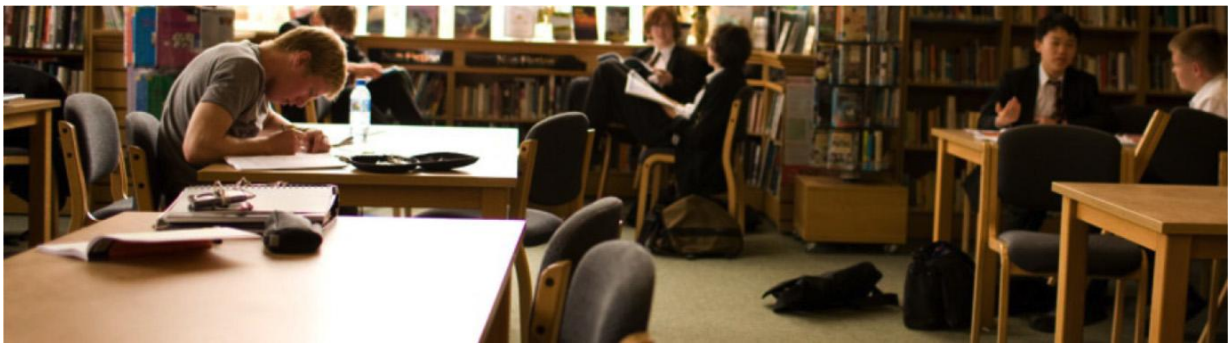
Guide

Cisco Application Control Engine Monitoring Guide



What You Will Learn

This document describes many of the ways in which you can monitor the Cisco® Application Control Engine (ACE) and how to observe its performance and identify problem areas both before and after trouble occurs.

Overview

Cisco ACE is based on a purpose-built multinetwork processor architecture. All data received by the interconnect at the switching fabric goes through a field programmable field array (FPGA), also known as the classification and distribution engine (CDE). Although the Cisco ACE Module has a 16-Gbps throughput channel, the Cisco ACE 4710 appliance is bound by four 1 Gigabit Ethernet ports. The Cisco ACE 4710 uses one network processor. The Cisco ACE30 Module is equipped with four network processors: two per daughter card (Figure 1). The CDE uses a hashing algorithm to help ensure that a single connection reaches the same network processor for its duration. Since the Cisco ACE 4710 has just one network processor, it does not use a CDE (Figure 2). Cisco ACE Modules use Cavium Octeon (CN5860) processors. Each processor has 16 cores and runs at 600 MHz.

The network processors make up the data plane and are responsible for handling all the connection processing in the Cisco ACE. The control plane is maintained separately with a dedicated SiByte processor. The control plane manages the Cisco ACE configuration, health monitoring (probes), interactive management sessions (Telnet, SSH, Simple Network Management Protocol [SNMP], and syslog), and other control functions and control traffic. For additional information about Cisco ACE architecture and connection processing, please ask your Cisco Sales representative for a copy of "**Connection Handling Within the Cisco Ace Module Hardware Architecture**."

Figure 1. Main Components of the Cisco ACE30 Module

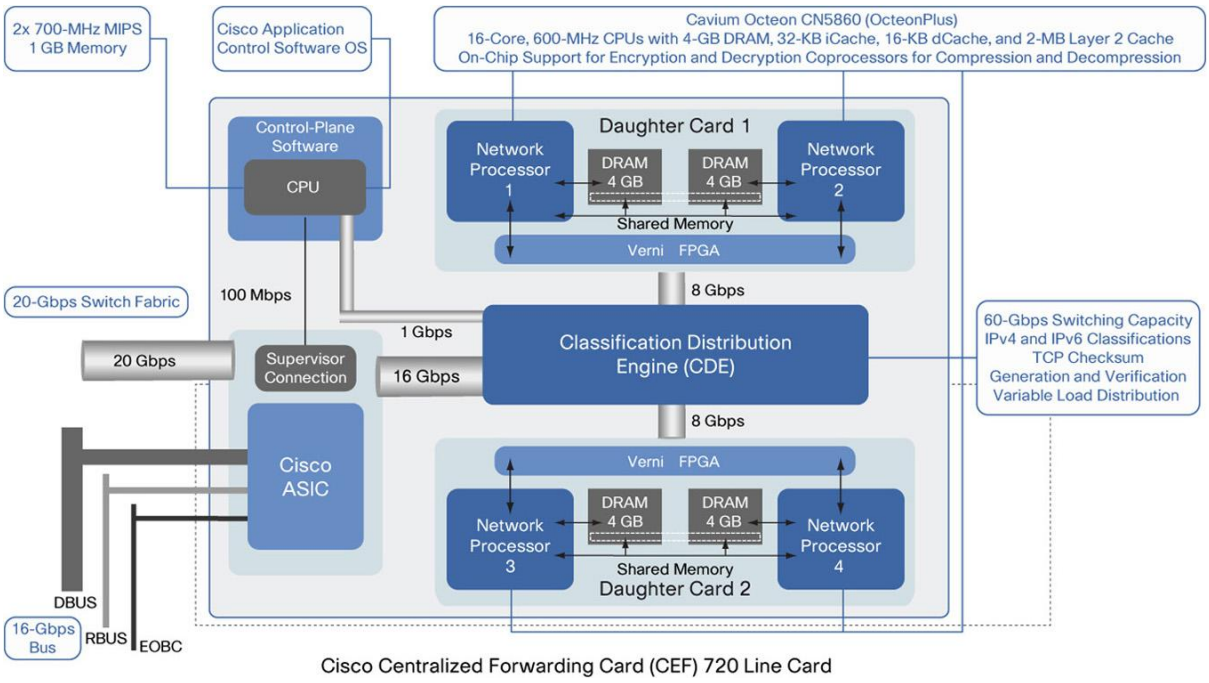
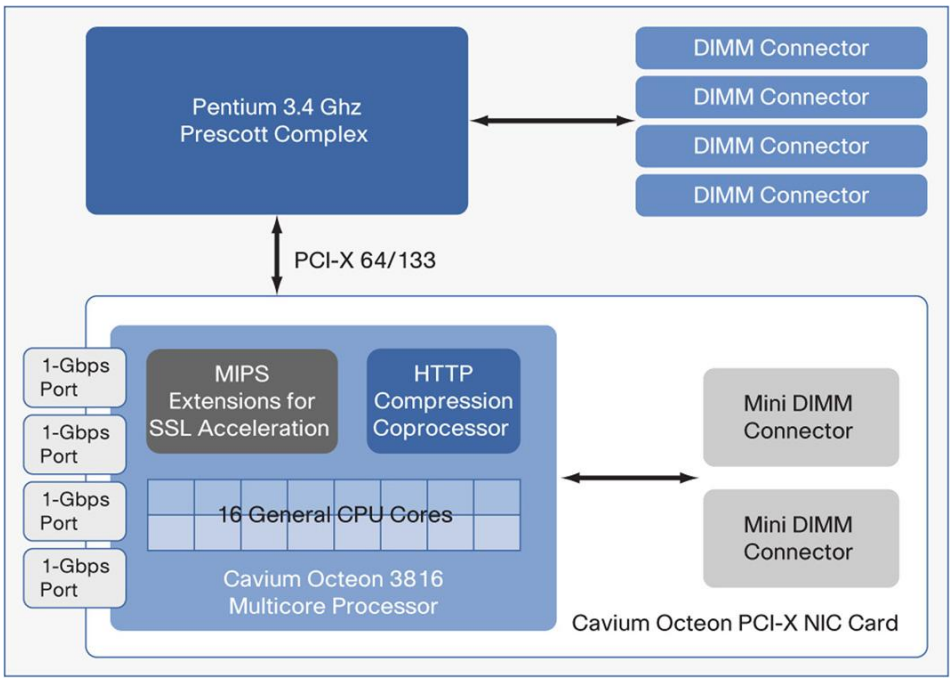


Figure 2. Main Components of the Cisco ACE 4710 Appliance



Data traffic is handled by each network processor using 15 cores, each running a set of data-plane software processes. The remaining core is for data-plane management and runs periodic processes (timers). The data-plane processes are grouped into the functional sets as follows:

