

# Connection Handling in the Cisco Application Control Engine (ACE30) Module

## What You Will Learn

The Cisco® Application Control Engine (ACE) family of products is a data center solution for increasing the availability, performance, and security of applications and web services. This document provides a technical explanation of how connections are handled in network processors of the Cisco ACE30 Module. This document also discusses connection load balancing, Network Address Translation (NAT), SSL offload, and HTTP compression at the hardware architecture level. This document is intended as a technical reference for networking professionals familiar with application delivery, network design, and facilitation of services, and those who are interested in better understanding how the Cisco ACE30 Module uses network processors to handle connection processing.

## Hardware Architecture

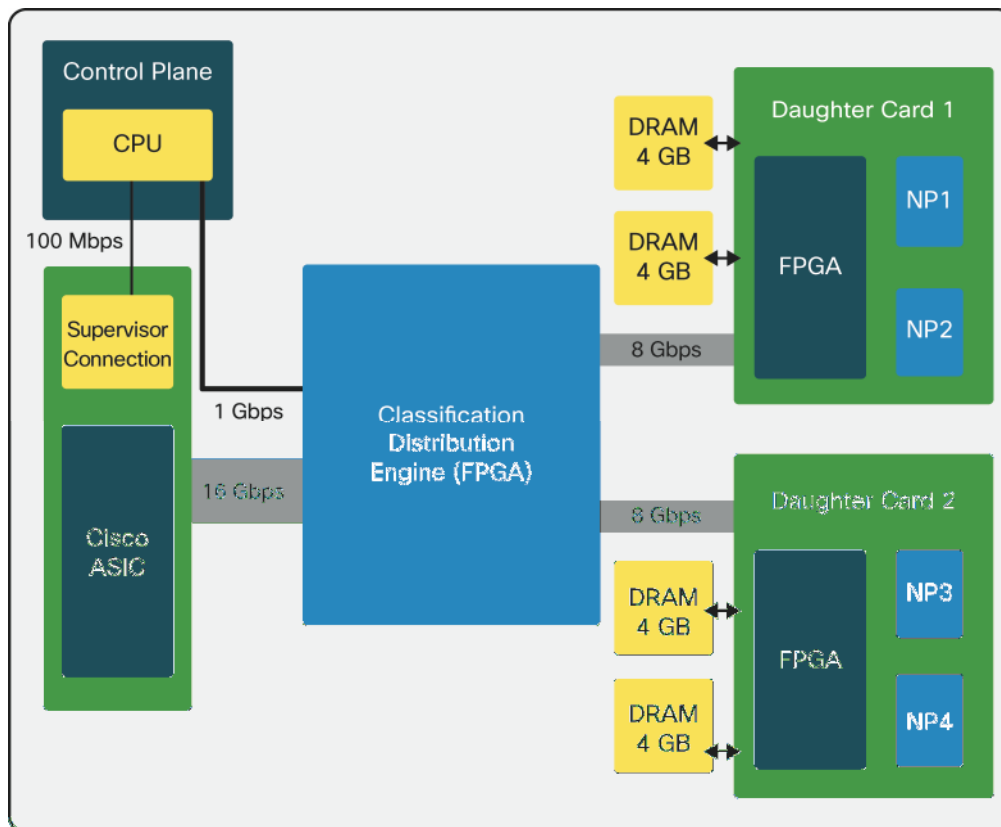
The Cisco ACE Module is a CEF720 line card connecting to the Cisco Catalyst® 6500 Series Switches. It receives all connections from clients and to the server over the Cisco Catalyst 6500 Series switching backplane using a Hyperion application-specific integrated circuit (ASIC), which is commonly found on WS-X6748 modules for the Cisco Catalyst 6500 Series. The Cisco ACE Module has interconnects to the Ethernet out-of-band channel (EOBC; 100 Mbps), shared bus (8 Gbps), and switch fabric (20 Gbps). The EOBC is used to allow the Cisco IOS® Software session command access and to allow the Cisco ACE30 Module to boot from the supervisor's storage if necessary. The shared bus interconnect is used only if communication to classic cards is required. The Cisco ACE30 Module uses a single interconnect to the switching fabric to provide a 20-Gbps connection to support up to 16-Gbps throughput.

After a connection is received by the Cisco ACE Module, it can be processed in one of three main components, depending on the type of connection:

- Classification and distribution engine (CDE): Application-specific field-programmable gate array (FPGA)
- Control plane: Dual-core SiByte MIPS processor clocked at 700 MHz, commonly referred to as the control plane
- Data plane: Two daughter cards, each card housing two Octeon processors (CN5860); each Octeon processor has 16 cores that run at 600 MHz

Figure 1 shows the logical layout of the Cisco ACE Module and includes the three main components.

**Figure 1.** Hardware Architecture Overview of the Cisco ACE30 Module



As traffic enters the Cisco ACE30 Module, there is a clear separation between the in-band and out-of-band communications channels. In general, traffic destined for the Cisco ACE Module is categorized as either to the device or through the device.

- To the device, or traffic destined for an interface VLAN IP address configured on the module: This traffic matches a class map of type management, which is applied as a service policy to an interface VLAN. The management class map supports these protocols: HTTP, HTTPS, Internet Control Message Protocol (ICMP), Keepalive Application Protocol (KAL-AP), User Datagram Protocol (UDP), Simple Network Management Protocol (SNMP), Secure Shell (SSH) Protocol, and Telnet. This traffic is called control-plane traffic.
- Through the device, or any traffic not matching a class map of type management and traversing the Cisco ACE30 Module: For load-balancing configurations, traffic must match a class map on a virtual address. For access control lists (ACLs), inspections, NAT, or TCP/IP, normalization traffic can match a class map, or more commonly traverse an interface configured on the module. This type of traffic is commonly referred to as client-server traffic or load-balanced traffic and is known as data-plane traffic.

