Citrix NetScaler nCore technology

Taking Web application delivery performance to new levels

Introduction

The growth in Web application development and deployment shows no signs of abating. In fact, it continues to accelerate. This is not surprising when one examines the benefits delivered by Web-based applications such as traditional ERP, CRM and Web 2.0. The explosion in Web applications, however, comes at a cost: performance. The myriad of new development methods and frameworks (e.g., Ajax, SOA, Comet and Rails), combined with the expanding use of classic Web services and JSON are placing ever greater performance demands on the network, and consuming additional server cycles and network bandwidth at an unforgiving pace.

Keeping pace with these applications means taking load balancers or application delivery controllers (ADCs) to a higher level of performance and capacity. Gaining this necessary performance, however, is no longer as simple as adopting the latest Intel® or AMD processors. Incremental gains in the performance of individual processors have tapered off. All future performance increases will come from architectural improvements, namely *multi-core processors* that, unlike older processors, embed multiple processing elements into the same processor package. For example, where an older Pentium® 4 could only process one instruction at a time, a 4-core Intel® Core™ 2 Quad can do four instructions in parallel.

For applications and software in general, performance does not magically scale simply by using these newer multi-core processors. In other words, four cores running at 3.33 GHz is not equivalent to one processor running at 13.32 GHz.

Breaking the single core barrier

Achieving high performance in a multi-core/multi-processor environment means overcoming a number of technical hurdles such as effective distribution of load, sharing of resources between processors, and making sure that critical functions that cannot be parallelized are managed correctly. The common element in all of these hurdles is synchronization.

Synchronization is the brake of high performance network processing—it slows everything at critical junctures to eliminate potential conflicts and make sure that the program's result is correct. The irony of synchronization's goal of ensuring correctness is that in designing synchronization, complexity increases. With increased complexity comes a greater likelihood of mistakes.

The impact of synchronization is shown when examining multi-threaded applications. In multi-threaded applications, tasks are broken down into multiple small programs (threads) that run in parallel. Initial implementations of multi-threaded applications can yield significant performance improvements. The performance curve, however, quickly flattens as application complexity grows and additional threads are added. The performance drop stems from the fact that the more software components interact with one another, the more synchronization is required, to the point where the benefits of multi-threading are outweighed by the time needed to synchronize all the multiple threads.

The obvious solution is to eliminate synchronization, which is easier said than done. Eliminating synchronization requires a complete software re-architecture, significant up-front planning and a deep understanding of the interaction between hardware, software and systems.

Citrix NetScaler nCore technology

Citrix® NetScaler® nCore™ technology is a high performance, parallel-processing architecture that efficiently leverages multi-core technology to scale to meet the requirements of the most demanding Web applications. Building a system that effectively utilizes multi-core processors requires that thought be given to two specific issues: (1) How does the system keep all of the cores busy, all of the time? (2) How does the system eliminate the need for inter-processor synchronization?

The two are interdependent. Removing the need for synchronization between processors allows each processor to focus on driving the best performance possible. To understand how this is accomplished, one must first grasp the defining properties of the existing NetScaler packet engine.

The NetScaler packet engine

The purpose of the Citrix NetScaler packet engine is to pull packets off the network, perform a multitude of TCP/IP processing, acceleration and optimization tasks, and enforce security policies. When the packet engine is finished handling one group of packets, it puts the response back onto the network and grabs new packets to process.

The efficiency of the NetScaler packet engine enables packets to be processed in microseconds. The NetScaler ultra-low latency packet engine optimizes Web application delivery and user experience. By using a carefully designed series of stages for processing packets, the packet engine eliminates the need for synchronization. This allows a single packet engine running on a single core to support multi-gigabit throughput performance levels.

The impact of the NetScaler packet engine's efficiency is best appreciated when contrasted with common Linux-based products. The typical Linux TCP stack must queue and de-queue a single packet when traversing between multiple layers. Each layer requires extensive synchronization overhead before an application even receives the data. Additional switching layers and ASICs used in some Linux-based products amplify induced latency due to shuffling of the packet between layers.

