

W H I T E P A P E R

VIRTUAL MATRIX ARCHITECTURE
Scaling Web Services for Performance and Capacity

- **UNDERSTANDING THE NEW INTERNET**
- **FUNCTIONS OF A WEB SWITCH**
- **DISSECTING WEB SWITCH DESIGN**
- **COMPARING DIFFERENT WEB SWITCH ARCHITECTURE**
- **THE VIRTUAL MATRIX ARCHITECTURE**
- **CONCLUSIONS**

Alteon WebSystems, Inc.

50 Great Oaks Boulevard

San Jose, California 95119

408-360-5500

408-360-5501 fax

http://www.alteonwebsystems.com

W H I T E P A P E R

Web infrastructure equipment is fundamentally different from traditional networking devices like switches and routers. While networking devices focus on switching individual frames and packets at ultra-high speeds, Web switches focus on tracking and processing Web sessions. The changing nature of Web applications, protocols and content delivery mechanisms mandates a radically different approach to Web switch architectures.

DESIGNING WEB SWITCHES FOR THE NEW INTERNET

Web switches must offer high performance, rich feature integration, and session integrity. They must fit into co-located points of presence where rack space is a premium and must be flexible enough to work in a variety of configurations as Internet architectures change. They must be able to "flex" to handle new traffic patterns and have the excess capacity to handle flash crowds and unexpected traffic loads. And they must do all of this in a secure, scalable manner.

Designers of Web switches are constantly trying to balance value pricing, feature richness, and good performance. Alteon's new Virtual Matrix Architecture offers all three, but to understand why this approach is superior we need to first look at some traditional Web switch architectures and their limitations.

UNDERSTANDING THE NEW INTERNET

The Internet is in a constant state of change. More and more users are joining the global network. At the same time, the things people want to do online are changing, from information gathering to shopping, entertainment, and conducting business-to-business transactions. To respond to the increased load and these fresh demands, e-business innovators are deploying new network architectures and new classes of applications. Understanding the new Internet model is essential in order to decide what the next generation of Web infrastructure equipment must look like.

This white paper examines shifts in Internet usage, and the resulting demands placed on data center infrastructure equipment. We consider various design choices and trade-offs in building Web switches, and reveal an innovative new approach—called a Virtual Matrix Architecture—that delivers fast, scalable Web traffic management along with investment protection and robust functions.

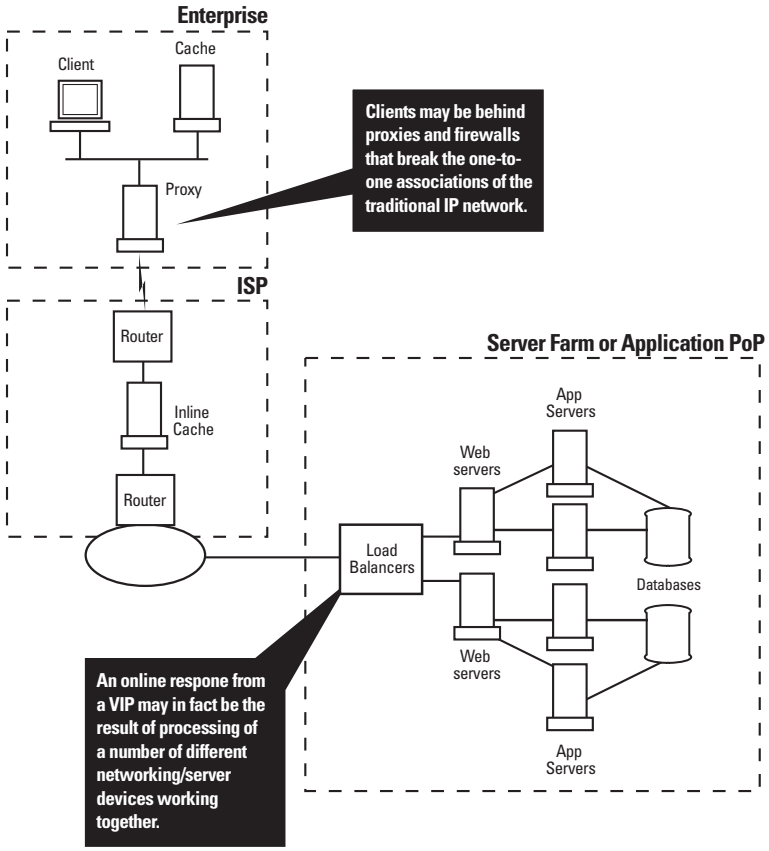
How the Internet has Changed

The old model of a single server is gone. Many servers handle traffic from a single, virtual IP address (VIP) representing an entire server farm. This allows a Web service to continue working even when a server in the farm fails. It also enables processing capacity for a Web service to grow smoothly through the addition of more servers. The composition of Web server farms is changing as well. Where once a Web server handled mostly static content, today groups of Web servers, applications servers, high-performance relational databases and multimedia servers collaborate in supporting sophisticated Web applications.

