SolarflareWhitePaper

Reducing Nginx Web Server Costs with Solarflare OpenOnload and Flareon 10/40GbE Server I/O Adapters

Executive Summary

The Solarflare Flareon^{TM} SFN7002F 10GbE Server I/O Adapter with Solarflare OpenOnload® can minimize web server costs by up to 33% and deliver up to a 120% increase in Nginx application performance over the Intel Ethernet Converged Network Adapter X710. Similarly, the Solarflare Flareon SFN7142Q 40GbE Adapter is able to increase 40G connection rates 3-fold and linearly scale Nginx performance, achieving nearly 40Gps line rate. These cost savings and performance increases are accomplished by using the OpenOnload application acceleration user-level networking stack that enables the Nginx application to be exceptionally CPU efficient and scale almost linearly with increasing number of CPU cores.

Introduction: Scope and Purpose

Many Web, Cloud, and CDN (content delivery network) customers are looking both to minimize capital expenditures (CapEx) and increase web server performance by upgrading and optimizing components of their Web platform including compute, storage, network, and the Nginx software stack itself. This paper shows how Solarflare OpenOnload and Flareon 10GbE and 40GbE server I/O adapters can reduce server costs while still improving Nginx performance. The key is the Solarflare OpenOnload application acceleration middleware that improves the network stack performance and reduces CPU overheads when running Nginx.

Nginx[1] is a high-performance HTTP server. Nginx (or its commercial version Nginx Plus) is deployed in many Web and content delivery networks (CDNs) for servers to support short-lived and long-lived HTTP connections. This paper investigates the specific use case where a web server handles many concurrencies of short-lived requests for moderately small payloads, such as web e-commerce, modelling the case of a typical website with sub-10 Kbyte request sizes. The benchmark also compared performance as a function of the web server configuration.

OpenOnload

Solarflare OpenOnload is a Linux-based, open source, high-performance application accelerator middleware. It is an implementation of TCP and UDP over IP which is dynamically linked into the address space of user-mode applications, and granted direct (but safe) access to the network adapter hardware. The result is that data can be transmitted to and received from the network directly by the application, without involvement of the operating system, using a technique called “kernel bypass.”

Transitioning into and out of the kernel from a user-space application is a relatively expensive operation: the equivalent of hundreds or thousands of instructions. With conventional networking such a transition is required every time the application sends and receives data.

[1] *http://wiki.nginx.org/Main Retrieved 2015-03-17*

sales@solarflare.com

US 1.949.581.6830 x2930

UK +44 (0)1223 477171

HK +852 2624-8868

www.solarflare.com

Kernel bypass avoids disruptive events such as system calls, context switches and interrupts and so increases the efficiency with which a processor can execute application code. This also directly reduces the host processing overhead, typically by a factor of two, leaving more CPU time available for application processing. The effect is most pronounced for applications which are network intensive including:

- Web-caching, load-balancing and Memcached applications
- HTTP web serving, web e-commerce servers
- Content Delivery Networks (CDNs)
- High-bandwidth video-streaming
- HPC (High Performance Computing)
- Market-data and trading applications

Benchmark Setup

Server

- Dell R630 server with two Intel Xeon E5-2620 v3 CPUs (2.40 GHz, 6 cores w/HT)
- 64 GB DDR4 SD-RAM at 1867 MHz
- Red Hat Enterprise Linux 7.0, kernel version: 3.10.0-123.el7.x86_64
- Nginx web server 1.7.7 (patched for `SO_REUSEPORT`[2])
- Network adapters:
    - Solarflare SFN7002F and SFN7142Q, ultra-low latency firmware
    - Intel X710
- Network adapter driver versions:
    - Solarflare Onload: OpenOnload-201502-u1/ v4.4.1.1017
    - Intel X710: i40e 1.2.37

Clients

- Eight Dell R210 servers each with one Intel Xeon E3-1230 CPU (3.20 GHz, 4 cores, no HT)
- 16 GB DDR3 SD-RAM at 1333 MHz
- ApacheBench (`ab`)[3] version 2.3 (patched: see Appendix I)
- Network adapters: Solarflare SFN6122F, Driver version OpenOnload-201502-u1
- The kernel networking stack handled the traffic on the client machines.

In each test, four `ab` instances were spawned on each client host, configured to attempt an equal number of concurrent connections varying according to the specific test. The sum of these attempted connections across all `ab` instances across all the client hosts is denoted in this paper by "concurrency". The `ab` instances were run without keep-alive, meaning that each `ab http` request established a new TCP connection.