As the NetScaler platform evolved, the benefits of non-synchronization have served as the differentiating factor that makes it possible to scale the packet engine over multiple processor cores with extraordinary efficiency.

The share nothing design

The common challenge in scaling performance is sharing. Whenever data needs to be shared by multiple components, performance quickly becomes a problem. However, when nothing is shared, several benefits emerge:

1. No overhead in synchronization
2. Significantly reduced complexity that results in improved stability
3. Inherent ability to better cope with failure because one component does not impact another component

By leveraging the key properties of the NetScaler packet engine, the nCore architecture was developed as a *share nothing* design.

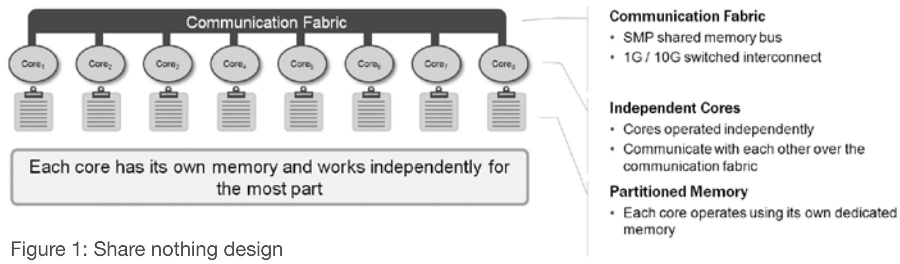

Figure 1: Share nothing design

How share nothing works

The nCore share nothing architecture operates with a series of packet engines, each with a completely self-contained implementation of all NetScaler features for application optimization and acceleration running in a private memory space. Because each packet engine can perform all possible functions, it is able to take a Web application transaction all the way from start to finish without assistance from other packet engines.

With no inter-engine feature dependencies, synchronization becomes a non-issue. Packet engines never wait for one another due to resource contention. This enables NetScaler to achieve extremely high throughput with very low latency.

When a packet engine runs on a CPU core, it is in one of two states: 1) looking for packets to process or 2) actually processing packets. Each packet engine looks for packets in hardware-based packet queues. In keeping with the share nothing philosophy, every packet engine has its own queue, which is not shared with other cores. Since the hardware implementation is fully symmetric, egress packets are sent down private per-processor queues as well.

Because each packet engine is capable of saturating a complete core, a 1:1 mapping is created between cores and packet engines to ensure every available cycle the hardware offers can be put to use. This improves linear performance as additional cores are added to the system, and increases the platform's future scalability.

Approaches to distributing flows across packet queues

There are multiple approaches to distributing network traffic across cores. Three popular approaches are:

1. Functional parallelism
2. Mapping network interfaces to processors
3. Distributing individual flows

Functional parallelism in an ADC is typically achieved by separation of tasks across cores. As a result, specific cores are tasked with specific functions. For example, core 1 might be dedicated to managing network traffic, core 2 to processing TCP/IP , core 3 to layer 7 (e.g., HTTP) processing, etc. On the surface, this is appealing as it isolates spikes in one feature from another. However, in the real world, this type of architecture leads to poor utilization. For example, if one application requires more layer 7 processing but no SSL processing, it is not possible to assign unused CPU resources from SSL to layer 7.