At the same time, Web clients can no longer be identified by their unique IP addresses as firewalls and proxies may alter source IP addresses in outbound connections. Consequently, today's Internet looks far more complex, as shown in Figure 1.

© 2000, Alteon WebSystems, Inc.

FIGURE 1
A modern Internet architecture



FUNCTIONS OF A WEB SWITCH

With the virtualization of Internet clients and servers, the ability to direct traffic for each Web transaction to the correct server – the highest performing, most available server with the correct content to service the particular client's request - has become the fundamental role of a Web switch. To manage explosive traffic growth, the Web switch is also being tasked with the responsibility of differentiating traffic by classes of users, applications and content types in order to provide the appropriate level of service for each. In addition, being a front-end to business-critical servers, the Web switch is a natural place to protect server resources from intrusion and attacks.

Offering these functions on high performance, manageable, and scalable networking platforms, Web switches are becoming the new anchors for next generation Web data center infrastructures.

Behind the Scenes

Unlike Layer 2 and 3 switches that are optimized for forwarding independent data packets based on well-defined MAC and IP protocol fields, Web switches are designed to perform session-oriented traffic management where the definition of a session varies with different applications. To manage application sessions effectively, the Web switch must examine traffic up to the Layers 4 to 7 protocol fields. Four key requirements drive Web switch designs:

© 2000, Alteon WebSystems, Inc.

Session Classification by Application and Content

A session is a set of related packets that forms a transaction between a Web user and his application. Front-ending Web servers, the Web switch must be capable of processing and forwarding traffic as a server sees it – by application sessions.

Classifying traffic by session is frequently more complex than simply examining the Layer 4 (TCP or UDP) port number in each packet. For example, HTTP version 1.1 allows multiple Web transactions to be transported over a single TCP connection. To forward each transaction to the best server with the appropriate content, the Web switch must examine the Universal Resource Locator (URL) within each HTTP request. Without content intelligent Web switching (or content switching), each server would have to mirror all content for the entire Web site, a sub-optimal use of server resources.

Working with session content is much more demanding than examining TCP/IP protocol headers because:

- Content is non-deterministic. Content identifiers such as URLs and cookies can be of varying lengths and can occur at an unpredictable location in a request. Scanning through an entire request for a specific string is far more processor intensive than looking at a known location in a request for a specific number of bytes.

- To parsing content requests the Web switch temporarily terminates the TCP connection from a client. In other words, the Web switch must first pretend that it is the server, ask the client what it wants, examine the request, and then open a connection to an appropriate server. While this is happening, the Web switch must temporarily buffer the request, which uses up system memory. This temporary termination is called a delayed binding.

- With delayed binding, two independent TCP connections span the Web session – one from the client to the Web switch; the second from the Web switch to the selected server. The Web switch must modify the TCP header, including performing TCP sequence number translation and re-checksum, on every packet that travels between the client and the server, for the duration of the session. This function, known as TCP connection splicing, heavily tasks a Web switch, particularly when the switch must process thousands of these sessions simultaneously.

Stateful and Persistent Connection Support

To ensure that all packets within a session are forwarded to the same server, a Web switch must maintain session states on every active session. Depending on session duration, the switch may need to track tens to hundreds of thousands of active connections, requiring large amounts of memory.

Furthermore, online applications such as shopping carts or search engines often require persistent treatment. This means a client must constantly talk to the same real server for the duration of the session, which may span multiple TCP connections. If a client-server association is not persistent, it may result in broken shopping carts and disgruntled users. Even if an application won't break when a visitor is sent to different servers during the course of a session, there are other reasons for persistent sessions. Servers store recently accessed information in memory. Retrieving information from local memory is thousands of times faster than retrieving it from a back-end database or hard drive. Sending a user to the same server for several consecutive transactions takes advantage of server cache memory, improving server efficiency and performance.