### Control Plane

Traffic destined for the Cisco ACE30 Module and allowed by the management policy is sent directly to the control plane for processing. The control plane provides the user interface for the Cisco ACE30 Module. Features provided by the control plane include:

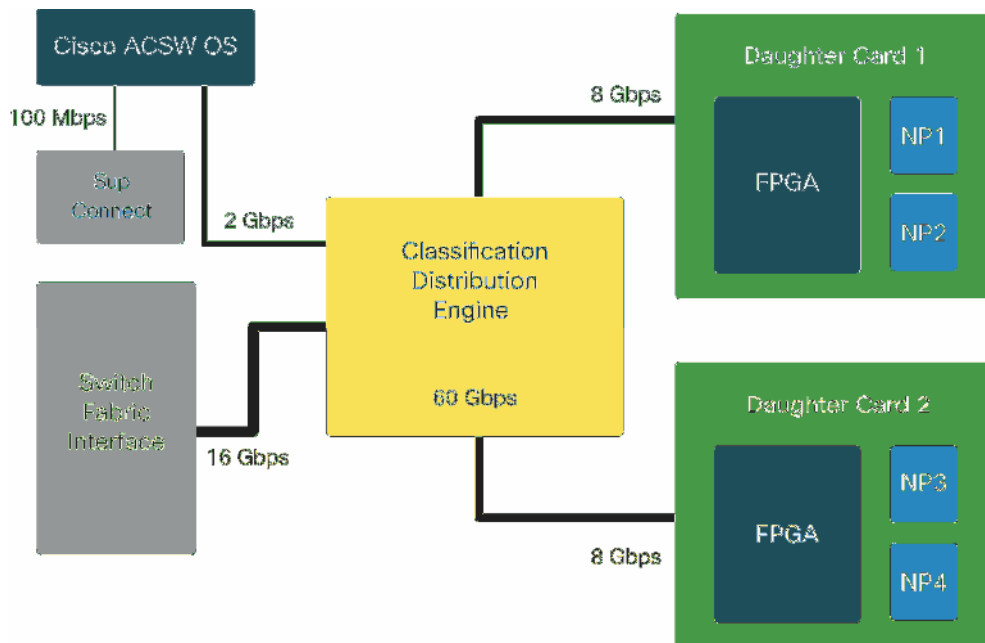
- Configuration manager
- Cisco ACE XML interface
- Device management through the command-line interface (CLI), MIBs, syslogs, etc.
- Address Resolution Protocol (ARP) resolution
- Routing table maintenance and routing protocols (only static routing is supported at this time)
- ACL compilation
- Health monitoring of real servers

The control plane can initiate connections for health monitoring of servers, logging of syslog messages, SNMP notification, and so forth. The control plane never initiates or receives connections or packets directly from the Cisco Catalyst 6500 Series. Control-plane traffic always crosses the network processors as they provide advanced protection for the control plane. For example, packets such as ARP requests are automatically rate limited so that they do not overwhelm the control plane. In addition, an out-of-band (OOB) connection interconnects the main components (except the daughter card slots) through a peripheral component interconnect (PCI) bus (32 bits at 33 MHz). The OOB connection is commonly used by the control plane to directly access the memory of the data plane. The configuration manager, running in the control plane, uses the OOB connection when pushing down a new configuration or the resulting ACL merge list from an access list compilation. Each of these control-plane communications occurs outside the data path. Thus, control-plane traffic, even oversubscription, will not affect data path traffic when traversing the Cisco ACE Module.

## Data Plane

The data plane consists of two of the three main components: the CDE and daughter cards, as shown in Figure 2.

**Figure 2.** Data-Plane Components



New connections enter the Cisco ACE30 Module from the Cisco Catalyst 6500 Series switching fabric. The svccl (service line card) command is used to allow VLAN traffic to access the Cisco ACE30 Module. The Cisco Catalyst

---

6500 Series Supervisor Engine 720 performs a destination MAC address lookup and identifies the Cisco ACE30 Module as a next hop, which is similar to the way other service-module traffic is handled. After it is allowed to the Cisco ACE Module, the initial packet of the connection goes to the CDE, where it is buffered for a very short time. The CDE is a FPGA developed by Cisco that can be thought of as a switch inside the switch. The CDE is a smart switch residing on the baseboard and acts as a central point of contact between all the main components located on the module. The CDE computes and, if necessary, adjusts the IP, TCP, and UDP checksums of every packet. It also performs virtual output queuing (VoQ) between the components—control plane and daughter card—to help ensure proper delivery of internal communications. The CDE appends a special header known as the IMPH header to each packet before sending it to the FastPath. The IMPH header is 18 bytes long and contains information from the DBUS header (the header sent to the Cisco ACE30 Module by the Cisco Catalyst 6500 Series Switch) as well as special messaging directly understood by the FastPath. Fields in the IMPH header can include notification of a checksum error, Layer 3 or 4 offsets, source and destination ports of the CDE, VLAN for determining the interface that the FastPath will use, and so forth.

Most important, the CDE is responsible for determining where to send a particular packet. Currently, there are four possible options: NP1 or NP2 (on daughter card 1) or NP3 or NP4 (on daughter card 2). All traffic entering the Cisco ACE30 Module, including traffic destined for the control plane, always must traverse one of the network processors. Considering the control plane as a host behind a firewall, where the network processors provide the protection, makes the CDE destination selection easier to understand. To make its path decision, the CDE uses a hash algorithm using information from the incoming packet and produces a 6-bit-wide result. The hash algorithm is based on the following:

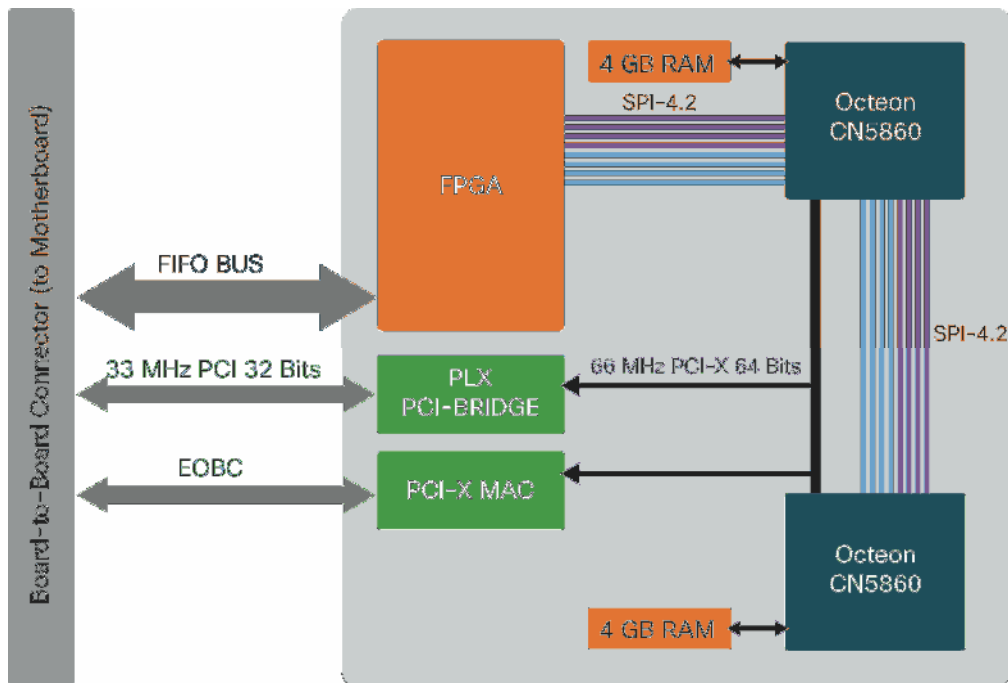
- Source and destination ports for UDP and TCP traffic
- Source and destination IP for IP traffic
- Source and destination MAC address for non-IP traffic