- Receive (RX): RX is a dedicated process function used that takes traffic from the CDE and buffers it in the network processor for use by other network processor processes. RX also supports HTTP normalization, conformance with RFC 2616, MIME-type validation, length and encoding checks, port misuse, tunneling prevention, and so forth.
- Fastpath: Fastpath handles most of the low-level data occurs such as MAC address rewrite, Network Address Translation (NAT), and TCP normalization. Fastpath also performs NAT on the packet: depending on the configuration, this includes MAC addresses, IP addresses, Layer 4 ports, and TCP sequence (SEQ) and acknowledgment (ACK) numbers.
- TCP: The TCP process sends and receives TCP data from other network processor functions when data is processed at Layer 7 and the connection is fully proxied.
- HTTP Layer 7 fixup: This process supports HTTP and other Layer 7 application processing (fixups) such as header matching, rewriting, cookie processing, and persistent and pipelined connection. For application fixups, this process replaces IP addresses embedded in the application protocol data with the appropriate virtual IP address, server IP address, or client IP address and binds control and application channels together to help ensure proper processing by the network processors and real servers.
- SSL: This process performs SSL record-layer processing inline with a hardware coprocessor.
- Connection Close Manager (CCM): The CCM removes the internal connection objects for an established connection upon receipt of a TCP finish (FIN) sequence, TCP restore (RST) command, or connection timeout.
- Load balancing: This process facilitates communication and messaging from the microengine to the general processor. It allows microengine functions to access general processor functions such as load balancing, SSL state, FTP and Real-Time Streaming Protocol (RTSP) fixups, and high-availability heartbeats.
- Reassembly: This process manages reassembly of fragmented packets and TCP/IP timer control during connection handling.
- Inbound Connection Manager (ICM): The ICM is responsible for the creation of new connections. If the connection is to be load balanced at Layer 3 or Layer 4, the ICM will facilitate communication with the Intel XScale core to select a real server for the connection destination. If the connection is at Layer 7, then it will be passed on to the TCP network processor functions. Many counters on this processor are useful for identifying the type of traffic that is flowing through the system and the number of connections that are created, destroyed, timed out, etc.
- Outbound Connection Manager (OCM): The OCM establishes the connection to the destination for the client connections. The OCM is also used in TCP reuse, the creation of syslogs for established connections, and NAT pool source-address and source-port selection.
- Timers: This process runs on a dedicated core (core 0) on each network processor and handles all network processor-specific timers.

Monitoring Data Plane Use

The following command provides a good indication of the load of the network processor cores in terms of packet and connection processing:

```
ACE/Admin# show np # -cpu | beg Util
ME % Utilization Statistics
--------------
Core 0:                                          3
Core 1:                                          0
Core 2:                                          0
Core 3:                                          0
Core 4:                                          0
Core 5:                                          0
Core 6:                                          0
Core 7:                                          0
Core 8:                                          0
Core 9:                                          0
Core 10:                                         0
Core 11:                                         0
Core 12:                                         0
Core 13:                                         0
Core 14:                                         0
Core 15:                                         0
```

Note: All **show np** commands must be run for each of the network processors (one network processor in Cisco ACE 4710 and one to four network processors in the Cisco ACE30 Module). This output must be combined for all network processors on the device to calculate the total load.

The network processor cores operate safely at any percentage of utilization. A network processor core that approaches 100 percent utilization is an indication that the traffic load is stressing its limit. Any core that reaches 100 percent utilization can cause backpressure and may lead to dropped packets or, in the worst case, dropped connections.

Monitoring Backpressure

Backpressure occurs when a bottleneck develops as traffic performance is increased. Backpressure is the mechanism used to slow down the system when queues start to fill up internally because a microengine is reaching its peak utilization. This scenario can occur when incoming traffic exceeds the capacity of the Cisco ACE. As backpressure builds, the first-in, first-out (FIFO) queue for the CDE and internal queues for each data-plane function in the network processor cores can become filled. A sustained period of time in which the Cisco ACE is running at or above capacity can lead to overflows and dropped packets. When the Cisco ACE reaches capacity, new packets are dropped, though not necessarily entire connections. However, if the overflow continues, connections may be dropped by clients and servers.

- To monitor the CDE queues on the Cisco ACE module, use the show **cde health** command and verify that **Fifo Full drop count** is not incrementing.

```
ACE/Admin# show cde health | include Fifo
Fifo Full drop count                              0
```

- To monitor the Fastpath processing queues, use the **show np <1|2|3|4> me-stats "-s fp"** command and verify that **FastQ Transmit Backpressure, SlowQ Transmit Backpressure, Drop: Transmit Backpressure,** and **Drop: Next-Hop queue full** are not incrementing.

```
ACE/Admin# show np 1 me-stats "-s fp -v" | include Backpressure
FastQ Transmit Backpressure:                      0             0
SlowQ Transmit Backpressure:                      0             0
Hyperion Transmit Backpressure:                   0             0
Drop: Transmit Backpressure:                      0             0

ACE/Admin# show np 1 me-stats "-s fp -v" | include queue
Drop: Next-Hop queue full:                        0             0
```

- To monitor the TCP processing queues, use the **show np <1|2|3|4> me-stats "-s tcp**" command and verify **Drops due to FastTX queue full, Drops due to Fastpath queue full, Drops due to HTTP queue full, Drops due to SSL queue full, Drops due to AI queue full,** and **Drops due to Fixup queue full** are not incrementing. If TCP receives backpressure, it can drop packets, fail to acknowledge packets, and fail to properly track the next packet in the TCP connection.

```
ACE/Admin#  show np 1 me-stats "-s tcp -v" | include queue
Drop reproxy msg queue full:                      0             0
Drops due to FastTX queue full:                   0             0
Drops due to Fastpath queue full:                 0             0
Drops due to HTTP queue full:                     0             0
Drops due to SSL queue full:                      0             0
Drops due to AI queue full:                       0             0
Drops due to Fixup queue full:                    0             0
```

Note: **Drop reproxy msg queue full** will increment based on the incoming connections when the reproxy msg queue is full. Ideally, this queue should not increment. When it increments, clients will receive a connection reset message.

- To monitor the number of drops in all processor queues, use the general **me-stats** command parameter - **sdrop** as follows:

```
ACE/Admin# show np 1 me-stats -sdrop
Fastpath Statistics: (Current)
-------------------
Errors:                                        50844[1]
DROP: RX Interface miss:                        6103
DROP: Unknown Msg received:                      378
Packets dropped (encap invalid):                  32
(Context ALL Statistics)
```

[1] This counter in the Fastpath queue is usually incremented when a normalization failure occurs. The output of **sh np <x> me-stats "-s normalization"** will list the failure reasons.

```
DROP: Connection Route:                          1903

Receive Statistics: (Current)
------------------
Packet drops:                                     107

ICM Statistics (Current)
--------------
Errors:                                         58902[2]
Close Connection validation error:              29879
(Context ALL Statistics)
Route lookup Error:                              3845
MAC Lookup Error:                                 540

OCM Statistics: (Current)
--------------
(No relevant stats)
(Context ALL Statistics)
(No relevant stats)

HTTP Statistics (Current)
--------------
Buffers dropped:                              5098553[3]
Conn mismatch errors:                          553559[4]
```

A drop in HTTP, ICM, or the Fastpath queue can be the result of a malformed packet. However, these processor statistics may also be the result of processor overload or backpressure. It is normal to see these counters increase gradually over time; however, continuously increasing drops exceeding several hundred per second may warrant further investigation.

- To get a clearer understanding of why there are drops in ICM or similar modules, you can use the following command to show the systemwide global delta. The -**M1** option shows the change in value for each counter, and **ex " 0$"** excludes any counters that have not changed) since the last output. In the following code, the **Next-Hop queue full** counter is a subset of the Errors counter, and it is increasing rapidly. This output tells you that the next process in the flow, the load-balancing (LB) module, is the bottleneck.

```
ACE/Admin# show np 1 me-stats "-sicm -M1" | ex  " 0$"
ICM Statistics (Delta)
--------------
```

[2] ICM tried to remove a message from its input queue, and the message was of null type, which should not occur. ICM counts the attempt, skips this message, and waits for more input.
[3] This value is incremented by HTTP for a number of cases, including the normal case. For example, when the buffer chain is dropped normally, this counter is incremented. Additional error counters are incremented if the drop occurs because of an error.
[4] When this value increments without **Exception with close** also incrementing, a nonfatal error occurred in which a message for a proxy ID was received that had either been closed or reused with a new sequence number. This scenario can occur for several reasons, the most common being that the connection was closed while data was being received (not unusual since HTTP gives priority to Close messages). When this situation occurs, the message is simply dropped. It is fairly common to see this statistic increment during any kind of stress, and this behavior does not by itself indicate a problem.

```
Errors:                                  1517033
Frames Received:                         4778117
Close Receive:                           6490316
Drop [Next-Hop queue full]:              1517033
(Context ALL Statistics)
Transmit -> fastpath:                      26376
Transmit -> TCP:                         1187743
Transmit -> OCM:                               3
Send   -> LB_L4:                         2047293
Connection [Inserts]:                    4805238
Connection [Deletes]:                    5502666
Proxy [Deletes]:                         1010133
CP Init Received:                          26376
My mac check Error:                           51
Replicate connection sent:               6013821
```