© 2000, Alteon WebSystems, Inc.

In the old IP model, persistence was easy to achieve since a unique source IP address can be bound to a unique destination IP address. But as we have seen, proxies, firewalls and VIP-addressed server farms have invalidated the simple one-to-one model.

As a result, we need more unique information to associate a client and a server with absolute certainty. In today's Internet, this is achieved through information embedded in session content—such as Secure Sockets Layer (SSL) session identifiers, cookies, and URLs. Hence, accurate persistence support is dependent on the switch's ability to provide high performance, content-based, traffic classification.

Bandwidth Management

The Internet is a shared resource among businesses and hobbyists. It is critical for an e-business to ensure its fair share of Internet bandwidth. A business Web site that shares the same hosting facility with an Internet movie site will want to prevent the shared router links from being clogged up by large video downloads. This requirement gave rise to Quality of Service (QoS) technologies that are permeating Internet backbone devices. As an aggregation point for multiple servers connecting to the Internet, a Web switch can enforce QoS at the content source. Strict enforcement requires the ability to meter, log, and control bandwidth usage among different Web sites, hosting customers, applications, and content types – capabilities known as bandwidth management.

To effectively support bandwidth management without slowing down performance, a Web switch can have one of two things. It must either incorporate custom hardware that can handle hundreds to thousands of flows with varying subscription rules, or the switch should be equipped with parallel processors that can keep pace with processing server output bandwidth management for up to hundreds of servers.

Security Protection

In addition to offering granular access and rate control support based on a wide variety of traffic attributes, a Web switch needs to be security hardened to protect valuable data center resources from electronic intrusion and attacks. While firewalls and edge routers have traditionally provided this function, their performance suffers upon intensive filter processing. The Web switch, a next generation traffic manager, can incorporate powerful filtering hardware or processors to offload the access control function from much more expensive edge routers and firewalls, while providing much higher filtering throughput.

DISSECTING WEB SWITCH DESIGN

To fulfill the requirements described above, a Web switch performs session setup, parses traffic for application or content identifiers, applies server selection algorithms, monitors the health and performance of servers, meters and controls server bandwidth usage, processes traffic filters and network topology updates, and so on. These functions are part of the control plane on the switch. The control plane of a Web switch is certainly more complex and less defined than that on a Layer 2/3 switch. Not only are some of these functions extremely CPU intensive, they are exercised much more frequently than the control functions on a Layer 2/3 switch and therefore require more powerful processor technologies.

The "simple" tasks on the Web switch, including Layer 2 and 3 forwarding, network and port address translation, TCP connection splicing, and so on, occur frequently. Similar to the forwarding plane on Layer 2/3 switches, the forwarding plane of a Web switch can be accelerated through hardware-assists.

© 2000, Alteon WebSystems, Inc.

Interaction between Control and Forwarding Planes

When a new frame arrives at a switch port, the forwarding plane decides if it is part of a new or an existing session by looking in a table of state information. If it does not find a state telling it how to handle the connection, it forwards the new session to the control plane for scheduling, parsing, and so on. Figure 2 shows this process.

When a flow arrives on a switch port and it has already been analyzed and scheduled, its state information is available to the forwarding plane. The forwarding plane consults this information and sends the packet on its way according to a previously set decision on the necessary packet transformations. This process does not involve the control plane and can be done very efficiently. Figure 3 shows the flow of traffic in this case.

Even though existing session processing can bypass the control plane, the control plane functions on a Web switch are complex and processing intensive, and must be implemented on powerful processing engines.