The load-sharing algorithm implemented by the CDE is not user configurable. The goals of the CDE are to help ensure that a given flow always hashes to the same network processor and to reduce the megaproxy effects for commonly used protocols such as TCP and UDP. Although the CDE can be thought of as a switch, it does not perform any destination MAC address–based lookup. The CDE’s main purpose is to compute a hash, append the Cisco internal header, and direct the packet to one of the network processors.

### Daughter Card Complex

After the CDE has made its path decision, the packet enters one of the two daughter card complexes. Because all the real work of the Cisco ACE30 Module is done within the daughter cards, it is important to understand the main components of a daughter card (Figure 3).

**Figure 3.** Daughter Card Complex



Each daughter card complex consists of the following main components:

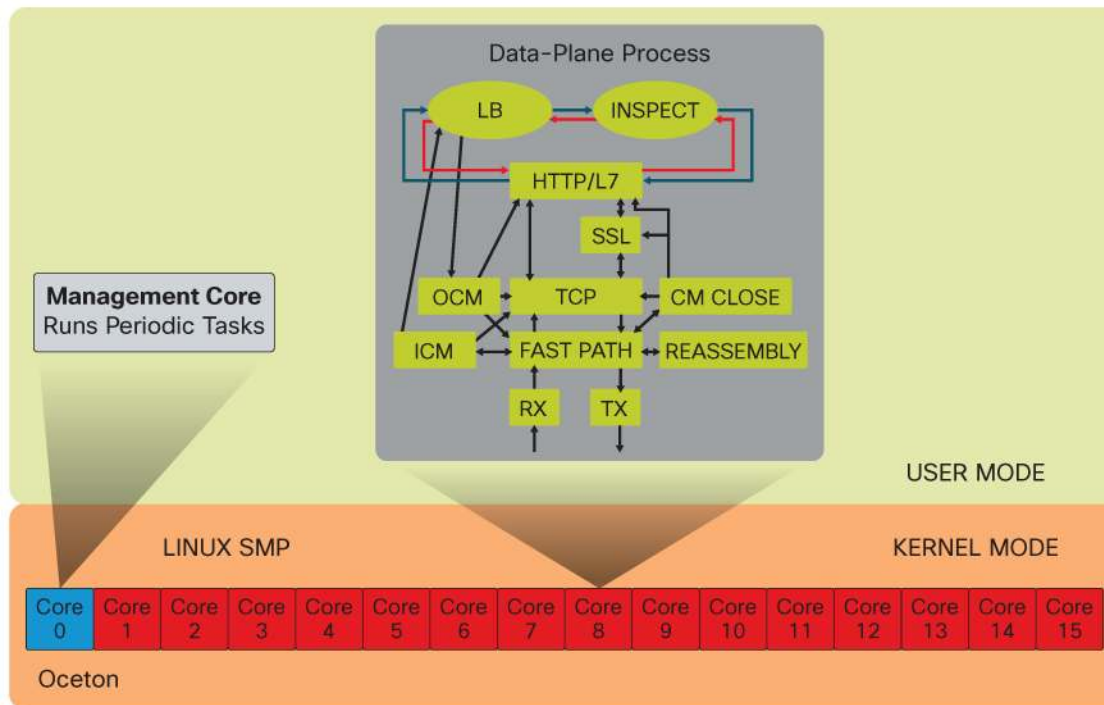
- VERNI FPGA, which provides FIFO32-to-SPI4.2 bridging capability
- Two Octeon CN5860 processors; each Octeon has 16 cores running at 600 MHz with 4-GB DRAM support on chip encryption and decryption and a coprocessor for compression and decompression
- SPI4.2 interface
- PCI Extended (PCI-X) interface

Verni FPGA's main role is to route traffic to specific Octeon network processors based on the IMPH header, which is inserted by CDE. Verni FPGA has four VoQ's for each Octeon processor, which act like data paths that the packets take to reach each Octeon processor. Other important components are the PLX PCI bridge and PCI-X MAC address, which are used to communicate OOB with the control plane.

### Data-Plane Software Architecture

The data-plane software architecture of the Cisco ACE30 (Figure 4) is similar to that of Cisco ACE 4710 running Linux in symmetric multiprocessing (SMP) mode. The Cisco ACE 4710 has only one Octeon network processor, whereas each daughter card has two Octeon network processors per card. The data-plane software process runs as a user process and has complete user-mode access to Octeon hardware without the overhead of kernel-mode context switching. To achieve parallelism, 16 instances of the data-plane process run on 16 cores per Octeon network processor, and CPU affinity is achieved by tying each instance to a specific core. Out of 16 cores, one core is dedicated to processing and interacting with the control plane to handle management tasks and the timer functional block.