- The system global queues can also be checked when you suspect drops are occurring. Checking these queues helps provide a view of the various modules' health overall. A high number of outstanding messages, represented by the third data column in the following results, can reveal which functions on the Cisco ACE data plane may be overloaded. Here, you can see that the LBRX, SSLXRX, and TCPHP queues have a number of messages outstanding. This result shows a backlog of connections waiting for load balancing and for SSL processing.

```
ACE/Admin# show np 1 me-stats -Q
Queue summary:
        lbrx 2878 2796 262103 ACTIVE
      lbrxhi 18288 18310 11 ACTIVE
      lbtome 3194 3194 0 EMPTY
      sslxrx 14352 854 248646 ACTIVE
      sslxtome 123 123 0 EMPTY
      ihmerx 0 0 0 EMPTY
      fasttx 248 248 0 EMPTY
          fp 0 10 0 ACTIVE
        fphi 63 63 0 EMPTY
        airx 0 0 0 EMPTY
         icm 1740 1740 0 EMPTY
      slowtx 3093 3093 0 EMPTY
       reass 2639 2639 0 EMPTY
       ocmlo 1810 1810 0 EMPTY
       ocmhi 0 0 0 EMPTY
       tcprx 1643 1643 0 EMPTY
       tcptx 4083 4083 0 EMPTY
      httprx 3411 3411 0 EMPTY
      httptx 3525 3525 0 EMPTY
     fixuprx 0 0 0 EMPTY
```

```
  fixuptx 4 4 0 EMPTY
  sslmerx 1227 1227 0 EMPTY
sslmerxhi 1527 1527 0 EMPTY
  sslmetx 2093 2093 0 EMPTY
  cmclose 639 639 0 EMPTY
   ipcplo 0 0 0 EMPTY
   ipcphi 2432 2432 0 EMPTY
  xtomelo 3430 3430 0 EMPTY
  xtomehi 371 371 0 EMPTY
    haxrx 3354 3354 0 EMPTY
     aitx 2189 2189 0 EMPTY
   syslog 0 0 0 EMPTY
    tcphp 18708 18626 262103[5] ACTIVE
 fasttxhi 30942 30952 5 ACTIVE
```

More information about the **me-stats** output and other counters is available at http://docwiki.cisco.com/wiki/Cisco_Application_Control_Engine_%28ACE%29_Troubleshooting_Guide_--_Show_Counter_Reference.

Note: The statistics shown here are available exclusively from the command-line interface (CLI) and XML web interface.

Monitoring the Control Plane

The control-plane processor processes all control traffic (Address Resolution Protocol [ARP], Hot Standby Router Protocol [HSRP], Internet Control Message Protocol [ICMP] to virtual IP addresses, routing, syslogs, Simple Network Management Protocol [SNMP], probes, etc.) and handles configuration management to parse the CLI for syntax errors and enforce configuration dependencies and requirements before pushing the configuration to the data plane.

A three-way moving average of control-plane processor utilization can be obtained using the **show** command:

```
ACE/Admin# show processes cpu | inc util
CPU utilization for five seconds: 81%; one minute: 15%; five minutes: 10%

0001-sfdi-ace-a/Admin# sho processes cpu | inc utiliz
CPU utilization for five seconds: 23%; one minute: 24%; five minutes: 24%
```

The update interval for the control-plane CPU utilization is fixed at five seconds. Thus, there is no benefit in tracking the interval time itself with SNMP; however, the object identifier (OID) is provided in the following results for completeness. The other intervals are averaged over their duration. Since the updates occur at 5-second intervals, there is little advantage in polling these OIDs more frequently. An increase in CPU utilization over time

[5] When the LBRX, SSLXRX, or TCPHP queues exceed 250,000 messages, outstanding traffic may be dropped, and after 262,000 messages, all packets are dropped. Any queue with more than 250,000 messages outstanding will exhibit significant packet loss. For the most reliable service, you should not exceed 200,000 messages in any queue.

can be clearly tracked using the average over a 5-minute interval. This approach provides general trending information and mitigates the effects of short-lived control-plane CPU spikes.

```
[root@mgmt mibs]# snmpget  -v 2c -M . -m ALL -c cisco 172.25.91.20
.1.3.6.1.4.1.9.9.109.1.1.1.1.9.1 2> /dev/null
CISCO-PROCESS-MIB::cpmCPUMonInterval.1 = Gauge32: 5 seconds


[root@mgmt mibs]# snmpget  -v 2c -M . -m ALL -c cisco 172.25.91.20
.1.3.6.1.4.1.9.9.109.1.1.1.1.10.1 2> /dev/null
CISCO-PROCESS-MIB::cpmCPUTotalMonIntervalValue.1 = Gauge32: 81 percent


 [root@mgmt mibs]# snmpget  -v 2c -M . -m ALL -c cisco 172.25.91.20
.1.3.6.1.4.1.9.9.109.1.1.1.1.7.1 2> /dev/null
CISCO-PROCESS-MIB::cpmCPUTotal1minRev.1 = Gauge32:15 percent


[root@mgmt mibs]# snmpget  -v 2c -M . -m ALL -c cisco 172.25.91.20
.1.3.6.1.4.1.9.9.109.1.1.1.1.8.1 2> /dev/null
CISCO-PROCESS-MIB::cpmCPUTotal5minRev.1 = Gauge32:10 percent


ACE/Admin# show version | inc free
    total: 956492 kB, free: 288268 kB
```

Monitoring Relevant Metrics

Different traffic patterns and configurations can stress different portions of the system and require different resources in the network processors in the module. For this reason, a single metric or CPU utilization value is not available to indicate the overall load on the system; however, you can monitor certain critical metrics to see whether the current traffic pattern is approaching any bottlenecks.

The following sections describe the main metrics for characterizing the load of the system and show how to check the relevant counters.

- Concurrent connections: The number of simultaneous connections that a device can support is a function of available memory. If the number of concurrent connections reaches the supported limit, no new connections can be established until existing connections are freed.
- Interconnects and bandwidth: The bandwidth that a device supports is based on the device's interconnection links to the network and the amount of time needed to process application traffic. Exceeding the bandwidth can lead to packet loss at the interlinks or within the device itself.
- Connections per second (CPS): CPS is the measure of the number of new client connections to an application within a second. A connection setup can be a simple TCP three-way handshake with an immediate TCP reset, or it can be more complex, such as an SSL operation in which TCP is set up, SSL is terminated, and an HTTP request is processed before the SSL session and TCP connection are closed properly. Although supported CPS values typically vary between application types, exceeding the limits will result in rejection of new connection attempts.

Concurrent Connections

The Cisco ACE allocates data-plane memory to help guarantee concurrent connection support for basic Layer 4 connections (such as TCP, User Datagram Protocol [UDP], and IP Security [IPsec]), Layer 7 connections (proxied flows, typically for application-aware load balancing or inspection), and SSL connections when using SSL acceleration. The Cisco ACE can support the maximum number of bidirectional concurrent connections allowed, regardless of the features enabled (Table 1).

Table 1. Concurrent Connection Support

Connection Type	Cisco ACE Appliance Limit	Cisco ACE Module Limit
Layer 4	1 million	4 million
Layer 7	128,000	512,000

The state for both directions (client to virtual IP address and Cisco ACE, and server to Cisco ACE) of a TCP connection is maintained through distinct connection objects. The following listing is an example of a current connection table:

```
ACE/Admin# show conn

total current connections : 6
conn-id    np dir proto vlan source                destination           state
----------+--+---+-----+----+---------------------+---------------------+------+
6          1  in  TCP   110  10.82.217.52:1566     172.25.91.20:23       ESTAB
7          1  out TCP   110  172.25.91.20:23       10.82.217.52:1566     ESTAB
1          1  in  TCP   222  209.165.201.22:2225   172.16.1.100:80       ESTAB
2          1  out TCP   422  192.168.1.20:80       209.165.201.22:2225   ESTAB
9          2  in  TCP   222  209.165.201.22:2256   172.16.1.100:80       ESTAB
10         2  out TCP   422  192.168.1.30:80       209.165.201.22:2256   ESTAB
```

Note: The **detail** parameter can be added to show the connection idle time, the elapsed time of the connection, the byte count, and the packet count for each connection object.

In this example, a connection has been opened from client 209.165.201.22, which connected to virtual IP address 172.16.1.100 on port 80 (`blue highlight`). The connection was received from VLAN 222 and sent to VLAN 422, to server 192.168.1.20, without rewriting the source IP address of the client (`green highlight`). The second line of the output (`green highlight`) indicates source and destination IP addresses and Layer 4 ports of the return traffic (server to client), which is why the client IP address now appears in the “destination” column.