Connectivity

A high port density top of the rack 10GbE switch was used to connect the clients and the server. This switch has forty-eight 10GbE SFP+ ports and four 40GbE QSFP ports. The clients

[2] *http://forum.nginx.org/read.php?29,241470 Retrieved 2015-03-18.*
[3] *http://httpd.apache.org/docs/2.2/programs/ab.html Retrieved 2015-03-17.*

and server were each connected to the switch on only one of the two ports of the dual-port network adapters. The network interface on the server was configured with a single IP address, whereas sixteen IP addresses were configured on each client interface to avoid TCP port-space exhaustion.

Methodology

The benchmark's goal was to maximize the request rate that Nginx can handle as a function of the server configuration, i.e., the number of CPU cores used was artificially restricted in order to demonstrate the comparative efficiency of OpenOnload versus the Linux kernel and thus provide a metric for measuring the impact on server CapEx. Of significance is the relative performance gain as a function of the number of cores. Note that while the results are indicative of performance gains, absolute gains will be dependent on actual workloads. Because OpenOnload presents a standard BSD sockets API, Nginx required no modification.

The number of Nginx processes (workers) was varied up to the maximum number of physical cores in the system. These processes were single-threaded. They each bound a single listening socket to the same port with the `SO_REUSEPORT` socket option set, so that the requests were distributed amongst the server processes. The content served was static and 10,000 bytes in length and was stored in a RAM-backed file system. Interrupts and Nginx processes were then optimally affinitized to logical cores for the given configuration. The specific optimization settings depended on the specific adapter and whether OpenOnload was used.

Two features of OpenOnload particularly relevant to the benchmarking are `SO_REUSEPORT` and *socket caching*. The `SO_REUSEPORT` socket option allows multiple TCP listening sockets to bind to the same IP address and port, and to distribute incoming connection requests between these sockets. *Socket caching* is a feature that reduces the processing overheads of establishing a TCP connection. OpenOnload-201405 and Linux kernel 3.9 added support for `SO_REUSEPORT`, while support for socket caching was added to OpenOnload-201502.

Benchmarking Parameters

In order to model a realistic web server scenario, it was necessary to run the `ab` clients with a fair level of concurrency. If the concurrency level was set too high, the server load increased beyond its capacity and the aim of the benchmark to assess processing efficiency became confounded by other factors, e.g. TCP connection retries, etc. It was decided to run the tests at a concurrency of 2048. This figure was reached by running preliminary experiments at a variety of concurrencies and selecting the maximum number of concurrent connections where the time to service a request at the 99th percentile was below 1000 ms for all of the adapters under test.

Configuration With OpenOnload

For this configuration, the Nginx processes were accelerated with OpenOnload configured in *spinning* mode. This means that the application context performs most tasks that would otherwise have been done in the interrupt context. As a consequence, OpenOnload will generate very little interrupt load, eliminating the need to dedicate (logical) CPU cores to interrupt handling. Therefore, for benchmarking, logical cores were assigned on the basis of using NIC

local processor logical cores first and then using the NIC remote processor cores. Interrupt threads were allocated using NIC local processor cores.

Configuration Without OpenOnload

When OpenOnload was not used, no common configuration was found that obtained the best performance on all adapters in terms of requests per second. The configuration that delivered best performance for the Solarflare SFN7002F server adapter is where the Nginx process instances fit into a single processor, with both interrupts and application simply pinned to the same hyper thread on the NIC's local processor. For the Intel X710, the configuration that delivered best performance was where each core was dedicated exclusively to either interrupt handling or application processing.

Results

Benchmark experiments were conducted on 10GbE Intel adapters and Solarflare 10GbE and 40GbE adapters. **Figure 1** shows the performance results of the SFN7002F 10GbE adapter with and without OpenOnload versus the Intel X710 with kernel driver. To show the impact of flow steering on the Intel X710 adapter, benchmark results are shown with Flow Director enabled.