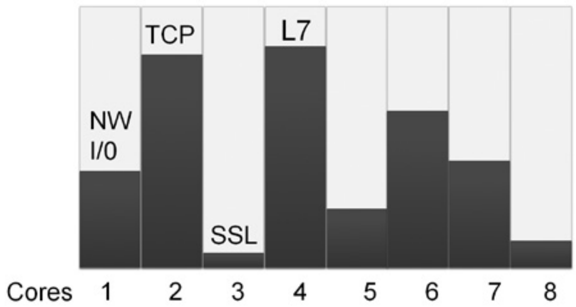

Figure 2: Functional parallelism

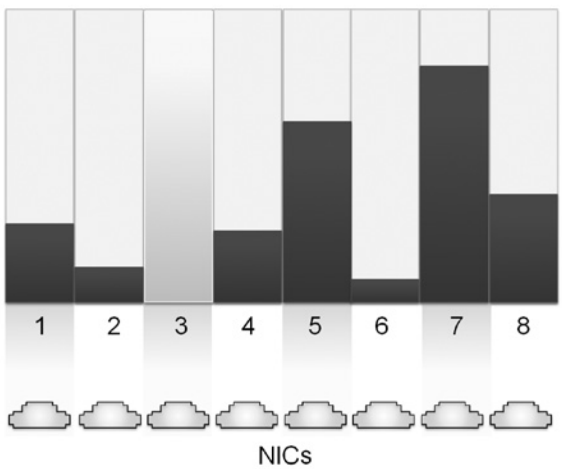

Figure 3: Mapping network interfaces to processors

Mapping network interfaces to processors is an approach that hails from legacy networking where physical ports are hardwired to specific hardware. The modern version, alternatively, maps IP addresses to specific cores. Like functional parallelism, however, this leads to uneven load distribution where a single application can saturate the capacity of a core and not benefit from other idle cores in the system.

After evaluating functional parallelism and mapping of network interfaces to processors, the approach chosen for nCore technology was the distribution of individual flows. This approach is the most granular load sharing mechanism that best leverages the fact that a single packet engine is capable of performing every feature in the product. The result is a truly even distribution of traffic load that ensures that no single application or feature can run out of capacity if there are other cores with available capacity.

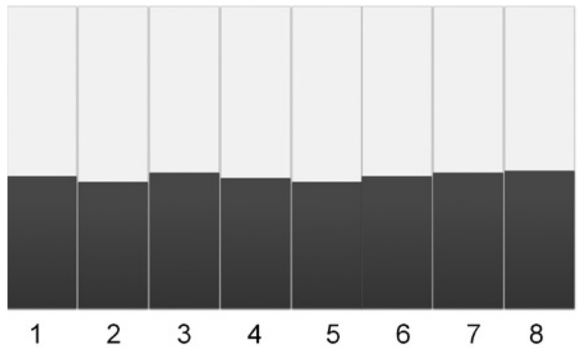

Figure 4: Distribution of individual flows

To perform the distribution of individual flows at extremely high speeds, a symmetric flow distributor is used. This special piece of hardware can take packets that arrive on a single network interface and make a fast decision as to which packet engine ingress queue should receive a given piece of network traffic. Because decision making is based on TCP/IP header information, the distributor ensures that packets for a given flow are always sent to the same packet engine.

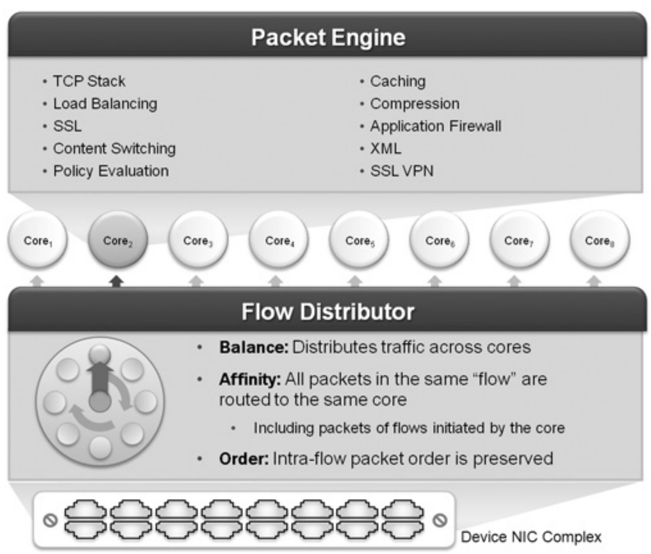

Figure 5: Ingress queues to cores

The algorithm used by the flow distributor guarantees an equal distribution across all packet engines, thereby ensuring that all packet engines support the system's full workload. In the nCore architecture, precious processor resources are never lost to synchronization overhead or unbalanced distribution of network traffic.

nCore management

Historically, ADC resource management has been challenging. When processing power is scarce, it is desirable to devote all available capacity to the packet engine. This leaves a thin margin in which to support necessary management functions such as providing SNMP communications and allowing management console access across one or more interfaces. NetScaler always reserves capacity for management features, allowing the administrator to effectively manage the system even under heavy traffic load.