The total number of concurrent connections per context is maintained in the **show stats** counters and in the SNMP OIDs:

```
ACE/Admin# show stats connection

+-----------------------------------------+
+------- Connection statistics ------------+
+-----------------------------------------+
 Total Connections Created  : 118
```

```
 Total Connections Current  : 6
 Total Connections Destroyed: 96
 Total Connections Timed-out: 20
 Total Connections Failed   : 0

[root@mgmt mibs]# snmpget  -v 2c -M . -m ALL -c cisco 172.25.91.20
.1.3.6.1.4.1.9.9.254.1.1.1.1.3.3  2> /dev/null
CISCO-SLB-EXT-MIB::cslbxStatsCurrConnections.3 = Gauge32: 6 connections
```

Note: The **Total Connections Current** counter counts the number of used connection objects, not the number of TCP flows. The number of TCP flows can be roughly calculated as half the number of connection objects minus any UDP connections.

The **Total Connections Current** counter is always up-to-date, and the maximum value can be 8 million on the Cisco ACE Module and 1 million on the Cisco ACE 4710 appliance.

Because of the Cisco ACE architecture, which provides distinct paths for new and established connections, the number of existing concurrent connections does not greatly affect the rate at which new connections can be set up. Nevertheless, a very large number of concurrent connections can eventually affect the performance of the system in setting up new connections when connection resources are completely consumed.

The number of concurrent connections is relevant when load-balancing protocols that maintain many low-bandwidth and long-lived connections, such as Telnet, TN3270, and terminal services sessions.

TCP connections that are properly terminated with RST or FIN/FIN_ACK/ACK are immediately removed (destroyed) from the concurrent connection table. If a TCP connection closes without being properly terminated, it is torn down when the TCP idle timers expire.

When deploying virtual IP addresses for load-balancing applications using UDP, adjust the idle timeout value to expedite the removal of UDP connections from the concurrent connections table. The default inactivity timeouts are defined by protocol:

- TCP: 3600 seconds
- HTTP/SSL: 300 seconds
- UDP: 10 seconds
- ICMP: 2 seconds

Depending on application requirements, the idle timeout value for UDP flows can often be set to the minimum value. UDP Fast Aging and UDP Boost can be added to further increase performance and optimize load balancing.

To see the number of total connections in the system, use the command **show resource usage summary**.

```
ACE/Admin# show resource usage summary | end proxy
  conc-connections              0          0          0    8000000          0
  mgmt-connections              2         92          0     100000          0
  proxy-connections             0          0          0    1048572          0
```

To see the number of active SSL connections on all the device's network processors combined, use the command **show stats crypto server termination**.

```
ACE/Admin# show stats crypto server termination | inc active
SSLv3 active connections:                          0
TLSv1 active connections:                          0
```

The number of connections removed from the connection table can be checked using the connection statistics. Connections removed in the nominal case, in which the connections are properly closed using the TCP FIN sequence or are forcibly closed using a TCP reset, are tracked using the **Total Connections Destroyed** counter (`blue highlight`). The connections that have been removed because one of the three types of idle timer (SYN, half open, or connection idle) expired are shown with the **Total connections Timed-out** counter (`green highlight`).

```
ACE/Admin# show stats connection

+-----------------------------------------+
+------- Connection statistics ------------+
+-----------------------------------------+
 Total Connections Created  : 126
 Total Connections Current  : 2
 Total Connections Destroyed: 96
 Total Connections Timed-out: 28
 Total Connections Failed   : 0
```

Connections can be timed out for the following reasons:

- Connection idle: One side of the connection (client to server or server to client) has not transmitted traffic for longer than the idle timeout period configured for the virtual IP address. Adjust the value using **set timeout inactivity** in the connection parameter map.
- Connection pending (embryonic) timeout: A TCP SYN message has been sent to the selected server, and Cisco ACE is awaiting a SYN/ACK response from the server so that the TCP three-way handshake can be completed. Adjust the value using **set tcp timeout embryonic** in the connection parameter map.
- Half closed: A TCP FIN message has been sent to the selected server, and Cisco ACE is awaiting a FIN/ACK response from the server so that the TCP three-way FIN sequence can be completed. Adjust the value using **set tcp timeout half-closed** in the connection parameter map.

Note: UDP uses only the connection idle timer. Since UDP is a stateless protocol, it does not have an initiation or closure handshake and thus does not require additional timeout checks.

If new connections reach the Cisco ACE when the connection table has no more room for new connections, the Cisco ACE resets the new connection and increments the **Drop [out of connections]:** counter:

```
ACE/Admin# show np 1 me-stats "-socm -vvv"| inc "of conn"
Drop [out of connections]:                         0
```

Note: Even when this counter increases, open connections are not affected.

Interconnects and Bandwidth

The Cisco ACE Module interconnects with the Cisco Catalyst® 6500 Series Switch backplane using a single switch fabric interface. 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 ACE Module to boot from the supervisor's storage if needed. The shared bus interconnect is used only if communication with classic cards is required. The Cisco ACE Module uses a single interconnect with the switching fabric to provide a 20-Gbps connection, to support up to 16 Gbps of throughput. Cisco IOS Software today does not have any 16G (16-Gbps) interfaces, so the TenGigabitEthernet (10 Gigabit Ethernet) type has been used to indicate that it is indeed a single connection, without any need for EtherChannel (as in case of the Cisco Content Switching Module [CSM]). Although some of the information provided by the **show int TenGigabitEthernet X/Y** options is not meaningful for services modules, the following command options provide useful information: capabilities, counters, status, switchport, and trunk.

```
Router# show int TenGigabitEthernet 3/1 status

Port     Name               Status       Vlan         Duplex  Speed Type
Te3/1                       connected    trunk          full    10G MultiService Module


Router# show int TenGigabitEthernet 3/1 counters

Port            InOctets    InUcastPkts    InMcastPkts    InBcastPkts
Te3/1          634859790        3554991        2014724        1887878

Port           OutOctets   OutUcastPkts   OutMcastPkts   OutBcastPkts
Te3/1           56939210         306341              0         310695

Router#show counters interface tenGigabitEthernet 3/1
64 bit counters:
0.                      rxHCTotalPkts = 7457593
 1.                     txHCTotalPkts = 617036
 2.                   rxHCUnicastPkts = 3554991
 3.                   txHCUnicastPkts = 306341
 4.                 rxHCMulticastPkts = 2014724
 5.                 txHCMulticastPkts = 0
 6.                 rxHCBroadcastPkts = 1887878
 7.                 txHCBroadcastPkts = 310695
 8.                         rxHCOctets = 634859790
 9.                         txHCOctets = 56939210
10.                rxTxHCPkts64Octets = 0
11.             rxTxHCPkts65to127Octets = 0
12.            rxTxHCPkts128to255Octets = 0
13.            rxTxHCPkts256to511Octets = 0
14.           rxTxHCpkts512to1023Octets = 0
```

```
15.         rxTxHCpkts1024to1518Octets = 0
16.                       txHCTrunkFrames = 0
17.                       rxHCTrunkFrames = 0
18.                        rxHCDropEvents = 0

32 bit counters:
 0.                     rxCRCAlignErrors = 0
 1.                     rxUndersizedPkts = 0
 2.                      rxOversizedPkts = 0
 3.                       rxFragmentPkts = 0
 4.                            rxJabbers = 0
 5.                          txCollisions = 0
 6.                            ifInErrors = 0
 7.                           ifOutErrors = 0
 8.                          ifInDiscards = 0
 9.                    ifInUnknownProtos = 0
10.                         ifOutDiscards = 0
11.              txDelayExceededDiscards = 0
12.                                txCRC = 0
13.                           linkChange = 0
14.                     wrongEncapFrames = 0
All Port Counters
 1.                             InPackets = 7457593
 2.                              InOctets = 634859790
 3.                            InUcastPkts = 3554991
 4.                            InMcastPkts = 2014724
 5.                            InBcastPkts = 1887878
 6.                             OutPackets = 617036
 7.                              OutOctets = 56939210
 8.                           OutUcastPkts = 306341
 9.                           OutMcastPkts = 0
10.                           OutBcastPkts = 310695
11.                               AlignErr = 0
12.                                 FCSErr = 0
13.                                XmitErr = 0
14.                                 RcvErr = 0
15.                              UnderSize = 0
16.                              SingleCol = 0
17.                               MultiCol = 0
18.                                LateCol = 0
19.                           ExcessiveCol = 0
```

The connection to the Cisco Catalyst switch backplane is indeed 16-Gbps full-duplex; however, note that most traffic will traverse this connection twice.

