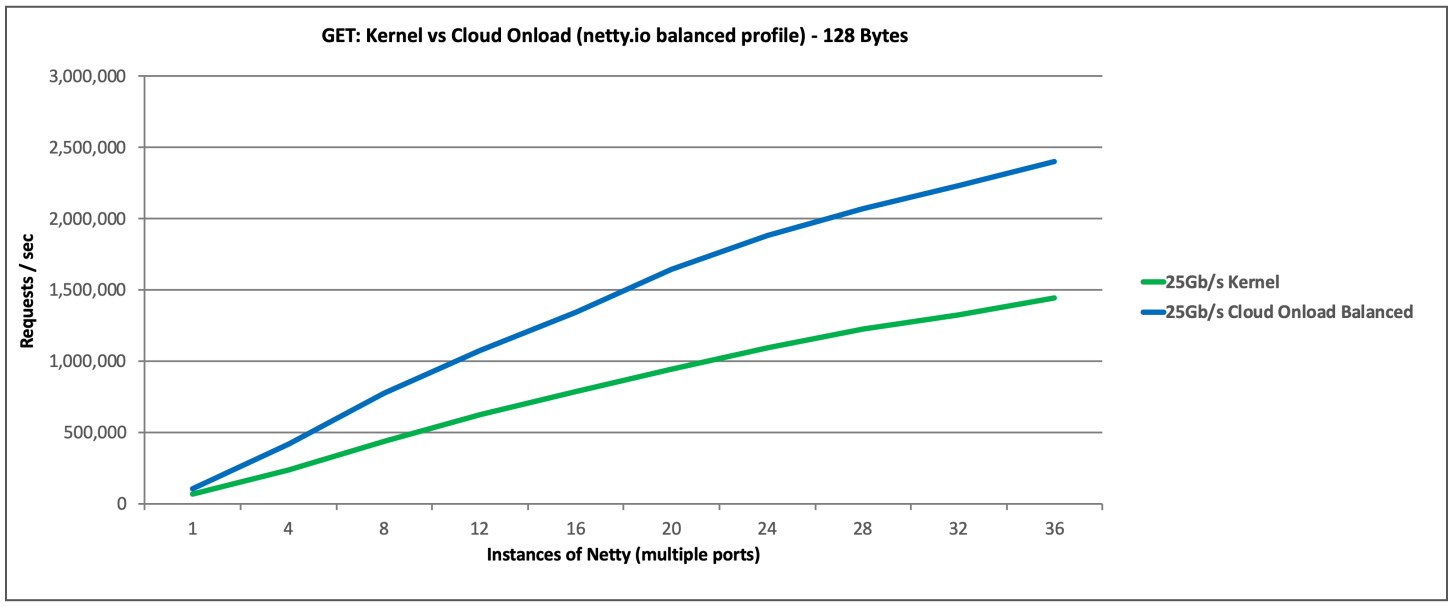
Netty.io Running with Onload® Sees a 70% Performance Gain

SOLARFLARE®
A XILINX COMPANY

Netty

What is Netty.io?

Netty is a Java NIO (non-blocking I/O) based asynchronous event-driven network application framework for rapid development of maintainable high performance protocol servers and clients. It greatly simplifies and streamlines network programming such as TCP and UDP socket server. The design features include a unified API for various transport types – blocking and non-blocking socket; based on flexible and extensible event model which allows clear separation of concerns; a highly customizable thread model – single thread, one or more thread pools such as SEDA; and true connectionless datagram socket support. The performance features include better throughput, lower latency; less resource consumption; and minimized unnecessary memory copy.