**Figure 4.** Software Architecture of Single Octeon Network Processor



The data-plane software process consists of the following functional modules: FastPath, Inbound Connection Manager (ICM), Outbound Connection Manager (OCM), Connection Manager Close (CM-Close), Receive (Rx), Transmit (Tx), Timers (running on core 0 only), Reassembly, HTTP, SSL, Load Balancing (LB), and Inspect. The main difference between the Cisco ACE20 Module implementation and the Cisco ACE30 and ACE 4710 implementation is that in the Cisco ACE20, each functional module runs on a dedicated microengine, whereas in the Cisco ACE30 and ACE 4710 implementation, all the functional modules are combined into a single process or thread. All 16 cores in each Octeon processor share the memory, and shared data (connection table, sticky table, etc.) resides in shared memory for access by all cores. Hence, there is no need for packet and flow affinity for a core because any core can handle any packet; however, note that flow affinity is needed to the Octeon network processor, and this is provided by CDE. In summary, the execution model for the Cisco ACE30 and ACE 4710 is to run the data-plane process in parallel mode. This model has some advantages over the pipelined model used in the Cisco ACE20:

- Performance not limited to a single core: For example, FastPath could be sending packets on all 16 cores simultaneously instead of being limited to a single microengine in the Cisco ACE20 data plane.
- Better cache hit rates: Since all processing for a given packet happens on one core, better cache hit rates are achieved.

### Connection Handling

As a connection is sent from the Cisco Catalyst 6500 Series Supervisor Engine 720 to the Cisco ACE30 Module, the initial connection packet is dispatched to one of the four network processors by the CDE. The data-plane Rx module accepts the packet from the hardware and forwards a pointer to the packet to a FastPath module after preprocessing. Within the FastPath module, checks are made to determine whether the packet matches an

existing connection, or whether it is a new connection requiring a connection setup. If the packet is a TCP or UDP packet, a 5-tuple-flow lookup (source IP address, source port, destination IP address, destination port, and incoming VLAN) is performed to determine whether the flow has already been established.

If a new connection is required, the FastPath module sends the pointer to the packet to the ICM module. The ICM module performs a 5-tuple lookup to determine which features need to be applied to the flow. The output of the lookup is an access control entry, which is applied to permit or deny the traffic and also the list of actions to apply (such as load balancing, inspection, and NAT).

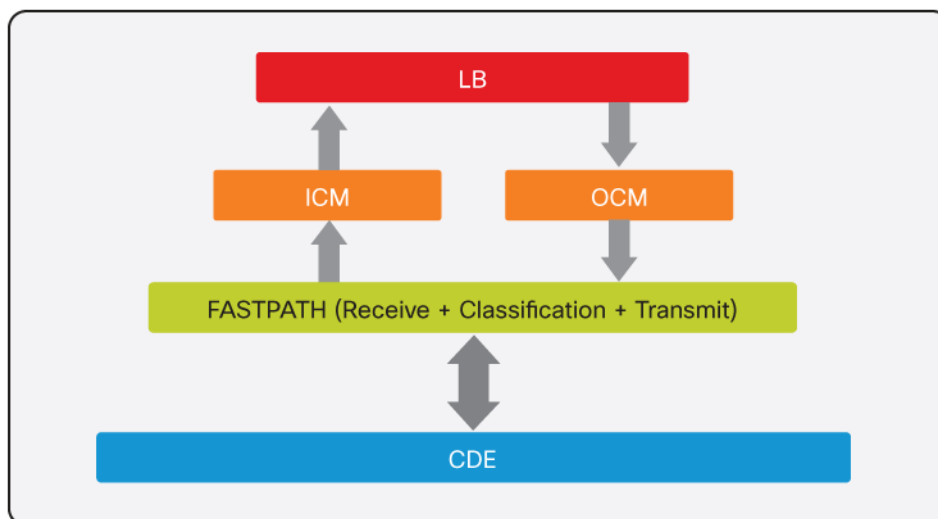
If the packet is denied, the packet is dropped, and a syslog message is sent to the syslog server directly from the ICM, or the message is sent to the control plane, from which it is forwarded to the syslog server.

If the packet is permitted, further checks are performed against it, notably checks for invalid TCP flag combinations, sequence number verifications, TCP window checks, unicast reverse-path forwarding (RPF) check, etc. If the packet successfully passes the checks, a connection ID is created, and one of four other cases must exist:

- Connection is destined for a Layer 3 or 4 virtual IP address
- Connection is destined for a Layer 7 virtual IP address
- Connection is not destined for a virtual IP address
- Connection is for Layer 3 or 4 load-balancing traffic

If the packet matches a virtual IP address with any port (Layer 3) or a well-defined destination port (Layer 4), the traffic is considered to be server load balancing (SLB) only. The logical packet flow is shown in Figure 5.

**Figure 5.** Processing SLB-Only Connection

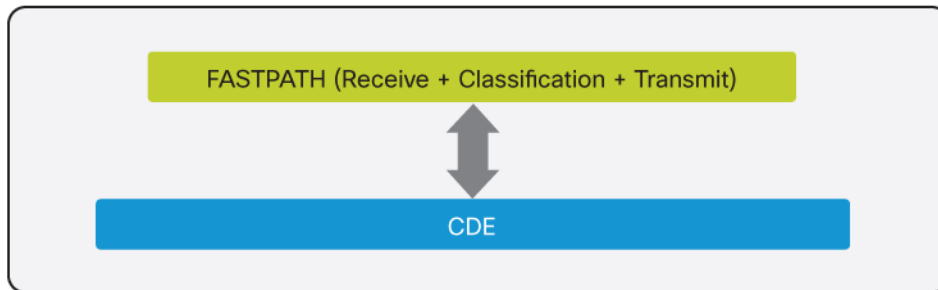


After the ICM module determines that the packet is destined for an SLB-only virtual IP address, the module sets up the Layer 2 rewrite for server response traffic in the inbound connection record and sends a message to LB module. Depending on the nature of the policy configured on the interface, the ICM module may send a reference pointer to the full packet to inspect module for more detailed processing, as is done with FTP-, RTSP-, SIP-, and HTTP-inspected traffic. The LB module is responsible for making the load-balancing decision. It returns a result (the real server that the connection should use) to the OCM module. If Cisco ACE is configured to perform NAT or

Port Address Translation (PAT) on the connection, the OCM will allocate a client NAT or PAT address to help ensure that the CDE sends return flows to the same network processor that the initial packet traversed. This process helps ensure network processor persistence per connection, even for NAT and PAT configurations. The OCM updates the connection object to include the Ethernet and IP rewrites and forwards the packet to the FastPath module. The FastPath module rewrites the Ethernet and IP headers and sends the packet to the Tx module, which forwards the packet to the hardware. After the packet arrives at the CDE from the daughter card complex, the CDE forwards the packet to the Cisco Catalyst 6500 Series Supervisor Engine 720 for correct Layer 2 or 3 forwarding to the real server.