For the Cisco ACE 4710, statistics can be collected directly from the individual physical ports on the appliance. The Cisco ACE 4710 interconnects with a Cisco Catalyst switch either through a single dedicated physical port or through all four ports in an EtherChannel. When an EtherChannel is used, the Cisco ACE 4710 interconnect provides up to 4 Gbps of bandwidth. The following command options provide useful information: capabilities, counters, status, switchport, and trunk.

```
4710b/Admin# sh int gi1/1 counters
GigabitEthernet Port 1/1 Counters:
-------------------------------------
RX RGMII Packets: 103090411
RX RGMII Control Packets: 0
RX RGMII DMAC filtered Packets: 0
RX RGMII Dropped Packets: 0
RX RGMII Bad Packets: 0

RX RGMII Octets: 7437725842
RX RGMII Control Octets: 0
RX RGMII DMAC filtered Octets: 0
RX RGMII Dropped Octets: 0

RX packets : 103090411
RX octets : 7437725842
RX dropped Packets : 0
RX broadcasts : 1743676
RX multicasts : 100947192
RX runts : 0
RX giants : 0
RX FCS/Align Errors : 0
RX runt FCS : 0
RX giant FCS : 0
Total inbound packets : 0
Total inbound octets : 0
Total inbound errors : 0
TX Packets: 1662627
TX Octets: 115707100
TX Broadcast Packets: 1651262
TX Multicast Packets: 2
TX Control Packets: 0
TX Underflow Packets: 0
TX Single Collision Packets: 0
TX Multiple Collision Packets: 0
TX Excessive Collisions and Dropped Packets: 0
TX Excessive Deferral and Dropped Packets: 0
```

```
4710b/Admin# sh int | beg ^Giga
GigabitEthernet Port 1/1 is UP, line protocol is UP
 Hardware is ACE Appliance 1000Mb 802.3, address is 00:1b:24:5e:8d:43
 Description:
 MTU 9216 bytes
 Full-duplex, 1000Mb/s
 COS bits based QoS is disabled
 input flow-control is off, output flow-control is off
    103100458 packets input, 7438449090 bytes, 0 dropped
    Received 1743826 broadcasts (100956876 multicasts)
    0 runts , 0 giants
    0 FCS/Align errors , 0 runt FCS, 0 giant FCS
    1662785 packets output, 115717528 bytes
    1651420 broadcast, 2 multicast, 0 control output packets
    0 underflow, 0 single collision, 0 multiple collision output packets
    0 excessive collision and dropped, 0 Excessive Deferral and dropped
GigabitEthernet Port 1/2 is UP, line protocol is UP
 Hardware is ACE Appliance 1000Mb 802.3, address is 00:1b:24:5e:8d:43
 Description:
 MTU 9216 bytes
 Full-duplex, 1000Mb/s
 COS bits based QoS is disabled
 input flow-control is off, output flow-control is off
    15730013 packets input, 1184901444 bytes, 0 dropped
    Received 13563385 broadcasts (847967 multicasts)
    0 runts , 0 giants
    0 FCS/Align errors , 0 runt FCS, 0 giant FCS
    1660444 packets output, 113144020 bytes
    1651418 broadcast, 0 multicast, 0 control output packets
    0 underflow, 0 single collision, 0 multiple collision output packets
    0 excessive collision and dropped, 0 Excessive Deferral and dropped
GigabitEthernet Port 1/3 is UP, line protocol is UP
 Hardware is ACE Appliance 1000Mb 802.3, address is 00:1b:24:5e:8d:43
 Description:
 MTU 9216 bytes
 Full-duplex, 1000Mb/s
 COS bits based QoS is disabled
 input flow-control is off, output flow-control is off
    3803672 packets input, 330213841 bytes, 0 dropped
    Received 2524303 broadcasts (942447 multicasts)
    0 runts , 0 giants
    0 FCS/Align errors , 0 runt FCS, 0 giant FCS
    2395769 packets output, 251131685 bytes
    1651419 broadcast, 0 multicast, 0 control output packets
```

```
    0 underflow, 0 single collision, 0 multiple collision output packets
    0 excessive collision and dropped, 0 Excessive Deferral and dropped
GigabitEthernet Port 1/4 is UP, line protocol is UP
 Hardware is ACE Appliance 1000Mb 802.3, address is 00:1b:24:5e:8d:43
 Description:
 MTU 9216 bytes
 Full-duplex, 1000Mb/s
 COS bits based QoS is disabled
 input flow-control is off, output flow-control is off
    5333725 packets input, 395167371 bytes, 0 dropped
    Received 2602246 broadcasts (849170 multicasts)
    0 runts , 0 giants
    0 FCS/Align errors , 0 runt FCS, 0 giant FCS
    1670198 packets output, 114360686 bytes
    1659792 broadcast, 0 multicast, 0 control output packets
    0 underflow, 0 single collision, 0 multiple collision output packets
    0 excessive collision and dropped, 0 Excessive Deferral and dropped
```

When an EtherChannel is configured for the Cisco ACE 4710, the combined statistics are available as follows:

```
4710b/Admin# sh int po2

PortChannel 2:
----------------------------
Description:
mode: Trunk
native vlan: 0
status: (UP), load-balance scheme: src-dst-port

PortChannel 2 mapped phyport: 1/1 1/2 1/3 1/4
PortChannel 2 mapped active phyport: 1/1 1/2 1/3 1/4
PortChannel 2 allow vlan:  vlan<110>  vlan<703>-<799>
    127984065 packets input, 9349913787 bytes, 0 dropped
    Received 20436446 broadcasts (103609810 multicasts)
    0 runts , 0 giants
    0 FCS/Align errors , 0 runt FCS, 0 giant FCS
    7390181 packets output, 594433471 bytes
    6614903 broadcast, 2 multicast, 0 control output packets
    0 underflow, 0 single collision, 0 multiple collision output packets
    0 excessive collision and dropped, 0 Excessive Deferral and dropped
```

When estimating how much traffic can be sent across Cisco ACE, consider the following important factors:

- Although the connection between Cisco ACE and the backplane is full duplex, most packets have to traverse that connection twice (once from the input interface to the Cisco ACE, and once from the Cisco ACE to the output interface). Therefore, a packet sent from a client to a server (or from a server to a client) that traverses Cisco ACE will travel twice over that connection. For example, with a 4-Gbps license, the maximum bandwidth through Cisco ACE should account for both client-to-server traffic and server-to-client traffic, and their sum cannot exceed 4 Gbps. Applied practically, in an environment in which server-to-client traffic represents 90 percent and client-to-server traffic represents 10 percent, the maximum for client-to-server traffic will be 0.4 Gbps, and the maximum for server-to-client traffic will be 3.6 Gbps. If the environment is changed to achieve an ideal case of complete symmetry (50 percent of the traffic in each direction), then the maximum would be 2 Gbps each way.

Note: A more exact calculation should also consider traffic consumed or generated by Cisco ACE itself (keepalives, for example), but that traffic is typically a very small percentage compared to client-to-server and server-to-client traffic.

- The exact bandwidth depends on the type of traffic: in particular, the average size of packets. Check the other Cisco ACE scaling documents for more information about how four Cisco ACE Modules in the same chassis can scale to more than 60 Gbps.

The amount of traffic that can be sent depends on the average packet size of the traffic. Lab performance tests indicate that an average packet size of 512 bytes is enough to reach line rate.

You can use the following **show** command to see how much aggregate traffic a Cisco Catalyst switch is sending to and receiving from a Cisco ACE 4710 when connected with a PortChannel:

```
Router#show int Po260
Port-channel260 is up, line protocol is up (connected)
  Hardware is EtherChannel, address is 0002.fce1.65cb (bia 0002.fce1.65cd)
  MTU 9216 bytes, BW 4000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Unknown duplex, Unknown Speed, link type is auto, media type is Gbic not
connected
  output flow-control is unsupported, input flow-control is unsupported
  Members in this channel: Gi4/1 Gi4/2 Gi4/3 Gi4/4
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input never, output never, output hang never
  Last clearing of "show interface" counters 1d01h
  Input queue: 0/2000/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 39000 bits/sec, 35 packets/sec
  5 minute output rate 44000 bits/sec, 45 packets/sec
     60775 packets input, 5904256 bytes, 0 no buffer
     Received 13499 broadcasts (0 multicast)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 0 multicast, 0 pause input
```

```
0 input packets with dribble condition detected
692465 packets output, 52851308 bytes, 0 underruns
0 output errors, 0 collisions, 0 interface resets
0 babbles, 0 late collision, 0 deferred
0 lost carrier, 0 no carrier, 0 PAUSE output
0 output buffer failures, 0 output buffers swapped out
```

The counters shown here are from the perspective of a Cisco Catalyst 6500 Series Switch; therefore, the input counters refer to traffic received from the Cisco ACE, and the output counters refer to traffic sent to the Cisco ACE.

Connections per Second

CPS is one of the hardest metrics to measure, since there is no hard limit. CPS performance heavily depends on configured features and traffic patterns.

