



Announcer You're listening to the Sun Microsystems Podcast Network.

Announcer Welcome to another edition of [Innovating@Sun](#), with your host, Hal Stern. Today's topic, Project Neptune. And now, here's Hal Stern.

Hal Stern

Hello and welcome to Innovating@Sun. I'm your host, Hal Stern, Vice President of Global Systems Engineering at Sun Microsystems. And my guests today are Shimon Muller, Distinguished Engineer in the Systems Group; Ashley Saulsbury, also a Distinguished Engineer in our Systems Group, and Erik Nordmark, Distinguished Engineer in the Solaris Networking Group. And these three caballeros have distinguished engineering are going to talk to us about the Neptune project. So, why don't you start off and tell us a little bit about what you do at Sun.

Shimon Muller:

My name is Shimon Muller. I've been at Sun for almost 19 years, all of this time working on networking. This includes networking product strategy, how it fits into our systems, and also network technology development.

Erik Nordmark:

I'm Erik Nordmark. I've been at Sun for 17 years, working on networking as well. But from the software and protocol side of things, with things like how does Solaris networking stock evolves, and what we need to do to it, as well as how do the protocols evolve. IETF standardization, IP v6, etc. And I'm continuing to do things in this space, with virtualization and other things that come up.

Ashley Saulsbury:

And I'm Ashley Saulsbury. I've been at Sun 11 years, which makes me the youngster in this crowd. I've been working on the whole range of different things in my time at Sun. For the last six years I've been working on Sun's virtualization technologies. And I'm the architect for the forthcoming logical domaining technology, which will be coming out this year.

Hal:

Great. Erik, you said something interesting. You talked about how protocols have evolved. And I think there's always been this sort of view of what TCP/IP can't do, as opposed to let's get the most out of it and figure out what it can do. Why don't you start by telling us a little bit about what the Neptune project is, and essentially how we designed it ground up.

Erik:

Some of the thinking we had a long time ago was, how are we going to deal with faster and faster NICs. Because it seemed like there was a speed bump coming, where it wasn't clear that Moore's Law would help the CPUs keep up with the speed of the network. So for a long time, we're running at 10 Megabit Ethernet, moved up to 100 Megabit, and to a Gigabit. And that sort of moved at a speed where it seemed like the processors could keep up. But, as we saw 10 gig coming, it wasn't clear how we were going to do this. So we started thinking about what are different approaches that you can do this. We had some things that sort of already existed, which is if you

took a machine with lots of 1 gig NICs, you had a similar issue, that you had things coming at you at an aggregate rate that was pretty significant. But, at least they came in from different NICs, different I/O buses, separate interrupts, etc. So we were thinking about, how could we actually evolve this. And some of the things that led to the Fire Engine project that went into Solaris 10 to put some of the groundwork into play for being able to deal with lots of aggregate bandwidth coming in. And then moving on to saying, if everything comes out of one single spigot, one single NIC, that's faster, how do we deal with that from a performance perspective.

Hal:

Why don't you define fast in this context? What's the network balance we're looking at for Neptune?

Erik:

The Neptune NIC has two 10-gig Ethernet ports on it. That can run at 1-gigabit speeds as well. Shimon is actually the one that has better details on the hardware capabilities here, but this is significant. For many years we used to have the rule of thumb that if you're going to deal with a megabit of bandwidth coming at you, you need about a megahertz of CPU speed. Which is very approximate, very rough rule, but it meant that, it means that if you actually apply it to deal with a 1 gigabit of bandwidth coming at you, you need a gigahertz processor. So, we clearly don't have in the industry today 10-gigahertz processors to deal with a 10-gigabit NIC throwing packets at you at that rate.

Hal:

Shimon, where is the secret sauce in the hardware then?

