by one lane, HTTP traffic by another, FTP by a third, and additional or alternate controls can be added based on source or destination address.

In an environment where security is important, hardware classification is generally perceived as a more secure solution. One could create a Solaris Container to house the application, assign a VNIC to that Container, and force all the Container traffic through dedicated hardware classified 'lanes'. From an operating system perspective, whether hardware or software enforced, the isolation in the lanes is preserved but as noted, hardware assist in classification is generally a more appealing solution.

Not every NIC has a flow classifier and even if it did, the number of lanes required by the OS may be more than

can be supplied by the NIC. For that reason, Crossbow also includes a software layer (see Diagram 2 below) for dealing with dumb NICs.

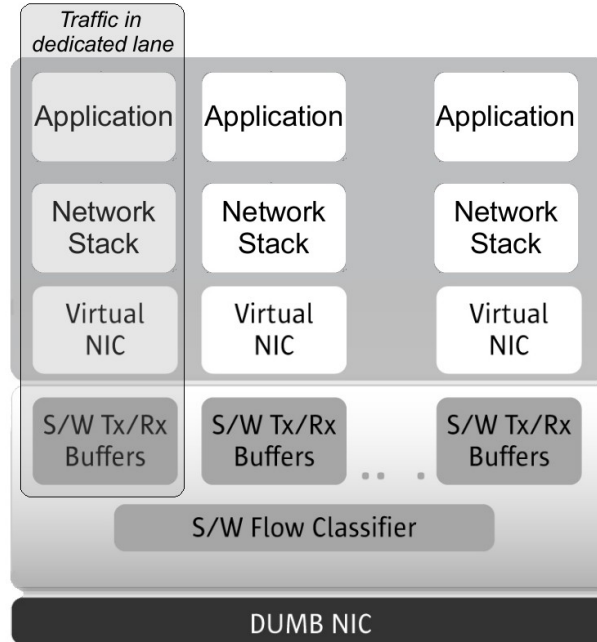


Diagram 2 – Traffic Lanes with Classification by the OS

In this case, all the architecture for supporting VNICs is part of the kernel — a software classifier with dedicated Transmit and Receive rings to create the lanes entirely in software. This same approach allows supporting applications with requirements for say, 20 VNICs on a system with a classifier that only feeds 16 hardware rings. The software enables creating real lanes from the hardware as well as virtual ones that share a real lane.

Crossbow is both network virtualization and network resource management. The ability to specify the characteristics of these lanes is an important element of the design. Crossbow allows managing resources through three mechanisms:

- Setting bandwidth limits
- Setting traffic priorities
- Assigning the number of CPUs to handle the traffic

Bandwidth limits can be used to guarantee minimum resource levels or to prevent a VNIC (or NIC port) from using more than it should (from an operations policy perspective). A physical 1Gb port could be broken into 3 virtual ones, which could then be assigned 100Mb, 100Mb, and 800Mb rates. Assigning a higher bandwidth to the last VNIC reflects the operational importance of the applications using that interface and therefore can be used very effectively to help applications meet Quality of Server (QoS) goals.

These resource management capabilities tie cleanly to the architecture. Here are two examples:

Pushing flow control close to the NIC is important. In traditional flow control implementations, just bringing a packet into an operating system queue from which it will potentially be dropped expends a great deal of the total processing cost of packet handling. If the flow is metered at the NIC, a dropped packet will have

less impact on the system. For NICs that manage their own Tx/Rx rings, dropped packets incur no CPU overhead.

Other QoS implementations tend to be a layer inserted into the network stack — a choke point that doesn't take into account contention for resources elsewhere. With OpenSolaris Resource Management, it is possible to provide an application with the minimum CPU, memory, and networking resources to holistically enforce QoS requirements; just setting bandwidth limitations is not enough.

Network resource controls are not limited to VNICs. They can also be used to manage physical NICs — or more accurately since a NIC may have multiple physical ports, to manage each of those ports by setting bandwidth, CPU, and priorities.