The main factor that influences CPS performance is the use of Layer 4 or Layer 7 processing. Layer 4 performance is higher than Layer 7 performance, since Layer 4 balancing decisions are based on the first packet of each connection, whereas Layer 7 balancing requires TCP termination, multipacket buffering, and parsing to make the balancing decision and set up a new connection.

You can use the following **show** command to see how many Layer 4 and Layer 7 connections have been set up by the Cisco ACE:

```
ACE/Admin# show stats loadbalance

+----------------------------------------+
+------- Loadbalance statistics -----------+
+----------------------------------------+
 Total version mismatch              : 0
 Total Layer4 decisions              : 2714
 Total Layer4 rejections             : 476
 Total Layer7 decisions              : 1388667
 Total Layer7 rejections             : 2164
 Total Layer4 LB policy misses       : 0
 Total Layer7 LB policy misses       : 0
 Total times rserver was unavailable : 22
 Total ACL denied                    : 0
```

Note: In a steady-state system, when no configuration is taking place, continuous incrementing of any of these counters should raise a flag.

The Cisco ACE has many other device- and connection-specific metrics than just the ones discussed here. For more information, please visit the Cisco DocWiki at http://docwiki.cisco.com/wiki/Cisco_Application_Control_Engine_%28ACE%29_Troubleshooting_Guide.

Monitoring Remote Devices

The Cisco ACE lets you access an abundance of information remotely, providing both devicewide and context-specific statistics. You can monitor Cisco ACE by using SNMP and by using the web XML interface. Every CLI command equivalent is available using the XML interface. The following remote command examples illustrate the monitoring and provisioning capabilities of the Cisco ACE with any third-party or custom network management application.

Device Availability

The state of the module can be monitored by SNMP with the following MIB:

```
MIB:      ciscoEntityFRUControlMIB
Object:   cefcModuleOperStatus
OID:            1.3.6.1.4.1.9.9.117.1.2.1.1.2


Possible Return Values:
1:unknown
2:ok
3:disabled
14:outOfServiceEnvTemp
15:poweredDown
```

Fault Tolerance

The command **show ft group detail** provides information about the state of each module.

My State: Displays the state of the local Cisco ACE

Peer State: Displays the state of the remote Cisco ACE

A state of FSM_FT_STATE_ACTIVE tells you that the Cisco ACE Module is the active Cisco ACE.

A state of FSM_FT_STANDBY_HOT tells you that the Cisco ACE Module is the standby Cisco ACE, and that the configurations are synchronized.

Fault-tolerance information can be monitored through the following MIB:

```
MIB:      CISCO-L4L7MODULE-REDUNDANCY-MIB
Object:   ciscoL4L7moduleRedundancyMIB
OID:            1.3.6.1.4.1.9.9.650


Description:
This MIB provides details about the fault tolerant statistics available in the
show ft peer, show ft group detail and show ft stats command output.
```

XML: Configuration

The running configuration can be monitored through the XML agent:

```
xml_cmd=<request_raw>show running-config</request_raw>

<response_xml>
<exec_command>
<command>
show running-config
</command>
<status code="100" text="XML_CMD_SUCCESS"/>
<xml_show_result>

   Output removed for brevity

</xml_show_result>
</exec_command>
</response_xml>
```

Load-Balancing Services

Real Servers

When monitoring real servers (rservers) individually, you use the following OID:

```
MIB:      ciscoEnhancedSlbMIB
Object:   ciscoEnhancedSlbMIBNotifs
OID:            1.3.6.1.4.1.9.9.470.0


Possible Return Values:
4:  cesRserverStateUp
5:  cesRserverStateDown
6:  cesRserverStateChange

Descriptions:
'cesRserverStateUp':   This notification is generated when the real server
identified in cesRserverTable changes state to 'inservice' by the user
intervention
'cesRserverStateDown': This notification is generated when the real server
identified in cesRserverTable changes to 'outOfService' state by the user
intervention
'cesRserverStateChange':      This notification generated when the real server
identified in cesRserverTable changes to a new state other than that is initiated
by the user
```

Server Farm Rserver (Rserver Instance in a Server Farm)

When monitoring an rserver that is part of a server farm (which is referred to as a rserver), you use the following MIBs:

MIB: ciscoEnhancedSlbMIB
Object: ciscoEnhancedSlbMIBNotifs
OID: 1.3.6.1.4.1.9.9.470.0

Possible Return Values:
1: cesRealServerStateUp
2: cesRealServerStateDown
3: cesRealServerStateChange

Descriptions:
'cesRealServerStateUp': This notification is generated when a real server changes to 'inservice' state by the user intervention
'cesRealServerStateDown': This notification is generated when a real server changes to 'outOfService' state by the user intervention
'cesRealServerStateChange': This notification generated when a real server changes to a new state other than that is initiated
by the user

Virtual Servers

Monitor virtual servers as follows:

MIB: ciscoSlbMIB
Object: ciscoSlbMIBNotifications
OID: 1.3.6.1.4.1.9.9.161.2.0

Possible Return Values:
5: ciscoSlbVServerStateChange
6: ciscoSlbVServerVIPStateChange

Descriptions:
'ciscoSlbVServerStateChange': This notification is sent when a virtual IP address (VIP) is removed from a class map. This notification is sent with the following var-binds:
•slbVServerState
•slbVServerStateChangeDescr
•slbVServerClassMap
•slbVServerPolicyMap
The ciscoSlbVServerStateChange is specified in the CISCO-SLB-MIB.

'ciscoSlbVServerVIPStateChange': The state of Vserver changes. This notification is sent with the following var-binds:

- slbVServerState
- slbVServerStateChangeDescr
- slbVServerClassMap
- slbVServerPolicyMap
- slbVServerIpAddressType
- slbVServerIpAddress
- slbVServerProtocol

The change in the Vserver state could be due to a number of different reasons, such as binding to the interface, removing an active serverfarm from the policy, and associating the virtual IP address (VIP) with a class map.

The ciscoSlbVServerVIPStateChange is specified in the CISCO-SLB-MIB.

XML: Show Serverfarm

Server farms can be monitored with the XML agent. The output of this command can be parsed for greater detail.

```
xml_cmd=<request_raw>show serverfarm SERVERFARM1</request_raw>
<response_xml>
<exec_command>
<command>
show serverfarm SERVERFARM1
</command>
<status code="100" text="XML_CMD_SUCCESS"/>
<xml_show_result>
 serverfarm     : SERVERFARM1, type: HOST
 total rservers : 1
 ---------------------------------
                                                  ----------connections-----------
       real                  weight state         current    total      failures
   ---+--------------------+------+------------+----------+----------+---------
   rserver: SERVERFARM1
       161.44.52.238:5060    8      OPERATIONAL  0          0          0

</xml_show_result>
</exec_command>
</response_xml>

xml_cmd=<request_raw>show serverfarm SERVERFARM1 detail</request_raw>
<response_xml>
<exec_command>
<command>
show serverfarm SERVERFARM1 detail
</command>
<status code="100" text="XML_CMD_SUCCESS"/>
<xml_show_result>
 serverfarm     : SERVERFARM1, type: HOST
```

```
 total rservers : 1
 active rservers: 1
 description    : -
 state          : ACTIVE
 predictor      : ROUNDROBIN
 failaction     : -
 back-inservice    : 0
 partial-threshold : 0
 num times failover       : 0
 num times back inservice : 0
 total conn-dropcount : 0
 ---------------------------------
                                                  ----------connections-----------
      real                   weight state         current    total      failures
  ---+--------------------+------+------------+----------+----------+---------
  rserver: SERVERFARM1
      161.44.52.238:5060    8      OPERATIONAL  0          0          0
        max-conns            : -         , out-of-rotation count : -
        min-conns            : -
        conn-rate-limit      : -         , out-of-rotation count : -
        bandwidth-rate-limit : -         , out-of-rotation count : -
        retcode out-of-rotation count : -


</xml_show_result>
</exec_command>
</response_xml>
```

Context Resources Monitoring

Resource monitoring is useful for proactively viewing current resource use and planning for additional resources and upgrades if necessary.