Shimon:

The secret sauce is really - what we try to do here is following the emerging trends of the compute environment. Today's compute paradigm is all about parallelism and multithreading. To some extent, it's nothing new to Sun. We've had a multithreaded operating system, Solaris, for over a decade now. We also built multiprocessor systems for a very long time as well. What has changed is that this paradigm has actually been extended into microprocessor architectures. We all know that Sun is leading the way with CMT technology and Niagara and Rock. What happened here is that the former high-end server were refrigerator-sized, sort of became a server on a chip. The rest of the industry's following in the same direction, but we have a strong lead having orders of magnitudes of threads. The bottom line is this, from a purely networking perspective. This miniaturization of systems introduced some new challenges for us. The old way of doing networking no longer became practical. Erik mentioned earlier on that to achieve parallelism we used to plug in a dozen NICs into a high-end system. And that worked for a certain extent. But when you go to miniaturized system, that no longer works. Obviously, in the Niagara-based system, you can not plug in so many NICs, because the system's just too small. You don't have so many I/O slots. It also becomes economically too expensive to do that. So, this interesting challenge, we took it and tried to make an opportunity out of it. The secret sauce was to provide a technology that takes the fat pipe coming over the network and breaks it up into smaller workloads, that we can spread out to threads. This is really the key part of the innovation of the Neptune technology.

Hal:

Great. So if we look at this then as this notion of applying parallelism at multiple levels - at the hardware level, at the OS level, at the system level - what are able to go do with that? Is it just on throughput, or can we start actually applying some of these resources to new problems like virtualization?

Shimon:

Let me address that. What happens with this kind of technology is - I see this technology is, if you will, a missing

piece of the puzzle for building a completely multithreaded system. I see the unique combination of what we have with Solaris with Crossbow. CMT processors with LDOMs . And the state-of-the-art networking that we have with Neptune, it allows us to built a sort of a killer multithreaded piece of machinery that is uniquely optimized for network facing operations. A big part of this kind of applications is what we call virtualization. It's another important trend in the industry that most of the industry's going in. And one of the killer applications that's often being mentioned is consolidation of systems to achieve high utilization. Today we can virtualize compute cores. We can virtualize memory. We can virtualize storage. Up until now, we could not virtualize an I/O sub-system in general. And networking specifically. Again, Neptune addresses these by providing a fat pipe that can be broken up into smaller pieces and directed into the hardware threads and the processor. Which are tied to threads in the operating system, and it's pretty much across the stack. This is a unique advantage to Sun, because we own the entire stack, all the way from the operating system, from the applications, all the way to the wire. Ashley can probably expand a little bit more on the specific virtualization part with LDOMs, and how Neptune ties into it.

Hal:

Clearly, I/O virtualization is something we hear about as an up-and-coming area of concern as we look at power virtualization of your system-level virtualization. Where does this intersection of Network from a network interface card and Solaris as the OS above it, where is that going to actually build some benefit.

Ashley:

I think this goes all the way back to Erik's initial point about needing a megahertz for a megabit of networking bandwidth. When you get to these kind of high networking speeds, if you're going to take a purely software approach, which is sort of the traditional - I say traditional - the current, let's say, methodology for dishing out virtual network interfaces to multiple operating system instances running on a machine, then you need a hypervisor or an I/O virtual machine in your system. Which is going to handle the NIC and is going to receive all of the packets that come into that NIC, and then start handing them off to each of its different client virtual machines. So, if you've got a 10-gigabit Ethernet interface coming into your box, you need a 10-gigahertz CPU to actually perform that kind of functionality. Or, say ten 1-gigahertz CPUs. That's a lot of power. That's a lot of overhead, just to even handle that basic filtering. One of the huge advantages that Neptune has is an advanced packet filter. That actually removes that overhead. And we can send the traffic straight to each of the different virtual machines from the raw interface as we receive them. So what you actually have now is the advantage to utilize your CPU power, utilize your hardware, and direct the streams that are coming in to each of the virtual machines, each of the different applications that are making use of them. In the real world, it's not often the case that you have two machines talking directly to each other at say at 10 gigabits. The reality is, what you have is hundreds, or even thousands, of individual streams, each talking to separate applications. Each of which has its own context. Each of which has its own storage, etc., that is needed. And so, disambiguating that and handing it to individual threads is kind of the key part to getting a scalable networking solution for the modern world. And that's really one of the advantages that Neptune has, is the hardware filtering it does to actually split that apart very early on in the receive stage, and the transmit stages.