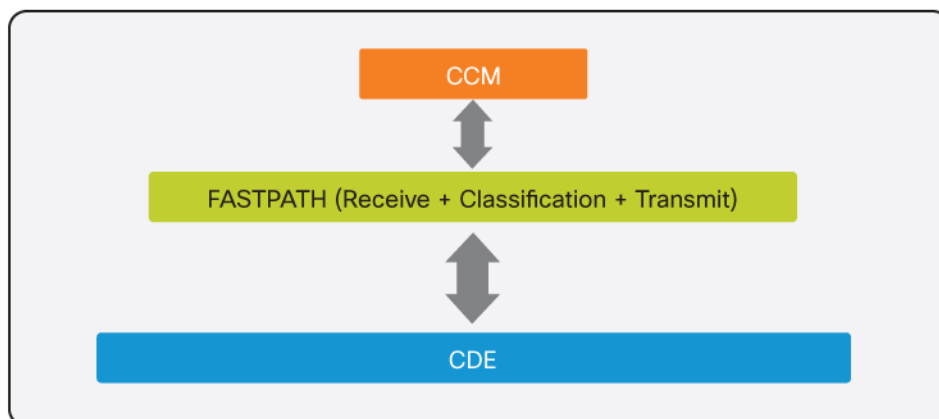
As subsequent packets from this connection arrive, the CDE directs them to the same network processor that handled the initial packet. These packets are received by the Rx module and pushed to the FastPath module. When the FastPath module performs the 5-tuple lookup, it finds an established connection. At this point, the FastPath module performs the appropriate rewrite (Ethernet, IP, and port) as defined by the connection object information. The packets are then sent out of the daughter card complex to the CDE to be delivered to the server or client as required; see Figure 6.

**Figure 6.** Handling Subsequent SLB-Only Packets



When the Cisco ACE Module receives a TCP packet with a Finish (FIN) or Reset (RST) flag set, the FastPath module instructs the CM module to tear down the connection, as shown in Figure 7.

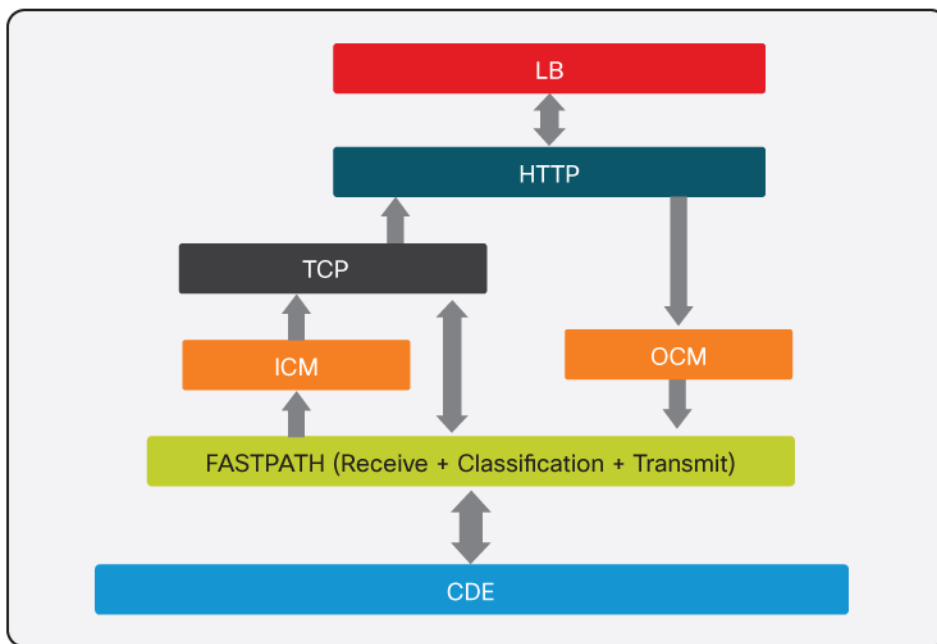
**Figure 7.** Connection Close



## Layer 7 Load Balancing

If the initial connection packet matches a virtual IP address and port (any or well defined) and matches an application load-balancing or inspection class map, it is considered a Layer 7 SLB connection. Typically, Layer 7 virtual IP addresses handle HTTP traffic in which the URLs are used to distribute to application services. However, many types of Layer 7 virtual IP addresses are supported by the Cisco ACE Module: TCP reuse, HTTP pipelining, SSL acceleration, voice-over-IP (VoIP) protocol inspections, and so forth. For these connections, the initial connection packet does not contain enough data to make a load-balancing decision; thus additional processing is required. Figure 8 illustrates the steps for handling a connection for a Layer 7 flow.

**Figure 8.** Processing Layer 7 Virtual IP Traffic for HTTP



Upon receiving the initial packet for a Layer 7 SLB connection, the ICM performs the same check as for SLB-only connections. The packet is then forwarded to the TCP module for TCP termination of the connection between the client and the Cisco ACE Module. The TCP module initializes its state for this flow and responds to the sender with a synchronize acknowledgment (SYN ACK), sending the TCP packet to the FastPath module for Layer 2 or 3 encapsulation and then to the client through the CDE. The client TCP ACK message is received by the FastPath module and sent to the TCP module after the 5-tuple lookup. The TCP module applies flag and sequence number checks. The TCP state machine is also responsible for handling other tasks such as calculating the round-trip time (RTT), sending an ACK message, and adjusting the window size. At this point, the client connection is set as a TCP proxy. The Cisco ACE30 Module is now the TCP termination point of the client-sourced connection, which enables application-layer load balancing and inspection.