Concurrent Connections

```
MIB:      ciscoL4L7moduleResourceLimitMIB
Object:   ciscoL4L7ResourceLimitNotifs
OID:            1.3.6.1.4.1.9.9.480.0


Possible Return Values:
1: clrResourceLimitReached
2: clrResourceRateLimitReached


Descriptions:
'clrResourceLimitReached':    This notification is generated when the configured
resource limit value specified in 'crlResourceLimitMax' is reached for a
```

```
particular resource. This resource is identified by 'crlResourceLimitType' in
ciscoL4L7ResourceLimitTable


CiscoResourceLimitType
1:all
2:macAddresses
3:concurrentConns
4:mgmtConnections
5:proxyConns
6:probes
7:stickyEntries
8:natTranslations
9:regexState
10:aclMemory
11:syslogBuffer
12:ipReassemBuffer
13:tcpOOOBuffer
14:sslConnections
15:hosts
16:ipsecSessions
17:asdmSessions
18:sshSessions
19:telnetSessions
```

Concurrent Connections Not Using Server Load Balancing

Concurrent connections for traffic that is not load balanced also can be reported with the following SNMP OIDs and MIB:

MIB: CISCO-SLB-EXT-MIB

- cslbxStatsCurrConnections
- cslbxStatsTimedOutConnections

XML: Context Resource Use

Resources per context can be monitored through the XML web interface. The output of the following XML command can be parsed for chargeback purposes, performance, and usability:

```
xml_cmd=<request_raw>show resource usage</request_raw>
<response_xml>
<exec_command>
<command>
show resource usage
</command>
<status code="100" text="XML_CMD_SUCCESS"/>
<xml_show_result>
                                                    Allocation
        Resource         Current       Peak        Min        Max     Denied
-------------------------------------------------------------------------------
```

```
Context: Admin
  conc-connections              9          9          0    6400000          0
  mgmt-connections             10         26          0       5000          0
  proxy-connections             0          0          0     524286          0
  xlates                        0          0          0    1048574          0
  bandwidth                   636      79201          0 2000000000        410
  connection rate               2         10          0    1000000          0
  ssl-connections rate          0          0          0      20000          0
  mgmt-traffic rate           198      32393          0  125000000          0
  mac-miss rate                 0          0          0       2000          0
  inspect-conn rate             0          0          0       6000          0
  acl-memory                14848      17504          0   47166260          0
  regexp                        0          0          0    1048576          0
  syslog buffer             11264      11264          0    4194304          0
  syslog rate                   0          8          0     100000          0
Context: ACS-AMEX
  conc-connections              0          0          0    6400000          0
  mgmt-connections              0          0          0       5000          0
  proxy-connections             0          0          0     524286          0
  xlates                        0          0          0    1048574          0
  bandwidth                     0          0          0 2000000000          0
  connection rate               0          0          0    1000000          0
  ssl-connections rate          0          0          0      20000          0
  mgmt-traffic rate             0          0          0  125000000          0
  mac-miss rate                 0          0          0       2000          0
  inspect-conn rate             0          0          0       6000          0
  acl-memory                    0          0          0   47166260          0
  regexp                        0          0          0    1048576          0
  syslog buffer                 0          0          0    4194304          0
  syslog rate                   0          0          0     100000          0
Context: ACS-TBO
  conc-connections              0          0          0    6400000          0
  mgmt-connections              0          0          0       5000          0
  proxy-connections             0          0          0     524286          0
  xlates                        0          0          0    1048574          0
  bandwidth                     0       1104          0 2000000000          0
  connection rate               0         12          0    1000000          0
  ssl-connections rate          0          0          0      20000          0
  mgmt-traffic rate             0          0          0  125000000          0
  mac-miss rate                 0          0          0       2000          0
  inspect-conn rate             0          0          0       6000          0
  acl-memory                 2552       2616          0   47166260          0
  regexp                        0          0          0    1048576          0
  syslog buffer                 0          0          0    4194304          0
  syslog rate                   0          0          0     100000          0
```

```
Context: BAIKAL
  conc-connections              0         10          0    6400000          0
  mgmt-connections              0         12          0       5000          0
  proxy-connections             0          8          0     524286          0
  xlates                        0          0          0    1048574          0
  bandwidth                    80       3044          0 2000000000          0
  connection rate               1         13          0    1000000          0
  ssl-connections rate          0          0          0      20000          0
  mgmt-traffic rate             0       1142          0  125000000          0
  mac-miss rate                 0          7          0       2000          0
  inspect-conn rate             0          0          0       6000          0
  acl-memory                 8240       8240          0   47166260          0
  regexp                        0          0          0    1048576          0
  syslog buffer                 0          0          0    4194304          0
  syslog rate                   0          0          0     100000          0
Context: Paul
  conc-connections              0         28          0    6400000          0
  mgmt-connections              2         10          0       5000          0
  proxy-connections             0          5          0     524286          0
  xlates                        0          0          0    1048574          0
  bandwidth                   100    1435341          0 2000000000          0
  connection rate               1         15          0    1000000          0
  ssl-connections rate          0          0          0      20000          0
  mgmt-traffic rate             0       3759          0  125000000          0
  mac-miss rate                 0          6          0       2000          0
  inspect-conn rate             0          0          0       6000          0
  acl-memory                 7680       7920          0   47166260          0
  regexp                        0          0          0    1048576          0
  syslog buffer                 0          0          0    4194304          0
  syslog rate                   0         16          0     100000          0
Context: batch5
  conc-connections              0          0          0    6400000          0
  mgmt-connections              0          0          0       5000          0
  proxy-connections             0          0          0     524286          0
  xlates                        0          0          0    1048574          0
  bandwidth                     0          0          0 2000000000          0
  connection rate               0          0          0    1000000          0
  ssl-connections rate          0          0          0      20000          0
  mgmt-traffic rate             0          0          0  125000000          0
  mac-miss rate                 0          0          0       2000          0
  inspect-conn rate             0          0          0       6000          0
  acl-memory                50880      50880          0   47166260          0
  regexp                        0          0          0    1048576          0
  syslog buffer                 0          0          0    4194304          0
  syslog rate                   0          0          0     100000          0
```

```
Context: batch6
  conc-connections              0          0          0    6400000          0
  mgmt-connections              0          0          0       5000          0
  proxy-connections             0          0          0     524286          0
  xlates                        0          0          0    1048574          0
  bandwidth                     0          0          0 2000000000          0
  connection rate               0          0          0    1000000          0
  ssl-connections rate          0          0          0      20000          0
  mgmt-traffic rate             0          0          0  125000000          0
  mac-miss rate                 0          0          0       2000          0
  inspect-conn rate             0          0          0       6000          0
  acl-memory               134144     134144          0   47166260          0
  regexp                        0          0          0    1048576          0
  syslog buffer                 0          0          0    4194304          0
  syslog rate                   0          0          0     100000          0
Context: dcnteam
  conc-connections              0          0    1600000    6400000          0
  mgmt-connections              0          0          0       5000          0
  proxy-connections             0          0          0     524286          0
  xlates                        0          0          0    1048574          0
  bandwidth                     0          0          0 2000000000          0
  connection rate               0          0          0    1000000          0
  ssl-connections rate          0          0          0      20000          0
  mgmt-traffic rate             0          0          0  125000000          0
  mac-miss rate                 0          0          0       2000          0
  inspect-conn rate             0          0          0       6000          0
  acl-memory                 1008       1008   31444174          0          0
  regexp                        0          0          0    1048576          0
  syslog buffer                 0          0          0    4194304          0
  syslog rate                   0          0          0     100000          0
Context: skinny
  conc-connections              0          0          0    6400000          0
  mgmt-connections              0          0          0       5000          0
  proxy-connections             0          0          0     524286          0
  xlates                        0          0          0    1048574          0
  bandwidth                     0          0          0 2000000000          0
  connection rate               0          0          0    1000000          0
  ssl-connections rate          0          0          0      20000          0
  mgmt-traffic rate             0          0          0  125000000          0
  mac-miss rate                 0          0          0       2000          0
  inspect-conn rate             0          0          0       6000          0
  acl-memory                    0          0          0   47166260          0
  regexp                        0          0          0    1048576          0
  syslog buffer                 0          0          0    4194304          0
  syslog rate                   0          0          0     100000          0
```

```
Context: C2
  conc-connections              0          4          0    6400000          0
  mgmt-connections              0          0          0       5000          0
  proxy-connections             0          4          0     524286          0
  xlates                        0          0          0    1048574          0
  bandwidth                     0        596          0 2000000000          0
  connection rate               0          4          0    1000000          0
  ssl-connections rate          0          2          0      20000          0
  mgmt-traffic rate             0          0          0  125000000          0
  mac-miss rate                 0          0          0       2000          0
  inspect-conn rate             0          0          0       6000          0
  acl-memory                 4528       4528          0   47166260          0
  regexp                       24         24          0    1048576          0
  syslog buffer                 0          0          0    4194304          0
  syslog rate                   0          0          0     100000          0
</xml_show_result>
</exec_command>
</response_xml>
```

TCL Script Examples

The following sample TCL script will connect to the Cisco ACE Admin context through the management interface, get a list of all contexts, and get a running configuration for each context and write that output to a file called <context>.cfg. This script uses HTTP. If you need secure transfer, you can use HTTPS instead. You will need to create a user with the role of Network-monitor to use to log in through the XML interface.