Erik:

You asked a bit, Hal, about are we using this for something other than performance? Well, virtualization at this level of, these packets for this MAC address or IP address, they go to a separate domain, that's one place where we're using this. To sort of do separation. And, as we evolve the Crossbow architecture, which people can participate in out on OpenSolaris.org, we're putting in some more resource controls so you can say, this virtual machine gets this much bandwidth. But also being able to do this, once you have things separated out from the Neptune and going up, you can also apply that to finer grain controls, saying that, well, I want my HTTP traffic, based on port numbers, etc., to only get this much resources. Get a particular set of threads in the hardware. And

my HTTPS traffic can get a larger chunk of that. In terms of the actual architecture, and from the hardware up through the layers of software, that's something that we also are going to use this for in the future.

Shimon:

Also, you asked earlier on how fast is fast. As far as Neptune is concerned, we can do 20 gigabits in each direction, which gives you sort of a 40-gigabit bite today. Which may sound a little bit exotic in today's terms. But it's been done for a multithreaded system of dozens of threads. Going into the future, we're building processors and systems that will go to hundreds and thousands of threads. Also, in the industry, going faster is the name of the game. And as we talk, there are people already working on 40-gigabit Ethernet, 100-gigabit Ethernet. Maybe both in parallel. So, our challenge is sort of to take the foundation that we build with Neptune, continue building on it, and continue scaling it orders of magnitude and try to stay as ahead of everybody else. So Neptune, the way I see it, is just the beginning.

Hal:

We've talked a little bit out Fire Engine. We talked a little bit about Crossbow. Erik, I guess maybe some more detail is in order here. In terms of the kinds of networking optimization, the kinds of stack optimization, that has gone into Solaris 10, so that a heavily threaded processor, a highly parallelizable network stack, a highly threaded operating system, can all work in conjunction with each other, in composition with each other, to go drive network [inaudible]. Can I have a little bit more detail on some of the work that we've done in both Crossbow and Fire Engine.

Erik:

If we go back a bit and look at what did the system look like before Solaris 10, it was a multithreaded system. But it was sort of, at some levels, too fine-grained, the multithreading, and at other levels of the system a bit too coarse. So taking an example of packets coming in to the system, if they came from a single NIC, they would end up piling up on effectively one cue, as the hardware hands them off to the software. And then after that, there would be potentially a larger number of threads that would sort of come and pick up packets to process them up through TCP/IP and up to the applications. But, at some cases, those guys, those threads were actually fight over things. They would actually end up contending with each other. Because there were too many of them. In other cases, there weren't actually enough of them. Because we didn't have enough packets coming in to the stack from the bottom. The path we started on with Fire Engine in Solaris 10 was saying, look, this doesn't actually fit. It's multithreaded, sure, but it's not organized. There's no lane disciplines in the system. So, if we instead say, once we've accepted a packet to process it, or a connection more likely, all the packets for that connection will stay organized by staying on the same CPU. Have the same CPU process them. Try to keep the same thread that processed them through the system. And thereby not having this sort of internal congestion as they fight over each other. And that fits very well with these CMT systems and multiprocessor systems as well, where you have a number of NICs coming in to the system. A number of network interfaces. Generating interrupts to you. And then you have these threads that pick them up and run them to completion. So that's essentially what Solaris 10 accomplished almost two years ago. As we move forward, people looked at, well, what about interrupts in this case? What people have done for quite a while is trying to cope with the large number interrupts coming off a network interface. When, in fact, when you're receiving one packet every minute, it's good to be told that, hey, there's a new packet for you to go look at. When you receive one packet every microsecond, well, that's not necessarily news that there's another packet coming in. Or another thirty packets coming in. Because they come all the time. So the model that has evolved throughout Solaris 10 here is that, let's have the system be more efficient by being able to switch over to polling the NIC when there's a lot of traffic, and going back to using an interrupt when things go quiescent and there's a pause in the traffic stream. So these pieces is the foundation for them taking Crossbow further and saying, how can I actually manage these resources better? One way is tying it

in with hardware like Neptune, where we getting parallelism, lots of threads coming up from the bottom, separate streams of packets. Another one is saying, how could we have better control of this system. Where, what we've already had created in terms of these threads that manage each individual lane, where we have the separate packet, we can now say, we're going to slow down that lane. By having that thread do less work. After it's received some number of packets, it's going to pause, saying, you've exceeded your quote for packets for this class of traffic. Hence, I'm not going to service any more packets for you. So that's the path of evolution that we're on in terms of exploiting this, and unlike traditional "quality of service" mechanisms, this is essentially free. It just having an additional check saying that, I've already received these many packets in the last time quantum. So now I'm going to pause for a little bit. You don't have to add an additional quality of service mechanism to do that.