FIGURE 2
How control and forwarding planes handle new connections

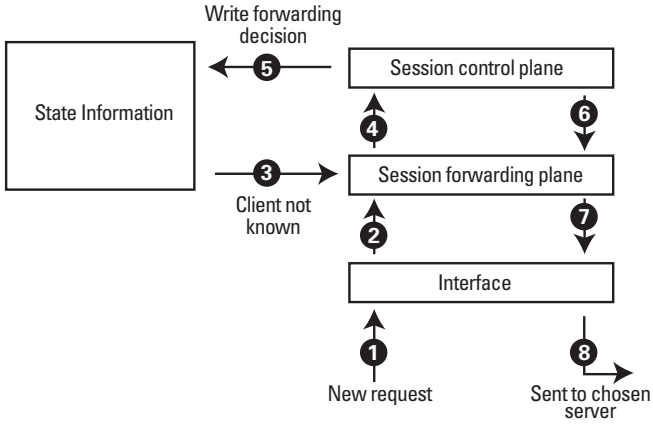
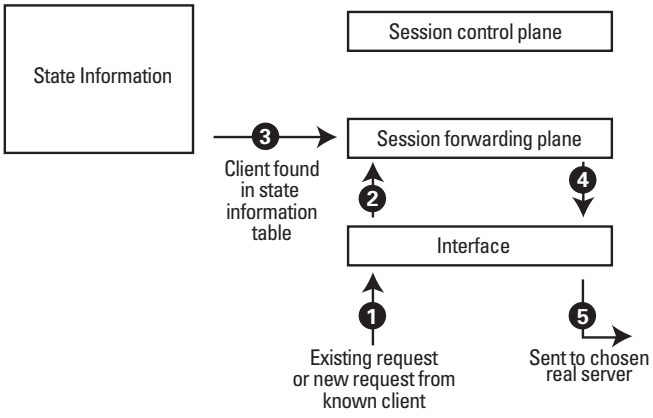


FIGURE 3
How control and forwarding planes handle existing connections



WEB SWITCH ARCHITECTURE

There were three different approaches to Web switch design: centralized CPU systems, distributed processing systems and a two-tier hybrid. Each amounts to a different combination of forwarding, control, and state sharing and has its own strengths and weaknesses.

A fourth—Alteon's Virtual Matrix Architecture—is a radically new approach to Web switch design that is purpose-built to combine the best attributes of all current designs.

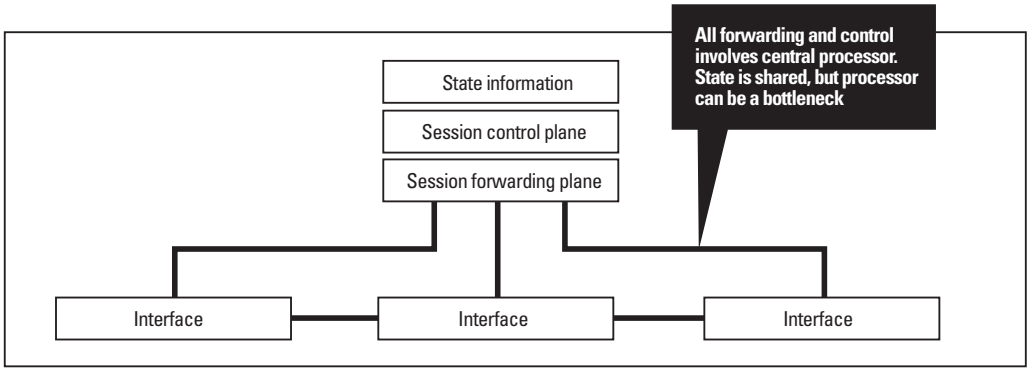
Centralized Systems

A centralized model (shown in figure 4) puts both control and forwarding in a single, central processor. This model is typical of Layer 2/3 switches that have later added session processing capabilities in software. (Note: while the forwarding plane for Layer 2/3 packets may be distributed at the port level, packets that require session processing must be touched by the centralized processor, which performs functions such as network address translation or TCP connection splicing, before they can be forwarded.)

© 2000, Alteon WebSystems, Inc.

This is a flexible model, particularly for topologies where traffic enters from a single port (such as a WAN link), since all processing power and memory can be brought to bear on such traffic. However, when session traffic is heavy or when traffic ingresses from multiple ports, all session processing—even the relatively simple forwarding plane work—must pass through the central processor which easily becomes a bottleneck. This model clearly lacks scalability and is only suited for sites with low traffic expectations and simple traffic management requirements.