Key Observations from Performance Testing

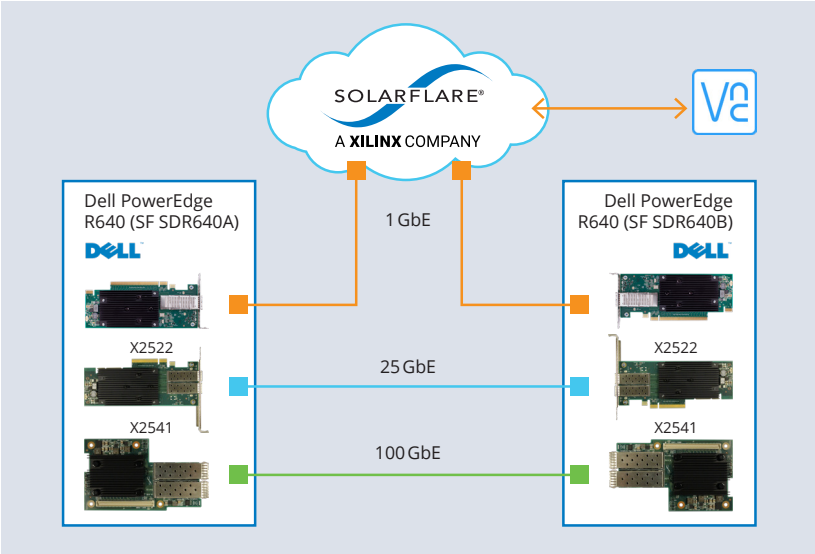
- Solarflare's Onload delivers a 70% performance gain on average for Netty with NIO with persistent TCP connections using GET transactions with 128-byte response payload size when using 25GbE with 40 Netty instances.
- When processing GET requests using 40 or fewer Netty instances Onload on 25GbE outperforms Netty utilizing the kernel communications stack with a 25GbE connection.
- We soon reach the limits of the Netty application or the physical system under test. If we look at 25GbE with Onload we can see that the link reaches a maximum of 2.4 million requests per second.

Why Netty Benefits from Kernel Bypass

Since Netty is network intensive, every request includes network processing overhead. Whenever an application like Netty touches hardware, other than the CPU or memory, and in this case the network, it must make at least one, and sometimes several calls to the operating system kernel. Each request is additional overhead that requires both CPU cycles and processing time. Solarflare's Onload moves the network processing required by Netty from the kernel into Netty's own application space in memory. This single modification improves the performance of Netty by 70% on average as can be seen in the graph.

Description of Test Platforms

For this testing we used two Dell EMC PowerEdge R640 with two dual socket Intel Xeon systems labeled "SFSDR640A" and "SFSDR640B", and two production networks, 25GbE and 100GbE, connected back-to-back. The "SFSDR640A" system was used as the Client and the "SFSDR640B" system was used as the Server. Both systems had two Gold 6148 CPUs clocked at 2.40GHz with 20 cores per processor, 98GB of memory, and two Solarflare network cards: X2522-25G dual port 25GbE and an X2541 single port 100GbE. This enables us to test performance against a range of shipping Solarflare adapters as shown above.



Tuning Configuration

Here are the changes we made to the standard install beyond simply leveraging Onload.

- Enable polling/spinning EF_POLL_USEC to 20 which causes Onload to busy wait for up to 20us before blocking when the application makes a blocking call such as recv() or poll().
- Prevent spinning inside socket calls by enabling the following to 0
- Enable EF_PKT_WAIT_SPIN to 0
- Enable EF_TCP_RECV_SPIN to 0
- Enable EF_TCP_SEND_SPIN to 0
- Enable EF_TCP_CONNECT_SPIN to 0
- Enable EF_TCP_ACCEPT_SPIN to 0
- Enable EF_UDP_RECV_SPIN to 0
- Enable EF_UDP_SEND_SPIN to 0
- Enable EF_UL_EPOLL to 3 and EF_EPOLL_MT_SAFE to 1 which provides the best scalability and speed.
- Enable EF_RXQ_SIZE to 4096
- Enable EF_HIGH_THROUGHPUT_MODE which enables receive event batching to improve transaction rate/efficiency.
- Disable EF_CTPIO and EF_PIO as these reduce CPU efficiency

Observations

Netty relies on the operating system's communications stack to process network I/O requests, but in high core count environments, this stack has become the new bottleneck. Here are some additional points to consider:

- Netty with Onload can service up to 2.4 million Get requests/second, while Netty using the operating system kernel can only handle 1.4 million Get requests/second, a 70% improvement.
- Therefore, Netty with Onload can potentially reduce your capex by 70%. In simple terms every two Netty servers leveraging Onload can service the same number of requests as three Netty servers using the standard Linux kernel.
- Conversely, adding Onload and a Solarflare 25GbE or 100GbE NIC to existing servers will provide additional headroom for growth or unanticipated business peaks as shown above.

For More Testing Details

Check out **Onload Netty.io Cookbook** for the exact installation and testing process along with the specific tuning and tweaking commands executed above.

The above benefit statements are the result of benchmarking designed to focus on the value of optimizing networking through Onload kernel bypass. Real world use cases are not the same as benchmarks and as such the role that networking plays may vary, so your overall measurable benefits may be different.

For more information please visit:

solarflare.com

SOLARFLARE®
A XILINX COMPANY

Contact Us:

US +1 949 581 6830
UK +44 (0) 1223 477171
HK +852 2624 8868
Email: sales@solarflare.com

SF-122198-CD Issue 3
Netty.io Benchmark Results 112219