10Gbps Connections

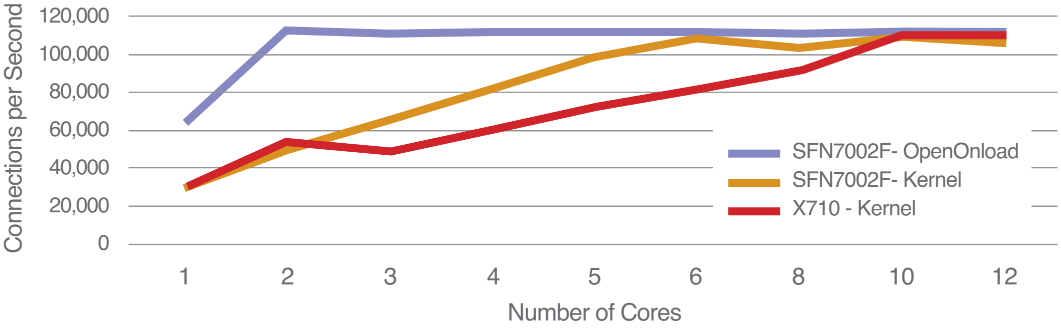


Figure 1. Solarflare vs. Intel Nginx Performance: Cores and Connections per Second.

Figure 1 plots Nginx performance versus the allocated number of cores. Performance is represented as request rate, in this case, connections per second. The figure shows that as the number of cores is increased (i.e. additional Nginx instances are run) the request rate increases until reaching a ceiling of 111.6K connections per second.

The results show that the Solarflare SFN7002F with OpenOnload delivers a maximum 120% increase in Nginx performance over the Intel X710 with Flow Director. With Flow Director off, OpenOnload yields a 128% boost over the Intel X710. When we look at how many CPU cores it took to saturate a 10GbE link, OpenOnload needed two cores. In contrast, the Intel X710 adapter needed 10 cores. OpenOnload is therefore found to be five times more efficient than the Intel X710 in its use of CPU resources. When compared with the Solarflare SFN7002F with its kernel driver, OpenOnload was found to be three times more efficient.

From a web server cost perspective, this would equate to a capital expenditure difference of purchasing a server with a processor having a minimum number of cores, e.g., 2 or 4 cores, as

compared to a processor with 10 cores. A comparison of the server under test costs, i.e., the Dell PowerEdge R630 server with the 6 core Intel Xeon E5-2620 v3 2.4GHz CPU against the Dell R630 with a 12 core Intel Xeon E5-2680 v3 2,5GHz CPU, results in a cost reduction of $1200. As shown in **Table 1**, a comparison of a Dell PowerEdge R320 web server with a 10 core processor option versus a 4 core processor option shows a 33% server CapEx reduction.

Dell PowerEdge R320 Server with 64GB memory

Processor option	Price*	Reduction
E5-2470 v2 2.4GHz w/10 cores	$3111	---
E5-2407 v2 2.4GHz w/4 cores	$2069	33.5%

* *From the Dell website 2015-11-04.*

Table 1. Web server 4 processor cores vs. 10 processor core price comparison.

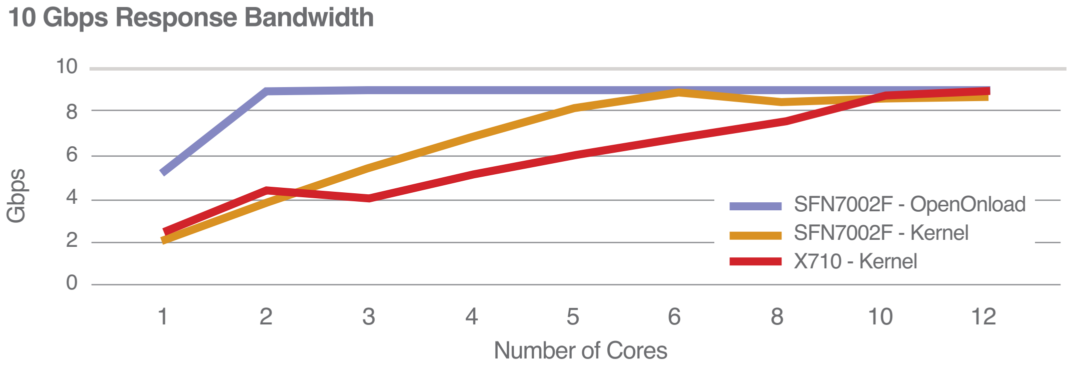


Figure 2. Solarflare vs. Intel Nginx Performance: Cores and Response Bandwidth (Data Rate).