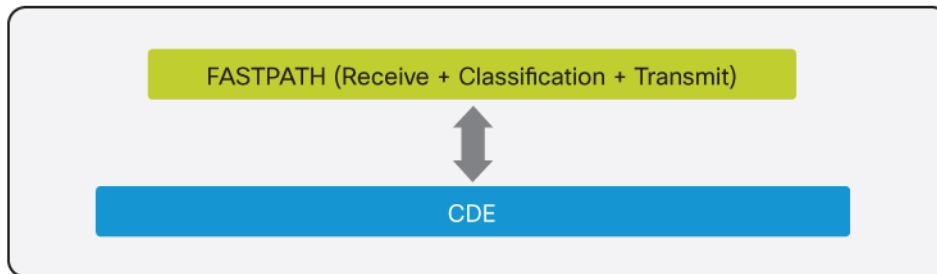
Upon receipt of the client's HTTP request, the packet is sent to the HTTP module through the FastPath and TCP modules. If the TCP packets are out of order, the TCP module will buffer the packets until they can be reordered and sent to the HTTP module in the correct order. The HTTP module will parse the client request for a match on the URL, HTTP header, cookies, and so forth. After a match is found, the HTTP module sends a message to the LB module to perform load balancing.

The load-balancing destination decision is sent directly from the LB module to the OCM module, which then sets up the outbound connection to the real server, where the Ethernet and IP layer rewrites are determined and updated in the connection object. Next, the OCM module forwards the packet to the TCP module to establish a (or reuse any available) TCP connection to the real server. At this point, TCP initiates a connection through the FastPath module and sends the packet from the CDE to the real server.

When the server responds with a TCP SYN ACK message, the CDE sends the packet to the appropriate Octeon network processor. When the packet arrives in the Octeon processor, the Rx and FastPath modules forward the packet to the TCP module. The TCP module performs the same TCP checks as were performed on the client side of the connection. If the checks are passed, TCP begins sending the server the client HTTP request as forwarded from the HTTP module.

After the TCP module has sent all the client data to the server and seen all the TCP ACK packets from the server, TCP configures the FastPath module with the appropriate TCP sequence number deltas for this flow. By instructing the FastPath module to manage the remainder of the flow, TCP has, in effect, unproxied the TCP connection. As a result, the remainder of the connection will be switched by the FastPath module, as shown in Figure 9.

**Figure 9.** Handling Unproxied Layer 7 SLB Packets



When the Cisco ACE receives a TCP packet with a FIN or RST flag set, the TCP module notifies the CM module, which then tears down the connection, similar to way SLB-only connections are removed (see Figure 7).

To conserve resources, the Cisco ACE attempts to unproxy TCP connections as quickly as possible, enabling higher scalability and performance. However, to properly handle client traffic, parts of the connection often need to be proxied to allow the Cisco ACE to make a load-balancing decision. This process is called reproxying, and it is used most commonly to handle persistence rebalancing, FTP inspection, and so forth. When a connection is initially proxied for Layer 7 load balancing, the connection is not unproxied until all the data has been sent from the server and acknowledged. In addition, the Cisco ACE inspects the server HTTP response to determine the amount of data that the server will return to the client. After the data size is determined, the connection object is updated, so that when the FastPath module sees the anticipated TCP sequence number, it pushes the packet to the TCP module for reproxying and inspection. If the HTTP server responds with a chunk-encoded HTTP response, the connection will continue to be fully proxied, because there is no way to calculate the TCP sequence number upon which to reproxy.

### NAT Translated and Inspected Connections

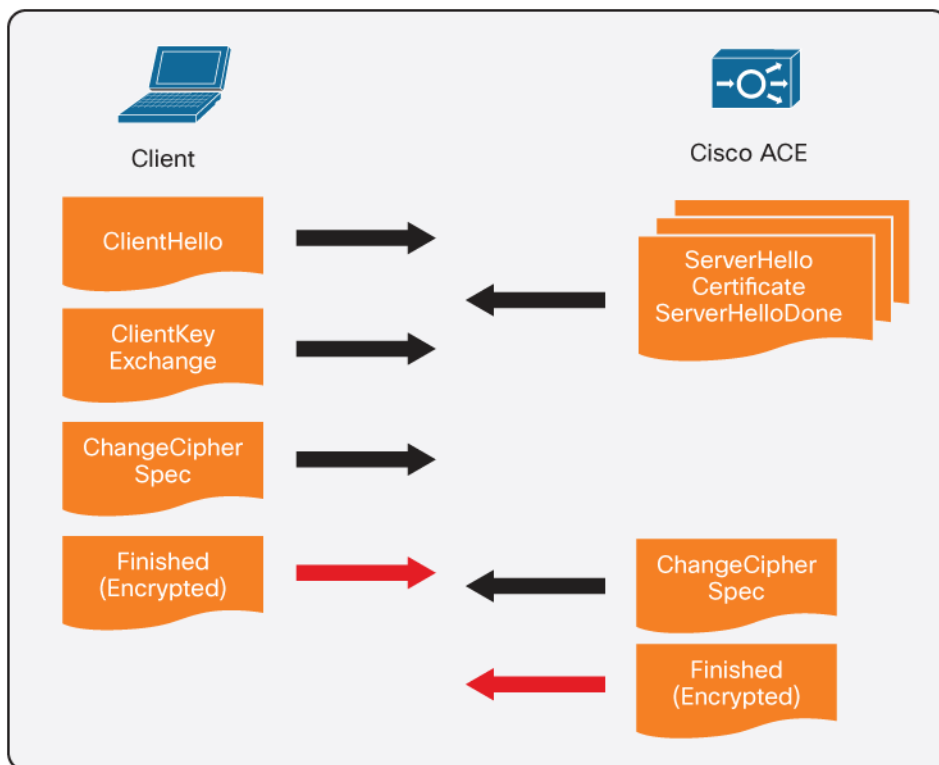
The previous sections discussed the actions that the Cisco ACE30 Module takes when a connection matches a virtual IP address. If the initial packet does not match any virtual IP address, the ICM module directly instructs the

OCM module to insert a new connection entry and route the packet out. An indication is not sent to the LB module; instead, the data plane gathers the statistics and creates syslogs for the new packet. These statistics are moved to the service policy statistics after the connection is closed. The packet is routed out to the CDE with a new Cisco internal header that informs the CDE that the packet is destined for the Cisco Catalyst 6500 Series switching fabric. This is the best-case scenario from a performance standpoint, because the life of the packet is very simple, and only a few functional modules are used.