FIGURE 4
Functional view of a centralized Web switch



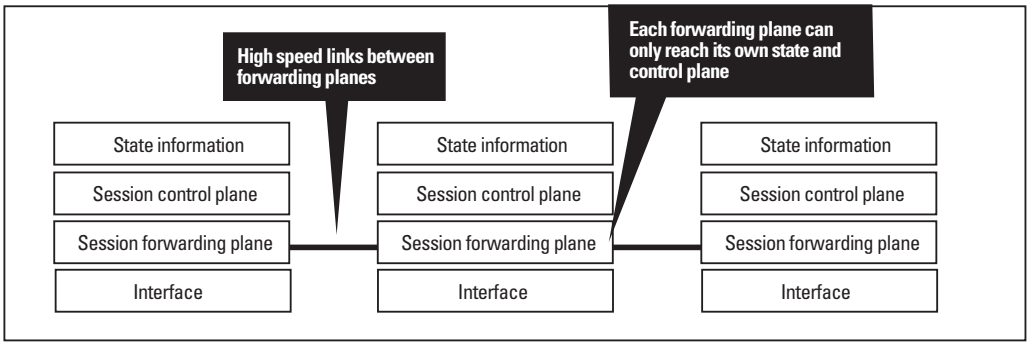
A centralized system has only one forwarding/control processor in the entire system. This means all sate, processing and memory is brought to bear on any particular port, but there is no parallelism or division of labor.

As traffic volumes increase exponentially and Web switches incorporate more sophisticated functions, this architecture is increasingly dated.

Distributed Per-port Processing

At the other end of the Web switch design spectrum is the per-port distributed model shown in figure 5. In this architecture, each port has an associated control and forwarding plane, but a given port cannot access resources on other ports when more memory or processing horsepower is needed.

FIGURE 5
Functional view of a distributed per-port Web switch



Distributed model has fast handling but each port can only use its associated control and forwarding planes

© 2000, Alteon WebSystems, Inc.

This model is extremely fast, since all functions are local to each port, and delivers excellent performance for both forwarding and control plane decisions.

Since the "distance" between control and forwarding planes is virtually nil - each port has its own control and forwarding plane functions – this architecture is optimized for speed and low latency. It is optimal for topologies where session traffic ingresses from a large number of ports. On the other hand, because processors and memory on all ports are independent, the model has some limitations:

- The cost of RAM on each port is high, since each port must, in theory, have enough memory to handle all traffic coming into the system. Each port can only access its local memory and does not take advantage of the unused memory on ports that are not receiving session traffic.

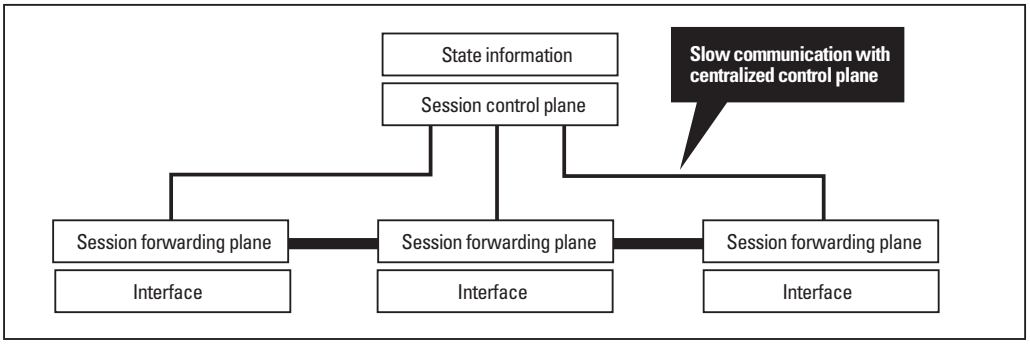
- Memory is not shared meaning that, in a distributed architecture, a port cannot use information on another port to make a decision. If traffic is initially coming into a particular port on a switch, and a topology change causes subsequent traffic to arrive on a different port, there is no way for the adjoining processors to share session state information and handle the existing sessions.

Two-tier Hybrid

An early compromise between centralization and distribution was the two-tier hybrid. Figure 6 shows this model. This architecture consolidates all control plane processing in a centralized processor while giving each port its own

FIGURE 6

Functional view of a 2-tier hybrid switch



Two-tier hybrid shares a fast link between forwarding planes but slower link to control plan can become a bottleneck

forwarding plane. This meant that the device could adjust to new traffic topologies and share control plane information while still performing forwarding functions on a per-port basis without consulting a central processor.

Some implementations of this architecture actually split the session forwarding plane where Layer 4 forwarding, which involves simple network and TCP port address translation, is distributed to the ports and Layer 7 forwarding, which involves TCP connection splicing, is processed by the centralized control processor.

© 2000, Alteon WebSystems, Inc.