Hal:

So what you've described now is really systems design at the largest possible granularity. You're looking at how we look at throughput from the point of view of a packet that's on the wire, into something eventually gets [inaudible] up to the application. And where we're going to go apply our resource management, and exercise resource management, up. Great stuff in theory. How's this turning into a product? Where are we going to see the 10-gig Ethernet technology really show up. First, in the way that's usable by our customers.

Shimon:

The first realization of the Neptune technology is in a product that we're announcing today, which is dual 10-gigabit Ethernet card. It's a low-profile PC express card that can plug in pretty much in all our systems. Both in the SPARC space and X86. All operating systems will be supported. Solaris, of course, we spent a lot of time in optimizing performance, and the capabilities of Neptune in it. There's going to be a Linux version of the driver. There's going to be a Windows driver within a couple of months as well. The more important realization of the same technology is - the same Neptune technology has been implemented in the Niagara 2 processor. And when we announce systems using that processor, we're going to see the same capabilities and a lot more. The network interface of the Niagara 2 processor is - the best networking is as good as it gets. So we're having really great hopes, seeing application performance in those kind of systems.

Hal:

Great. And clearly, I have to ask, what's next? Where do you go next with the technology?

Shimon:

I touched on that a little bit earlier on. As I said, the Ethernet technology's going to high speeds, all the way to 100 gigabits per second. We have a great foundation with Neptune. What we are going to work on now is to enhance it. Going faster. Providing additional capabilities. Always when you try to do technology cutting-edge, you always have lessons to learn. We'll learn some of them already. We'll learn more. And as we go, we will concentrate on enhancing the technology. Working very close with Erik and his folks in the Solaris land. With Ashley on the logical domains. And take advantage of the unique situation that we are in, that we're really all the entire stack from the application all the way to the wire. Being a networking guy, I think this is huge.

Hal:

Great. Anything else that you want to add?

Shimon:

One of the advantages that we do have with Neptune is it's - from a networking perspective, when you do a network interface, performance plays a huge role. So, we solved the performance problem, and going line rate at

10 gigabits is, nobody even comes close today. In the networking space, particularly in the Ethernet world, the value of technology is being measured of the cost per gigabit. And when you reach the point of three times cost for ten times performance, the technology becomes deployed pretty much everywhere. We took care of the performance side with Neptune, and the cost of the silicon side. There is some things that are happening in the industry for providing of lowering the cost for the physical layers. Which is a big part of the cost of the 10-gigabit technology today. And one of the major things that really happened is that this technology that had been developed for running 10 gigabit over twisted pair cables. The technology is pretty much done. It has been standardized, and there are multiple vendors working on it. And later this year, there are going to be multiple products being announced. And it's going to propagate into the market. The combination of the cost and the performance that we get out of Neptune will bring us way below this formula of three times the cost for ten times performance. Which is really - it's really very interesting from our perspective. The way I see the 10 gigabit technology is going to be become commoditized very quickly, just like 1 gigabit has happened in the last three to five years.

Ashley:

The other thing worth noting is, taking the Neptune design and actually integrating it on chip for Niagara 2. There's an implicit recognition that networking is probably the most important interface that you have on a modern computer or server system today. And having that interface tied directly into the memory subsystem, tied directly into the CPU subsystem, even on the same chip, to provide low-latency access and direct communication between the threads of the processor and the packet processing of the network interface itself, just emphasizes how important that particular aspect of networking is to a modern computer system. We've started down that road. We're surely going to continue down that road. And just keep pushing the scalability aspects of this technology.

Hal:

So clearly, as things get increasingly commoditized, we're able to once again reshape the boundaries of what we think we can and can not do with network systems. So, Shimon, Erik, and Ashley, thanks for being our guests here on Innovating@Sun. And I'm your host, Hal Stern.

Announcer You've been listening to Innovating@Sun. Join us next time for the latest in innovation from Sun Microsystems. Only on the Sun Microsystems Podcast Network.