One other element of the architecture is worth highlighting: the way Crossbow manages interrupt handling. In low-utilization mode, packets are handled in the traditional interrupt manner. This is an adequate approach for lower-speed, large-packet traffic. In high-speed, high-load networks, interrupts can have a negative effect on overall system throughput. There are various techniques to mitigate this, but fundamentally an interrupt per packet is simply not efficient on a busy network. Crossbow automatically switches from interrupt mode to polling mode when the packet arrival rate exceeds a threshold. Polling has a key advantage for busy networks: one poll by the driver can potentially return a chain of many packets in one operation, which is far more efficient than one packet per interrupt.

Performance

For performance data, see the networking section on opensolaris.com/networking.

Virtual Networking Scenarios

The value of virtual networking as described above should be apparent. The combination of virtual NICs and the ability to manage those resources makes an excellent match to virtual server technologies. For example, the ability to carve up a 10Gb or 1Gb physical interface into smaller 'lanes' and assign those to an xVM Hypervisor or Solaris Container is compelling because the hypervisor (in the case of xVM Server) or the Solaris Operating System (in the case of Solaris Containers) determines bandwidth, priorities, and/or CPU resources assigned to the virtual entity.

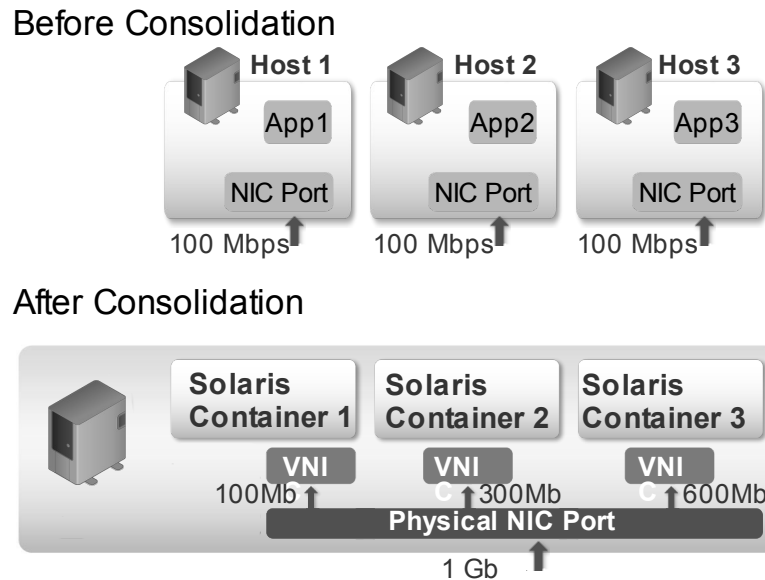


Diagram 3 - Simple Virtualization Scenario

In this case, three systems and their applications have been consolidated on one OpenSolaris system using three Solaris Containers. The consolidated server was moved to a 1 Gigabit per second (Gbps) network, so the host bandwidth limits prior to consolidation are enforced after consolidation even though it is on a new, higher-speed network. Many different bandwidth scenarios are possible. If the consolidation system had enough ports, the network topology of the environment before consolidation could be exactly duplicated after consolidation. Or alternatively, different bandwidths could be assigned based on the importance of the application's bandwidth requirements. The main point is that Crossbow gives an administrator much more control of the network resources while still enjoying the advantages of server consolidation.

A more sophisticated virtualization project is pictured below:

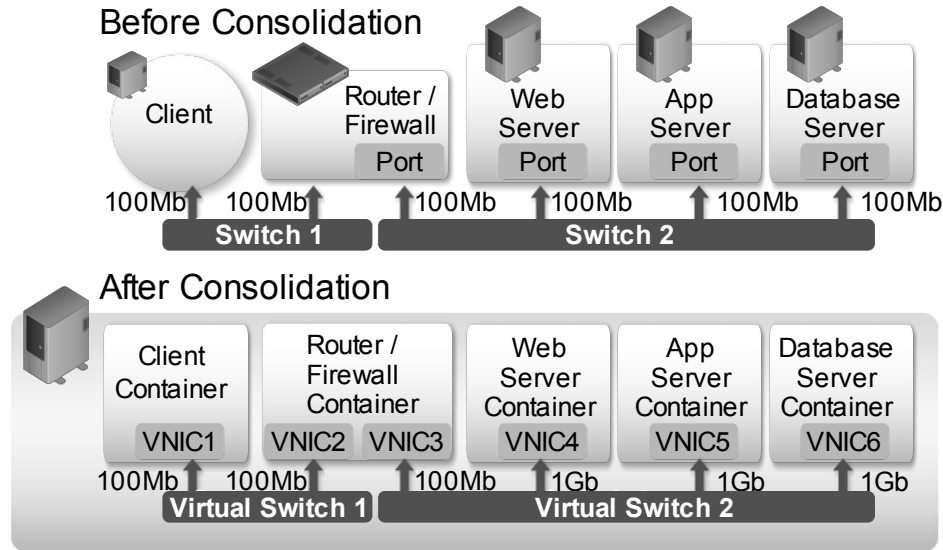


Diagram 4 – Sophisticated Virtualization Scenario

In this classic, three-tier architecture case, one host runs a Web server, another the application server, and the third runs the database. One client is shown. A developer might consolidate this environment on one system to work on some element of the interaction between these three systems. The testing organization might also use this consolidation to simplify the hardware requirements for testing the entire application environment. Note that we've added two new virtualization elements for this example — a switch and a router. The virtual switch makes it possible for the Solaris Containers to communicate directly between each other. In the first simple example of this section, the Containers could not communicate directly within the host; either a virtual switch or an external, physical switch would need to be added. The virtual router is the open-source Quagga project, included with OpenSolaris 2009.06. OpenSolaris also includes the IPFilter firewall, another useful element that could be used in server consolidations.

As in the first example, if we bring in resource management, many different solutions are possible. In this example, suppose we are modeling a new architecture where the three servers are communicating via a 1Gb link. Or we could just as easily set the bandwidths to match the preconsolidation environment. Or if we were going to deploy this scenario in production (and assuming we added some external network connections), there may be no reason to put any bandwidth limitations on the inter-server communication. Since those packets never go on a physical connection, they move at rates comparable to memory-speed rates. It may seem a stretch to deploy a three-tier architecture on one system, but with the incredible increase in multiple CPU cores per physical CPU chip, it may be worth considering. Certainly in that environment, the bandwidth between the servers is not likely to be the bottleneck.

A final example of the power of Project Crossbow is its use as part of a defense against Denial of Service (DoS) attacks. As pointed out above, Crossbow allows creating communication lanes from the physical network port on the NIC to the application. When the NIC hardware supports both flow classification and receiver ring buffers, the Crossbow architecture allows much more efficient mitigation of the effects of a DoS attack. Bandwidth limits will pace the kernel in polling the VNIC (or NIC) for packets. When packet rates are very high as in a DoS attack, packets will inevitably be dropped in the receive buffer. Not involving the kernel in dropping packets has a

significant impact on decreasing the load on the CPU, which in turn diminishes the impact on other services of the DOS attack. And the dynamic nature of VNICs means that it would be possible, having identified a DOS attack, to change the characteristics of the classifier to drop packets based on a variety of filtering rules or further limit the bandwidth of the flow.

Summary

Project Crossbow is the next step in the evolution of the Solaris networking stack, bringing bandwidth resource control and virtualization as key elements of the overall design. Crossbow adds the critical resource management element to OpenSolaris so that administrators can now control CPU and memory, as well as networking resources to meet quality-of-service goals. Crossbow also adds a critical piece of the system virtualization story — the ability to virtualize not only servers but their networking topology including firewalls, routers, and switches. And finally, Crossbow offers an architecture that dovetails with current trends in NIC design to extract the best performance from those devices.

Other Resources

See www.OpenSolaris.org/os/projects/crossbow, the OpenSolaris project page for Crossbow.

The www.opensolaris.com/networking site has specific information related to OpenSolaris 2009.06, Project Crossbow and other networking projects. You will also find pointers to additional Crossbow resources.

© Sun Microsystems, Inc. All rights reserved. Sun, Sun Microsystems, the Sun logo, OpenSolaris, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. or its subsidiaries in the United States and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the US and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.