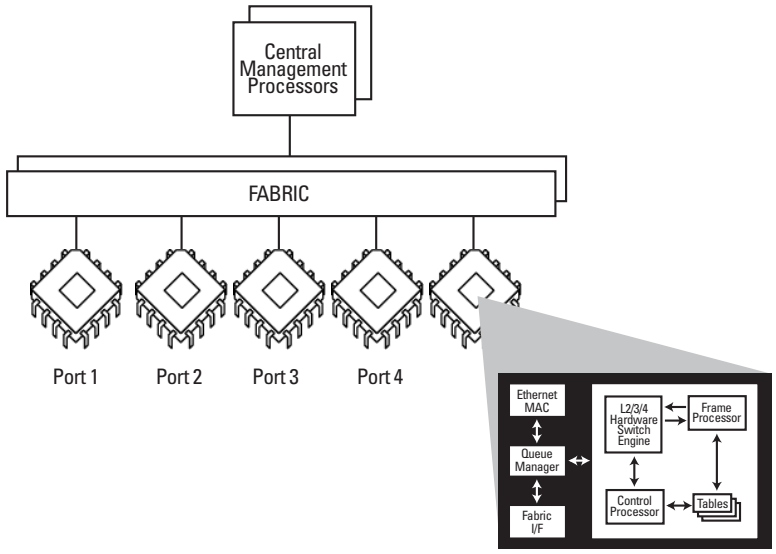
The efficiency of this architecture will depend on -the number of forwarding ports the centralized control plane will be servicing simultaneously and the power of the control-processing engine. In addition, this model generates more intra-switch communications that either exert more load on the switch fabric or require a separate, out-of-band link between the forwarding and the control planes. Depending on session traffic volume and control processing complexity, the control processor and the communications link to the centralized control plane can be potential bottlenecks.

To evaluate hybrid-switching architectures, one needs to know the "distance" from the forwarding plane to the control plane. Distance is a function of the architectural proximity between the two (such as shared memory, for example) and the speed of the medium over which they communicate (such as the bus rate). One should also evaluate the horsepower of the control processor while considering the expected traffic volume and the complexity of the processing functions needed.

INTRODUCING ALTEON'S VIRTUAL MATRIX ARCHITECTURE (VMA)

Alteon's new Virtual Matrix Architecture (VMA) dramatically changes the Web switch architecture by offering the resource aggregation and flexibility of centralized models and the performance of distributed approaches without the inherent performance limitations and lack of parallelism that hampers two-tiered hybrid models.

FIGURE 7
Alteon Web switch architecture



VMA is built on a Web switch design that integrates a large number of network processors directly into a high performance switch fabric. This design enables software control processing to occur directly inline with the hardware-assisted forwarding engine on each port. Figure 7 shows a diagram of the Alteon Web switch architecture.

Each switch port is equipped with an Alteon purpose-built, Application Specific Integrated Circuit (ASIC), called the WebIC, that contains a hardware-integrated switching engine and two RISC processors that are optimized for network control functions. With control and forwarding planes integrated into the same high speed ASIC, the

© 2000, Alteon WebSystems, Inc.

distance between the two planes is virtually nil. With a WebIC on every switch port, control and forwarding functions can execute in parallel on all ports, providing ultra-high packet and session switching performance.

The new WebOS VMA software further optimizes the Alteon switch architecture by enabling the network processors on ALL ports to execute control tasks simultaneously regardless of how many ports session traffic traverses. VMA software creates a virtual matrix of memory and processor resources across the switch that can be used to process traffic from any port at any time.

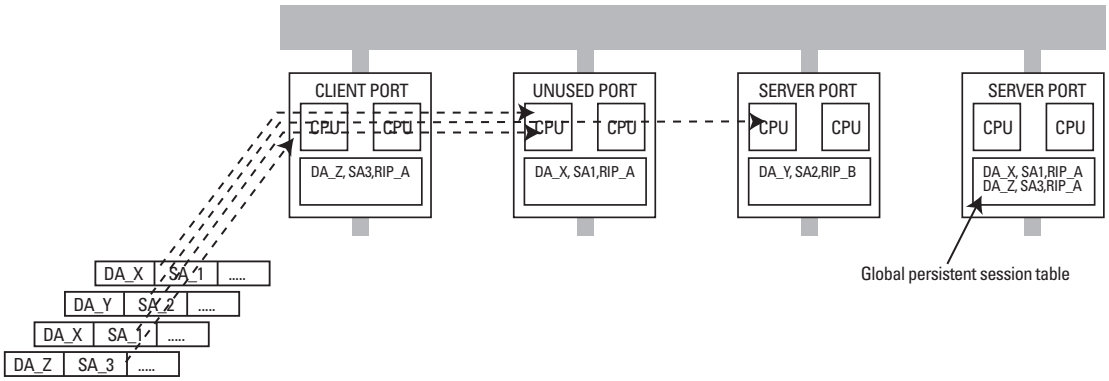
With no changes to hardware and backward-compatibility with installed Alteon 180 and ACEdirector Web switches, VMA delivers investment protection for IT buyers as their capacity requirements grow and their Web architectures change. By integrating higher-level control functions into ASICs and providing a rich set of session classification techniques, Alteon Web switches ensure that they can handle emerging applications seamlessly.