Beyond 10 Gbps: Scaling with OpenOnload and Solarflare Flareon Ultra 40GbE Adapters

The results in **Figure 2** demonstrate that Nginx with OpenOnload can saturate a single 10 Gbps link with the use of very few CPU cores. In order to demonstrate that link bandwidth is truly the bottleneck, an experiment was conducted with a Solarflare Flareon® Ultra SFN7142Q dual-port 40 Gbps adapter. As shown in **Figures 3** and **4**, OpenOnload and the SFN7142Q adapter scales with the number of cores and is only limited by the bandwidth of the link which is reached at 10 cores. At 10 cores, OpenOnload provides a 3-fold Nginx performance increase over the Linux kernel driver.

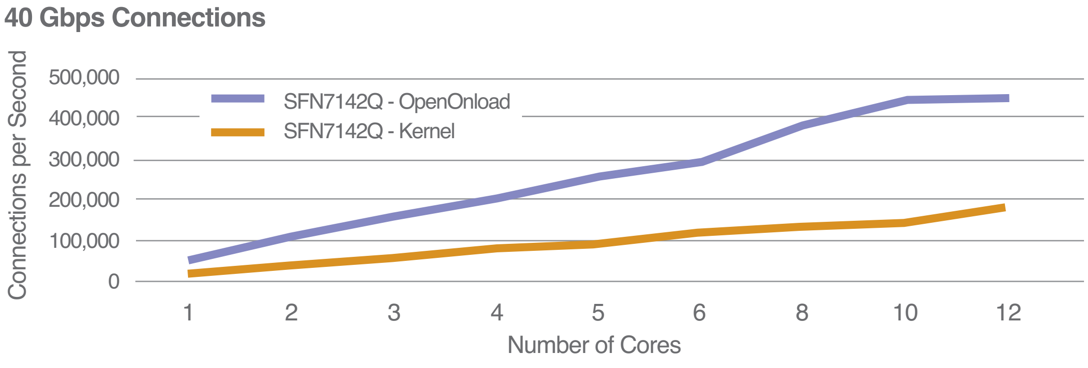
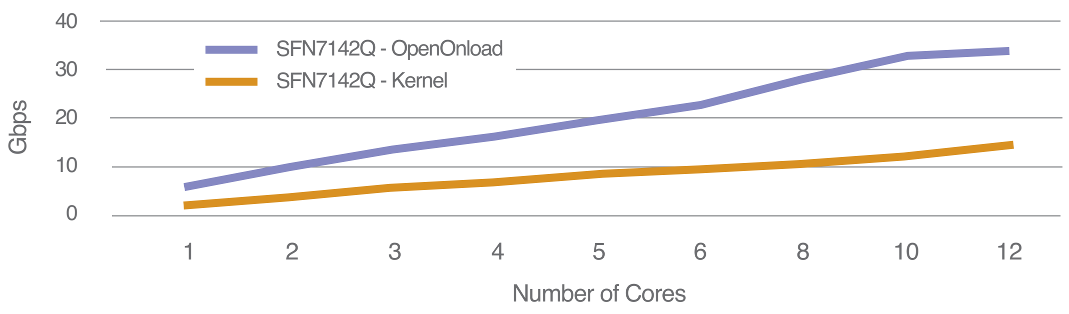


Figure 3. 40GbE Nginx Performance: Cores and Connection Rate.

SOLARFLARE®



40 Gbps Response Bandwidth

Figure 4. 10GbE Nginx Performance: Cores and Response Bandwidth (Data Rate).

Conclusions

With OpenOnload, an industry standard web server running Nginx and 10GbE with Solarflare OpenOnload and the Flareon SFN7002F 10GbE server I/O adapter can enable a web server CapEx cost reduction of up to 33% and achieve up to 120% higher increase in Nginx performance than the Intel X710. At 40GbE, the web server performance with Solarflare OpenOnload and the SFN7142Q 40GbE server I/O adapter scales almost linearly with CPU cores to the limit of the link bandwidth.

Bottom line: Web server deployments can realize substantial CapEx savings while at the same time improving Nginx performance with Solarflare OpenOnload and 10GbE and 40GbE server I/O adapters.

Appendix I. ApacheBench modification

To overcome some of ApacheBench limitations, its sources have been modified to achieve a number of features:

- Allow synchronizing multiple `ab` instances.
- Enable use of array of ip addresses per `ab` instance.
- Do not stop despite connection failures.
- Preset expected response size.
- Provide number of successful requests.

SOLARFLARE®

SF-115413-CD Issue 1
Reducing_NX_Costs_WP_111615

sales@solarflare.com

US 1.949.581.6830 x2930

UK +44 (0)1223 477171

HK +852 2624-8868

www.solarflare.com