### SSL-Terminated Connections

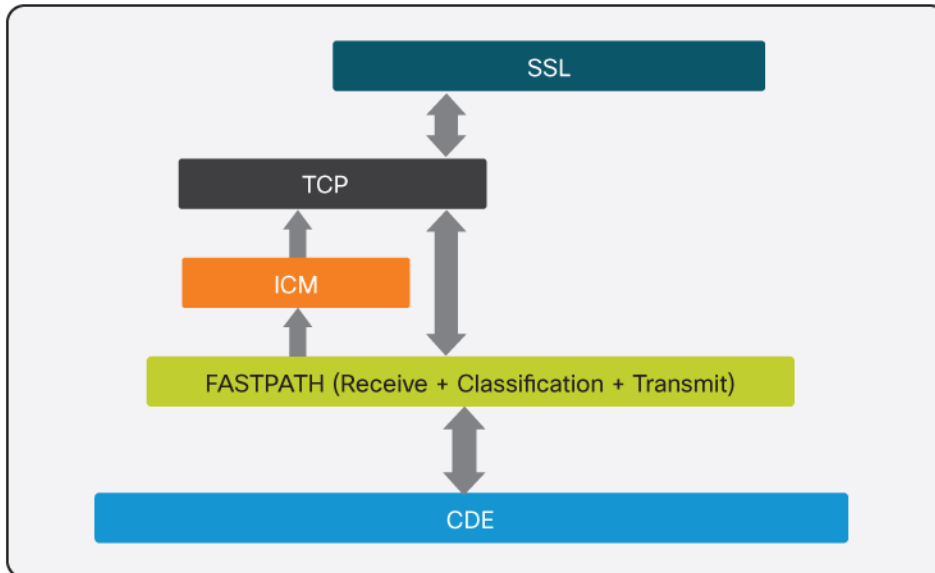
The Cisco ACE30 Module handles SSL-terminated connections at Layer 7 as full-proxy TCP connections. Thus, the initial setup is very similar to the setup defined in the previous section discussing Layer 7 load balancing. The main difference in setup is that the ICM module determines that the virtual IP address requires SSL application handling. The ICM module then instructs the FastPath module to send the SSL ClientHello message to the TCP module. The TCP module then passes a reference to the message to the SSL module, rather than sending the message directly to the HTTP module, as is done in Layer 7 load balancing. Figure 10 shows the external SSL messaging involved in terminating an SSL session.

**Figure 10.** SSL Handshake Messaging



After the SSL module receives the ClientHello message, it generates a ServerHello message, which is passed to the TCP module for transmission. In addition, the SSL module pulls the server certificate from the virtual IP address connection object to form the certificate message and generates a ServerHelloDone message. After these messages are created by the SSL module, they are passed to the TCP module, then to the FastPath module, and then to the CDE for transmission to the client. Figure 11 shows the logical processing of the SSL termination handshake.

**Figure 11.** SSL Processing of ClientHello, ServerHello, Certificate, and ServerHelloDone Messages



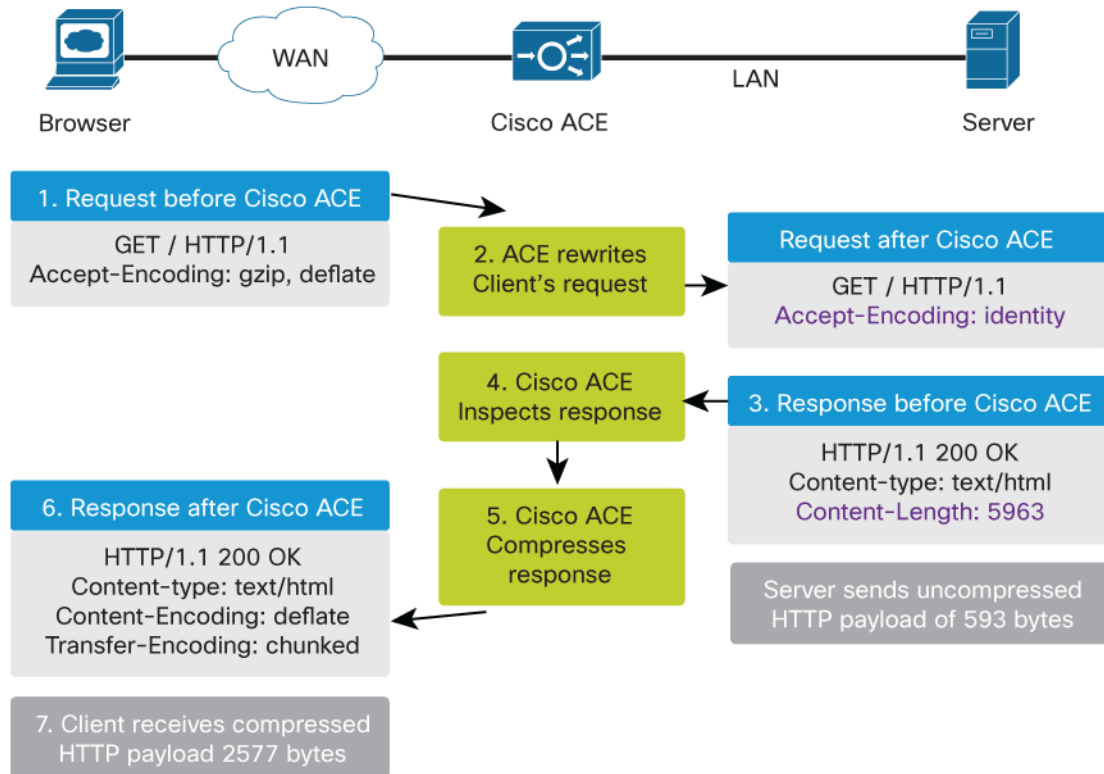
When the client responds, it typically sends the ClientKeyExchange (CKE), ChangeCipherSpec (CCS), and Finish messages as three SSL record messages combined in one TCP packet. The CDE sends the packet to the appropriate Oteon network processor, which then is handled by the Rx and FastPath modules, which forward the packet to the TCP module. The TCP module then passes a reference to the packet to the SSL module, which uses the CKE message to compute the master secret. When the client Finished message is received by the SSL module, the Rivest, Shamir, and Adelman (RSA) parameters are verified to help ensure that they have been exchanged correctly. Then the SSL module generates a ChangeCipherSpec message, which is passed to the TCP module. The message is then forwarded to the client, first through the FastPath module and then through the CDE. If the RSA parameters are correct, the SSL module generates a Finished message and sends it to through the CDE.