How VMA Works

VMA is a fast, rich, flexible architecture that makes efficient use of entire system's capacity while providing the parallel performance of distributed processing. In this architecture, all processors share load but no single processor must see all traffic. When a packet reaches the VMA-enabled switch, the ingress port applies an algorithm to the source IP address that uniquely selects one of the per-port processors on the system as a designated processor (The designated processor is actually composed of two physical RISC processors integrated into the WebIC, but for simplicity we'll consider the processor pair as a single unit). The designated processor selection algorithm ensures that traffic is evenly shared across all processors on the switch. The algorithm is also deterministic – all packets from the same source IP address will always be handed to the same designated processor.

Once the ingress port hands the designated processor the received packet, the designated processor examines its local session table and makes a forwarding decision which may include parsing content, selecting a server, performing network address translation or TCP connection splicing on the packet, metering bandwidth usage, and so on.

FIGURE 8
Functional view of Alteon Virtual Matrix Architecture



© 2000, Alteon WebSystems, Inc.

With an efficient designated port selection algorithm and high speed intra-switch communications, the added latency of moving a packet from the ingress port to the designated processor is minimal – about 50 microseconds for a 64-byte packet.

When the return traffic travels from server to client, the algorithm is applied to the destination IP address of the packet by the server port, resulting in the same designated processor being selected. In this way, the designated processor sees session traffic in both directions and state information can be kept at the designated processor's local memory where it can be accessed more quickly.

While each processor is handling a roughly equal subset of all traffic going through the device, therefore using memory optimally, certain information must be available to each processor. This includes packet filtering and content parsing rules and common information used for every forwarding decision. For example, each processor keeps the access control list entries for all ingress ports, allowing it to filter traffic differently for each ingress port. The designated port only handles session processing and L3 switching. Lower-layer processing such as L2 switching is handled directly by the processor on the ingress port.

Handling Persistent Connections in the VMA Model

VMA has been extended to handle persistent connections that may contain different source IP addresses (Refer to the proxies and firewall discussion in the How the Internet has Changed section.). One of the port processors on the switch (typically the processor on the switch uplink port) stores a global state table for all persistent sessions. We call this the persistence processor. Following is the operational sequence for supporting cookie-based persistence:

1. The designated port sees a new session request and terminates the connection with a delayed bind.
2. When a user first visits the site, the browser has no cookie for the site; hence the designated port is free to select any server based on the configured load balancing metric.
3. The designated processor sets up a connection to the server and performs TCP connection splicing between the client and server.
4. When the server sends a cookie back to the client, the designated port performs a hash on the cookie.
5. The designated processor tells the persistent processor the hash value and the server to which it maps.

Later on, the client returns to the site to obtain another web object. This time, the user's browser has a cookie for the site.

If the browser's request comes from the same source IP address as it did previously, then the designated processor already knows the cookie-to-server binding and forwards the request to the correct server. If, on the other hand, the browser's IP address has been altered by a proxy, there is a chance that the request will arrive at a different designated processor that has yet to see the cookie. VMA handles the situation as follows:

1. The designated processor sees a cookie but has no record of the cookie's hash value in its memory.
2. The designated processor forwards the client request to the persistence processor, where the previously chosen designated processor has stored a hash of the cookie and its mapping to the server.

© 2000, Alteon WebSystems, Inc.

3. The persistence processor looks in its state table and finds the appropriate server association. It forwards the client request to the chosen server and educates the designated processor on the server association.
4. When the server's response returns, it reaches the designated processor. This time, however, the processor knows how to deal with the query.