Keep in mind that a management policy must allow either HTTP or HTTPS:

```
class-map type management match-any remote-access
  match protocol http any
  match protocol https any
```

The following TCL scripts for resource and SSL use reporting abort during an error condition and print a short message. You may want to add supplemental code to send an email, SNMP trap, etc.

XML Script Example: Show Connection Count

```
#!/usr/bin/tclsh

# Global login credentials
set user "username"
set pass "password"

# Global mgmt IP address of Admin context
set mgmt_ip "xx.xx.xx.xx"

# this procedure will execute a curl command to send the XML
# command to ACE.  If the command fails to execute properly,
# the script will exit with and error.
```

```
# If the command executes with no problem, then the output
# of the XML command is returned
proc issue_command { cmd } {

    global user pass mgmt_ip

    if { [catch {set output [exec -- curl -s\
                 http://${user}:${pass}@${mgmt_ip}/bin/xml_agent\
                -d "xml_cmd=<request_raw>${cmd}</request_raw>"] } error] } {
   puts "Problem with exec: $error"
   exit 0
    }
    return $output
}

# Get a list of contexts from Admin context
set contexts [split [issue_command "show context | inc Name"] \n]

foreach line $contexts {
    if { [regexp {Name: (.*) ,} $line - context] } {
   puts "Getting config for $context"
   set out [issue_command "changeto $context\nshow conn count"]
   if { [regexp {<xml_show_result>(.*)</xml_show_result>} $out - running_config]}
{
       set fp [open "${context}.cfg" w+]
       puts $fp $running_config
       close $fp
   }
    }
}
```

XML Script Example: Show Resource Use

```
#!/usr/bin/tclsh

# Global login credentials
set user "username"
set pass "password"

# Global mgmt IP address of Admin context
set mgmt_ip "xx.xx.xx.xx"

# this procedure will execute a curl command to send the XML
# command to ACE.  If the command fails to execute properly,
# the script will exit with and error.
# If the command executes with no problem, then the output
```

```
# of the XML command is returned
proc issue_command { cmd } {
    global user pass mgmt_ip
    if { [catch {set output [exec -- curl -s\
                 http://${user}:${pass}@${mgmt_ip}/bin/xml_agent\
                -d "xml_cmd=<request_raw>${cmd}</request_raw>"] } error] } {
   puts "Problem with exec: $error"
   exit 0
    }
    return $output
}

# Max of 80%
set max_percent ".8"

# Get a list of contexts from Admin context
set contexts [split [issue_command "show context | inc Name"] \n]

# list of resources to check (see below for more rate selection possibilities)
# Ace-Mod3/Admin# sho resource usage context dcnteam resource rate  ?
#   bandwidth        Show bandwidth in bytes per second
#   connections      Show connections per second
#   inspect-conn     Show rtsp/ftp inspect connections per second
#   mac-miss         Show mac miss traffic punted to CP packets per second
#   mgmt-traffic     Show management traffic bytes per second
#   ssl-connections  Show SSL connection rate
#   syslog           Show syslog messages per second
set resources {bandwidth connections ssl-connections}

foreach line $contexts {
    if { [regexp {Name: (.*) ,} $line - context] } {
   puts "\n -- Checking resource usage for $context --"
   foreach resource $resources {

# Output Example for parsing
# Ace-Mod3/Admin# sho resource usage context dcnteam resource rate  ssl-
connections
#                                                    Allocation
#         Resource         Current       Peak        Min        Max       Denied
# -------------------------------------------------------------------------------
# Context: dcnteam
#   ssl-connections rate          0          0          0      20000          0

       set usage($resource) [issue_command \
```

```
                    "show resource usage context $context resource rate
$resource"]
       if { [regexp {\s+\S+\s+(\d+)\s+(\d+)\s+(\d+)\s+(\d+)\s+(\d+)}\
                   $usage($resource) - current peak min max denied] } {

          # at this point, each value (Current, Peak, Min, Max, and Denied)
          # are in the appropriately named variable (current, peak, min,
          # max, and denied)

          # determine is peak is greater than 80% of Max
          if { $peak > [expr {$max * $max_percent}]} {
              puts "\tPROBLEM: Peak ($peak) is 80% of Max ($max)\
                           for resource $resource"

              # at this point, should send an alert to let someone know
              # there is a problem
          } else {
              puts "\t$resource: OK ($peak)"
          }
       }
   }
    }
}

Sample Output:

./check_resources

 -- Checking resource usage for Admin --
       bandwidth: OK (1166)
       connections: OK (1)
       ssl-connections: OK (0)

 -- Checking resource usage for ACS-AMEX --
       bandwidth: OK (0)
       connections: OK (0)
       ssl-connections: OK (0)

 -- Checking resource usage for ACS-TBO --
       bandwidth: OK (0)
       connections: OK (0)
       ssl-connections: OK (0)

 -- Checking resource usage for BAIKAL --
       bandwidth: OK (356)
```

```
        connections: OK (2)
        ssl-connections: OK (0)

    -- Checking resource usage for C2 --
        bandwidth: OK (0)
        connections: OK (0)
        ssl-connections: OK (0)

    -- Checking resource usage for Paul --
        bandwidth: OK (100)
        connections: OK (1)
        ssl-connections: OK (0)

    -- Checking resource usage for batch5 --
        bandwidth: OK (0)
        connections: OK (0)
        ssl-connections: OK (0)

    -- Checking resource usage for batch6 --
        bandwidth: OK (0)
        connections: OK (0)
        ssl-connections: OK (0)

    -- Checking resource usage for dcnteam --
        bandwidth: OK (0)
        connections: OK (0)
        ssl-connections: OK (0)

    -- Checking resource usage for skinny --
        bandwidth: OK (0)
        connections: OK (0)
        ssl-connections: OK (0)
```

XML Script Example: Check for SSL Certificate Expiration

```
#!/usr/bin/tclsh

# Global login credentials
set user "username"
set pass "password"

# Global mgmt IP address of Admin context
set mgmt_ip "xx.xx.xx.xx"

#
# this procedure will execute a curl command to send the XML
```

```
# command to ACE.  If the command fails to execute properly,
# the script will exit with and error.
# If the command executes with no problem, then the output
# of the XML command is returned
proc issue_command { cmd } {

    global user pass mgmt_ip

    if { [catch {set output [exec -- curl -s\
                 http://${user}:${pass}@${mgmt_ip}/bin/xml_agent\
                -d "xml_cmd=<request_raw>${cmd}</request_raw>"] } error] } {
   puts "Problem with exec: $error"
   exit 0
    }

    return $output
}

set current_date [clock seconds]

# Get a list of contexts from Admin context
set contexts [split [issue_command "show context | inc Name"] \n]

foreach line $contexts {

    if { [regexp {Name: (.*) ,} $line - context] } {

   # output example
   # Ace-Mod3/Paul# sho crypto files | inc CERT
   # test-cert.pem                             787   PEM     Yes        CERT

   set crypto_out [issue_command\
                "changeto ${context}\nshow crypto files | inc CERT"]

   foreach elem [split $crypto_out \n] {

       if { [regexp {(\S+)\s+\d+\s+\S+\s+\S+\s+CERT} $elem - cert] } {
          set cert_out [issue_command\
        "changeto ${context}\nshow crypto certificate ${cert} | inc After"]

          if { [regexp {Not After :\s+(.* GMT)} $cert_out - cert_date] } {

              # check if certificate After date is less than today's date
              if { [clock scan $cert_date] < $current_date} {
                 puts "Context: $context - $cert is expired ($cert_date)"
```

```
                    # should send an alert!!

                } else {

                    puts "Context: $context - $cert is NOT expired ($cert_date)"

                    # get certificate After date minus 90 days
                    set cert_90_date [clock scan "-90 day" -base\
                                [clock scan $cert_date]]

                    # check if certificate After date expires within next 90 days
                    if { $cert_90_date < $current_date} {
                        puts "Content: $context - $cert will expire within 90 days\
                                    ($cert_date)"

                        # should send an alert!!
                    }
                }
            }
        }
    }
}
```

Sample Output:

```
./check_certificates
Context: Admin - testcer.pem is expired (May 24 23:59:59 2007 GMT)
Context: Paul - test-cert.pem is NOT expired (May 28 22:18:36 2008 GMT)
Content: Paul - test-cert.pem will NOT expire within 90 days (May 28 22:18:36
2008 GMT)
```

For More Information

Read more about the Cisco ACE Module, including case studies, white papers, data sheets, and end-of support and end-of-life notices. For Cisco data center solutions with Cisco Application Networking Services (ANS) and Cisco ACE, visit http://www.cisco.com/go/applicationservices or contact your local account representative. Subscribe to receive end-of-life and end-of-sale and support information by email.

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.

Printed in USA C07-689089-00 10/11