As encrypted application packets arrive in the Oteon network processor through the CDE, the Rx, FastPath, and TCP modules apply appropriate checks and pass in-order data to the SSL module. The SSL module reads the SSL record header to determine the length of the record and keeps accumulating data from the TCP module until it has a complete record. Then the SSL module sends an inline message to the hardware CPU for decryption. After the hardware CPU decrypts the packet, it is sent to the HTTP module, if Layer 7 load balancing is also required, or to TCP module for SLB-only connections. At this point, the connection is processed as either Layer 4 or, more commonly, Layer 7 load balancing. If the traffic is Layer 7 load balanced, the decrypted HTTP request is sent to the HTTP module for processing as described earlier in this document. Regardless of the load-balancing algorithm used, after the correct real server is selected, the TCP module establishes a TCP connection to the real server through the FastPath module, which is sent out of the CDE to the server. In addition, the clear-text return traffic from the real server is sent to the SSL module, which in turn sends an inline instruction to the hardware CPU for encryption. The encrypted data is then sent from the SSL module to the TCP module, to FastPath, and then out of CDE for delivery to the client. The Cisco ACE30 Module maintains the full-proxy Layer 7 connection to allow encryption and decryption of the client and server traffic.

## Compression

The Cisco ACE30 Module now can compress HTTP data, using a hardware co-processor (Zip engine) to do this job. Figure 12 shows the packet sequence of an HTTP1.1 request and response exchange when compression is performed.

**Figure 12.** Compression Sequence on Cisco ACE

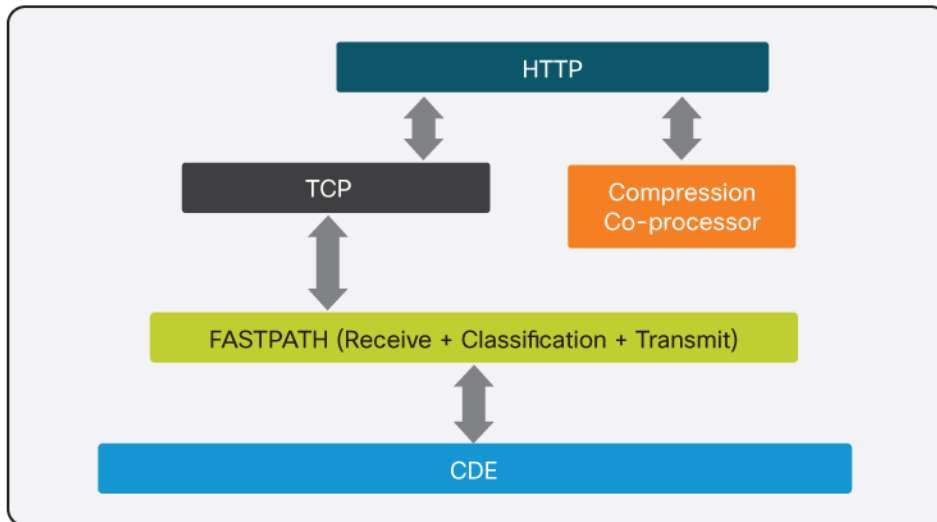


The connection handling for the traffic that needs to be compressed is similar to that for Layer 7 load balancing described earlier in this document. After the HTTP request is received by the HTTP module, the module performs the following operations for the HTTP request:

- The module uses the request line and header fields such as HTTP method, version, and user agent to determine whether compression can be performed for this request and response transaction. If so, it saves the state for this flow.
- If proceeding with compression for this request, the module next processes the Accept-Encoding header and decides what compression algorithm and format will be used to perform the task. It saves the information for this flow.
- The module forwards the request to the server, with the Accept-Encoding header modified to "identity".

Figure 13 illustrates the appropriate functional modules involved in the response compression.

**Figure 13.** Processing HTTP Response for Compression



The HTTP request will now follow the regular path from the HTTP module to the TCP module and then out of the FastPath module. After the server responds to the request, the response follows the same path through the FastPath and TCP modules and then to the HTTP module. Now the HTTP module performs the following operations on the HTTP response:

- Using the response status code, the content type information, and the state saved at the time of request processing, the HTTP module determines whether compression can be performed. If so, it saves the state for this flow.
- The module forwards the response header to the client after adding the appropriate Content-Encoding header set to gzip or deflates.
- After the response header processing ends, the HTTP module checks whether the state in the flow indicates that this response needs to be compressed. If so, it passes the input response buffer, along with some identifiers for the compression stream, to the Octeon Zip co-processor. The HTTP module also supplies an output buffer in which the compression engine will write the compressed output.
- The HTTP module then processes the output buffer that holds the compressed data; it produces a chunk based on the size of the compressed data in the buffer and generates the chunks, thus forwarding the compressed response body to the client.

## Conclusion

The Cisco ACE30 Module enables application and networking departments to achieve their business goals through a broad set of proven load-balancing and application-switching technologies, integrated with leading-edge application acceleration and security capabilities. A primary design element of the Cisco ACE30 Module, and a major differentiator between the Cisco ACE30 Module and other solutions in the marketplace, is its network processor architecture and its complete separation between the control and data planes. This design prevents control-plane traffic, even in the event of oversubscription, from adversely affecting client-to-server traffic in the data plane. The Cisco ACE30 Module terminates TCP traffic once, decrypts SSL, inspects traffic to help ensure protocol adherence, compresses traffic to reduce client bandwidth consumption and remove the need for the server to perform compression, applies application-layer rules to help ensure proper application delivery and

---

application persistence, and optimizes TCP use between the Cisco ACE30 Module and the application server. This integrated approach reduces latency and helps ensure a fast application delivery solution for data center application service environments.

**For More Information**

[www.cisco.com/go/ace](http://www.cisco.com/go/ace)



---

**Americas Headquarters**  
Cisco Systems, Inc.  
San Jose, CA

**Asia Pacific Headquarters**  
Cisco Systems (USA) Pte. Ltd.  
Singapore

**Europe Headquarters**  
Cisco Systems International BV Amsterdam,  
The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at [www.cisco.com/go/offices](http://www.cisco.com/go/offices).

---

Cisco and the Cisco Logo are trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and other countries. A listing of Cisco's trademarks can be found at [www.cisco.com/go/trademarks](http://www.cisco.com/go/trademarks). Third party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1005R)

Printed in USA

C11-689030-00 10/11