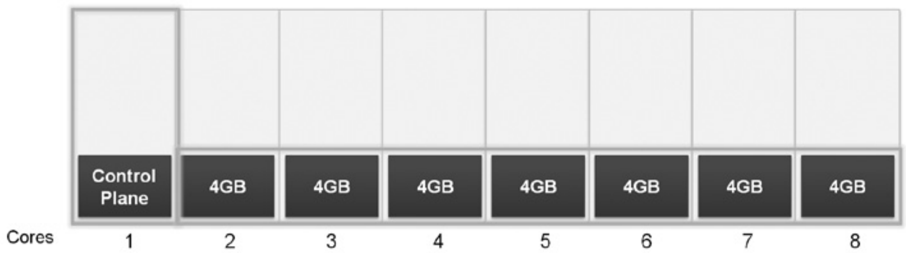

Figure 6: Management has a dedicated core with room to grow

Because nCore architecture changes the nature of processing capacity in a fundamental way, what was once scarce is now plentiful. Therefore, the architectural decision was made to dedicate an entire core and its corresponding private memory space to management functions.

This provides a new level of flexibility in the management tier that leaves room for further innovation and feature development. Equally important, this ensures that the administrator has a working management interface even under severe traffic scenarios.

Why nCore matters

The performance and scalability benefits enabled by nCore technology have significance for both current and future Web application delivery requirements. nCore technology provides:

- Better performance for Web 2.0 and rich Internet applications
- Improved ability to handle large traffic spikes
- Expanded capacity to support more users and more applications
- An all-in-one platform for Web application delivery requirements: L4-7 load balancing, caching, GSLB, compression, SSL VPN, SSL offload, application security, performance monitoring and more

For complex layer 7 workloads that tend to be more CPU intensive, nCore technology provides up to a sixfold improvement. Applications needing to support many concurrent users will benefit from a sevenfold improvement in concurrent connections.

Summary

Citrix NetScaler nCore technology is a high performance, parallel-processing architecture that efficiently leverages multi-core technology to scale to meet the requirements of the most demanding Web applications. The nCore share nothing design goes a step further by removing all synchronization overhead so future improvements in multi-core technology continue to scale linearly.

The linear scalability of the nCore architecture provides more than extraordinary performance—it creates opportunities for innovative solutions to meet tomorrow's Web application delivery challenges. With ample memory and processor capacity, new developments in Web technologies can be optimized, improved and scaled out without revisiting fundamental architectural decisions. The future of Citrix NetScaler nCore holds many more benefits in continued performance improvement and new features enabled by the resulting capacity.

Worldwide Headquarters
Citrix Systems, Inc.
851 West Cypress Creek Road
Fort Lauderdale, FL 33309, USA
T +1 800 393 1888
T +1 954 267 3000

www.citrix.com

Americas
Citrix Silicon Valley
4988 Great America Parkway
Santa Clara, CA 95054, USA
T +1 408 790 8000

Europe
Citrix Systems International GmbH
Rheinweg 9
8200 Schaffhausen, Switzerland
T +41 52 635 7700

Asia Pacific
Citrix Systems Hong Kong Ltd.
Suite 3201, 32nd Floor
One International Finance Centre
1 Harbour View Street
Central, Hong Kong
T +852 2100 5000

Citrix Online Division
6500 Hollister Avenue
Goleta, CA 93117, USA
T +1 805 690 6400

0509/PDF