Application Firewalls for High-end Network Security

How multi-core processors with hardware acceleration can enable fast, efficient application-level firewalls

Firewalls are an indispensible component in virtually all of today's enterprise networks. No IT manager would consider operating a WAN without the protection and security that firewalls provide. Today's firewalls are sophisticated and powerful to meet the ever-increasing needs of complex, high-bandwidth networks.

Earlier generations of firewalls, however, were much simpler. As long as a firewall could open or close a port based on authentication, the IT manager was able to control who had access to what applications and when. But now the Internet has become an essential part of daily life and the work environment. Firewalls are needed that provide additional control at the application level. A port may be open, but what application is using it? Based on the content, IT managers need to be able to apply *application level firewall* controls. This task is orders of magnitude more complicated and compute-intensive and can choke traditional firewalls pushing single-core x86-based firewalls to their limits and beyond.

Leading firewall makers like Cisco, F5, and Palo Alto Networks have turned to embedded multi-core processors with hardware acceleration to meet their processing throughput requirements. This case study describes how firewall tasks can be optimized on a multi-core processor.

Pushing the Limits of Performance

Firewalls have expanded their role and their capabilities in the past decade. Wikipedia offers the following definition:

> A firewall is a dedicated appliance, or software running on a computer, which inspects network traffic passing through it and denies or permits passage based on a set of rules. A firewall is normally placed between a protected network and an unprotected network and acts like a gate to protect assets to ensure that nothing private goes out and nothing malicious comes in.

Early generations of firewalls allowed or denied the passage of data by opening or closing ports. This meant opening or closing the ports for applications such as the Web (Internet browser), email, Voice over IP (VoIP), file transfer protocol (ftp), and others. Either allowing or denying office workers access, for example, to the Internet was not a workable option. Everyone needs access to these applications.

But providing access can open applications up to viruses and abuse—both internal and external. More granularity of control and protection was needed, taking into account the kinds of applications being used and the types of data being accessed or transferred, going above and beyond just user identity and ports or addresses. A firewall was needed that could understand applications and the application data behind the network. This would allow the application firewall to apply the correct checks based on the applications/application types for the data being monitored.

Packet Inspection

To perform its function, firewalls must execute a complex, compute-intensive series of operations. Many packet flows traverse the network simultaneously. The firewall must be able to identify and work on many concurrent flows, determine where they are

coming from, where they are going, what protocol is involved, what application is involved, what is permissible based on specific policies, and block or allow that flow.

For example, the firewall proxies a TCP session. [See "Application Scaling for Cloud Computing" white paper for a detailed description of how TCP works.] Http flows may be compressed, so the firewall must also decompress packets before it can analyze them. Once the application firewall proxies the TCP session and is able to monitor the data being communicated within the flow, it must detect the higher-layer protocols and applications being used above the TCP layer. How it detects these depends upon the specific higher-level protocols and applications. In general, it requires a combination of tracking the relevant ports/addresses, performing pattern matching operations on a set of signatures, and comparing heuristics such as bit rates and usage patterns.

Not Just Traffic, But What Kind of Traffic?

The complexity of this complex processing can be demonstrated through an example. When monitoring Internet traffic, there is not just one packet. There are flows of packets. And within these flows there are different applications and different protocols that require different methods for tagging or identifying them. The first challenge then is to identify the flows based on TCP, IP, and protocol.

The second challenge is to go to the next level and detect what kind of application or protocol the flow is. For example, is it email, ftp, or http traffic? After the application type is identified, different applications might have different policies. Email traffic requires different policies than wireless transfers, or removing spam. But if it is https traffic, the Web site must be determined and, based on a black list, possibly blocked. So it is a very different processing task after that point. Just to get to that point demands a lot of processing, because the firewall is in the middle between two endpoints. If the firewall cannot keep up with the flow, it could hold up the traffic. The challenge then is to accomplish all this processing in real time.

The inability to keep up with the flow could negatively impact latency-dependent applications, such as Voice over IP (VoIP) or streaming video, affecting the quality of the user. For this reason, the firewall may be called upon to prioritize based on applications. Other tasks may include pattern recognition as well as maintaining statistics where in order to identify the issue, apply policy, or identify protocol the firewall must track how many bits per second have been used by this flow, this user, etc. Needless to say, it is very difficult to perform all of these tasks with one processor.

In an effort to increase the throughput of firewalls, manufacturers divided tasks up among multiple coprocessors and wrote software programs to coordinate the activities of the different CPUs. But this required large, expensive processors and consumed a great deal of power. Something more was needed.