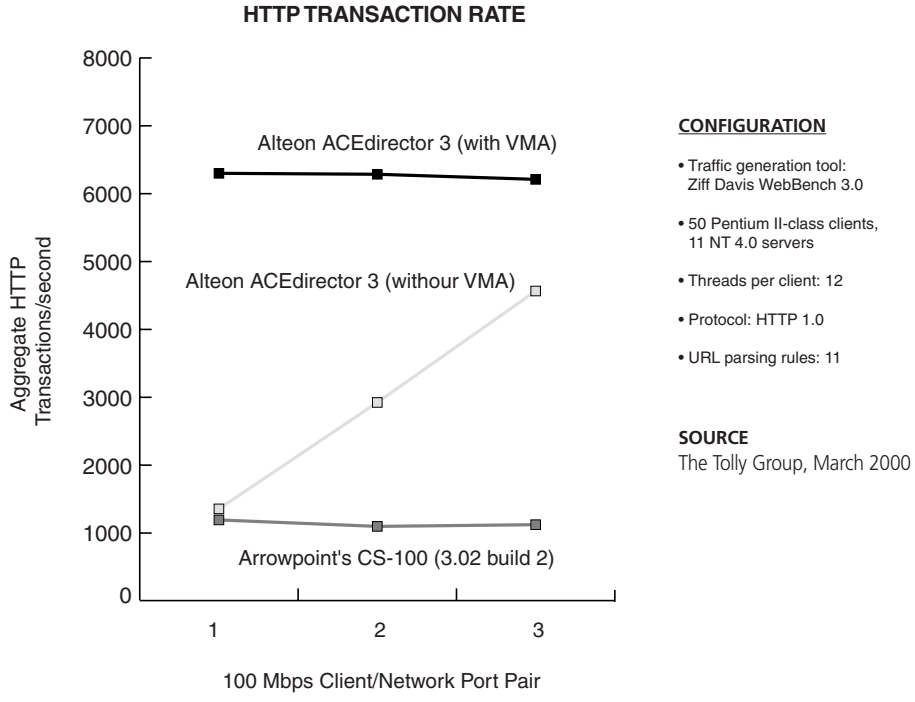
The persistence processor is dedicated to the task of synchronizing control information across processors that manage part of a persistent session. It is not used as a designated processor for processing non-persistent connections - a good example of the flexibility that the VMA model offers in terms of task distribution across distributed processors.

Advantages of the VMA Architecture

With VMA, the Web switching system delivers a high degree of parallelism since multiple processors can work in tandem to process incoming traffic, increasing total system performance. Memory on all ports is utilized. This dramatically increases the amount of memory available compared to a traditional distributed model. More memory means more session states stored—allowing session persistence to last longer. It also provides increased buffering for delayed binds and deeper, faster content parsing. Each port has access to all content and flow information in the device, eliminating topology and connectivity constraints. VMA is a software upgrade to all existing Alteon 180 and ACEdirector switches, underscoring the flexibility of Alteon's switch architecture.

Independently tests comparing Alteon Web switches enabled with VMA against Alteon Web switches without VMA, show than when enabled with VMA, Alteon Web switches are able to process HTTP sessions four times

FIGURE 9
Layer 7 URL Switching Performance Tests



© 2000, Alteon WebSystems, Inc.

faster on a single 100 Mbps connection (see Figure 9). The tests showed that the ACEdirector 3 equipped with VMA achieved 6,300 transactions per second (tps), versus 1,356 for the ACEdirector 3 Web OS version 6.0.52 product — more than four times the performance. Moreover, the results indicate that the VMA now enables ACEdirector 3 to support a larger number of concurrent connections across a single Fast Ethernet client/network port compared to earlier software versions.

CONCLUSIONS

A Virtual Matrix Architecture delivers what IT buyers demand from a modern Web switch:

- The flexibility to support new architectures and topologies. Because the VMA can dynamically bring processing resources to bear on heavily loaded ports, it adjusts automatically to surges in client traffic or server traffic, as well as changes in load from route fluctuations.
- The functionality that comes from fast processing of content-level information as well as from performing many concurrent delayed binds for incoming traffic. In the VMA model, all processors and memory are brought to bear on traffic.
- The performance of a distributed architecture in which control and forwarding planes are directly attached to the high-capacity backplane and in which ingress port hardware can immediately handle lower-layer forwarding of traffic.
- The value and TCO savings that come from good port density, solid investment protection, and superior price-for-performance.

In short, a hybrid distributed/centralized model is essential for next-generation Web switches and VMA delivers this on existing and emerging Alteon platforms. ■

© 2000, Alteon WebSystems, Inc.