Multi-core Processors Arrive

When it comes to processing internet traffic in real time, single-core processors have given way to multi-cores in the past few years, offering a potential solution to the need for speed in firewalls. Multi-core processors can be very efficient at parallel processing. Deep packet inspection requires the simultaneous analysis of many individual packet flows. Each flow can be assigned to a core, accelerating the process.

Multi-core is a significant advancement over single-core processors, but considering the scale and complexity of processing needed, today's multi-core processors must also provide application acceleration as well.

Multi-Core Programming is Key to Exploit Hardware Features

Multiple cores alone will not deliver the desired performance. In a scenario using general-purpose processors, packets arrive continuously. A processor must determine what flow each packet is associated with and forward it to the core that is currently

working on that flow. If it is a flow that some other processor is already working on, then processing cannot be done in parallel. This is because when two packets belong to the same flow, they must be processed in order.

Performance is also hindered by overhead associated with accessing shared resources. To prevent another core from disrupting that data, the resource must be locked. This is why hardware schedulers built into a multi-core chip can greatly simplify the overhead tasks, because they can detect individual TCP flows and distribute the processing of flows among cores. One must know multi-core programming in order to use these features. Cavium's OCTEON family of multi-core processors includes hardware schedulers and accelerators.

Improved Performance with Hardware-based Multi-core Synchronization

New multi-core processors are much more efficient at processing all of the tasks associated with packet inspection because of hardware-based multi-core synchronization. For example, there is no need for a separate CPU to determine what flow each packet belongs to. Hardware-based multi-core synchronization automatically receives packets, sorts them by flow, and routes them in the correct order to the core assigned to that flow, eliminating the need to synchronize among cores. Packets belonging to different flows can be worked on immediately; there are no delays due to scheduling or resource locking. Hardware acceleration also speeds up TCP operations, decompression, and pattern matching. And because all of the processing is performed on a single chip, there are no external coprocessors that can add latency.

Pattern Matching Hardware Acceleration

Physically removing the need to synchronize tasks among multiple cores through hardware acceleration reduces a lot of the overhead that processors without acceleration must contend with. Another source of overhead is pattern matching. As packets are inspected, they are compared to rules and processed accordingly. For

example, a stream of http packets is received. This stream is recognized as a Web site address, which triggers a comparison with a list of blocked Web sites in a database. If the two match, the firewall blocks the session. Rules and signatures can be written in regular expressions that provide more flexibility for specifying them. Regular expression matching requires significantly higher complexity and processing power or acceleration. Hardware acceleration speeds up the expression-matching/pattern-matching operation, improving performance.

10X Performance Boost in Processing Internet Traffic

Through multi-core processors with hardware acceleration, firewalls today can process packet streams of 5 Gbps or more. All processing is performed economically on a single chip. Firewalls based on general purpose multi-core processors using only software implementations struggle with 100 Mbps— even with external TCP or compression off-load engines to handle some of the processing. Separate chips require more power with lower performance.

Multi-core Programming

Firewall designers will find a fundamental difference between programming a single-threaded core versus programming a multi-threaded multi-core processor. The biggest difference is that software for multi-core processors must be written to facilitate processes to run in parallel (i.e., able to detect individual flows and then spawn threads to process each of the flows with a synchronization mechanism to manage shared data accessed by multiple threads).

Designers will take another leap from a generic multi-threaded programming to the architecture used by Cavium in its multi-core processors, which provides hardware acceleration for detecting packet flows, scheduling processing of individual flows, and accelerating operations like decompression/compression and pattern matching. There is

actually less software to be written because most of the processing is performed in hardware rather than software.

Deep Packet Inspection

As this document has shown, hardware acceleration is a desirable capability for multi-core processors. When considering multi-core processors, designers are presented with a choice: general-purpose multi-core processors that provide hardware acceleration by off-loading processing tasks to co-processors, or multi-core processors such as OCTEON that integrate virtually all of the processing operations on a single multi-core processor. OCTEON delivers the advantages of significantly lower power consumption at equivalent or higher performance and higher performance packet processing. This is the result of efficient processor implementation and utilization of more cores running at lower frequency.

About Cavium Networks

Cavium Networks is a leading provider of highly integrated semiconductor products that enable intelligent processing in networking, communications, wireless, storage, video and security applications. Cavium Networks offers a broad portfolio of integrated, software-compatible processors ranging in performance from 10 Mbps to 20 Gbps that enable secure, intelligent functionality in enterprise, data-center, broadband/consumer and access and service provider equipment. Cavium Networks processors are supported by ecosystem partners that provide operating systems, tool support, reference designs and other services. Cavium Networks principal offices are in Mountain View, CA with design team locations in California, Massachusetts and India. For more information, please visit: http://www.caviumnetworks.com.