Oracle® Communications Subscriber-Aware Load Balancer

Essentials Guide

Release L-CX1.0.0

Formerly Net-Net Session-aware Load Balancer

October 2013

ORACLE

Contents

About This Guide

Overview

Version L-CX1.0.0 provides the initial release of the Oracle Communications Subscriber-aware Load Balancer Essentials Guide. This guide describes that release.

Audience

This guide is written for network administrators and architects, and provides information about the SLB configuration. Supporting, related material is available in the ACLI Configuration Guide, Release version 6.2.0. Please refer to that document as needed.

Supported Platforms

Release Version S-CX6.3.0 is supported on the Acme Packet 4500 and Acme Packet 3800 series platforms.

Related Documentation

The following table lists the members that comprise the documentation set for this release:

Document Name	Document Description
Acme Packet 4500 System Hardware Installation Guide	Contains information about the components and installation of the Acme Packet 4500 system.
Acme Packet 3800 Hardware Installation Guide	Contains information about the components and installation of the Acme Packet 3800 system.
Release Notes	Contains information about the current documentation set release, including new features and management changes.
ACLI Configuration Guide	Contains information about the administration and software configuration SBC.
ACLI Reference Guide	Contains explanations of how to use the ACLI, as an alphabetical listings and descriptions of all ACLI commands and configuration parameters.
Maintenance and Troubleshooting Guide	Contains information about Net-Net SBC logs, performance announcements, system management, inventory management, upgrades, working with configurations, and managing backups and archives.

Document Name	Document Description
MIB Reference Guide	Contains information about Management Information Base (MIBs), Enterprise MIBs, general trap information, including specific details about standard traps and enterprise traps, Simple Network Management Protocol (SNMP) GET query information (including standard and enterprise SNMP GET query names, object identifier names and numbers, and descriptions), examples of scalar and table objects.
Accounting Guide	Contains information about the SBC's accounting support, including details about RADIUS accounting.

Document Revision History

This section contains a revision history for this document.

Date	Revision Number	Description
March 31, 2011		Initial release

Session Load Balancer Configuration

As Acme Packet customers deploy larger and larger SIP access networks, scalability problems are presenting unique challenges, particularly from an operational standpoint. Deployments that scale beyond the number of users serviceable by one Acme Packet SBC – as well as deployments that use a geographically redundant Acme Packet SBC for catastrophic fail over purposes – encounter "edge reachability" problems. In general there are two coarse techniques that carriers use today to support endpoint populations that exceed one SBC's capacity: they will either use a DNS-based distribution mechanism (something that Acme Packet itself has advocated), or they will pre-provision endpoints to point to specific SBCs (manually load balancing them). Each of these solutions has its drawbacks. Acme Packet's customers – many of them familiar with load balancing equipment deployed to scale protocols such as HTTP or SMTP – have expressed interest in a device that will perform a load balancing function for their SIP endpoints.

Acme Packet developed the Session-aware Load Balancer (SLB) to address the need for scaling a network edge to millions of endpoints. Designed as a standalone system capable of supporting up to two million users (where a "user" is defined as a unique source IP address), and flexibly deployable into existing network topologies, the SLB aggregates signaling from large user populations to reduce the edge reachability problem by an order of magnitude.

Functional Overview

The Session Load Balancer (SLB) is a discrete network element that is the *first logical hop* for all *signaling traffic* from user agents entering a carrier network (and likewise, the *last logical hop* for all signaling traffic exiting the network and destined for user agents). The SLB, upon receipt of a SIP packet from an unknown source, uses a provisioned policy, running on a host processor software component, to select an appropriate next-hop Session Border Controller (SBC) for traffic originated by that source IP address, also referred to as an *endpoint*. Subsequent packets from that endpoint are forwarded to the same SBC without further host involvement. The first packet, the one used to make the route decision, and all subsequent packets sent through the SLB to the next-hop SBC are encapsulated within an IP-in-IP format as defined in RFC 2003, *IP Encapsulation within IP*.

SBCs that participate in the load balancing-enabled deployment are enhanced by several new capabilities. First, the SBC supports the RFC 2003 tunnel for both packet transmission and reception. Second, the SBC periodically transmits health and performance data to the SLB; such information is evaluated and entered into the SLB's route determination algorithm. Lastly, the SBC participates in an SLB-initiated *rebalance* operation, as described in "Rebalancing" on page 8. A group of SBCs, with the above-listed capabilities, that receive signaling traffic from the SLB is referred to as a *cluster*.

The IP-in-IP encapsulation technique provides SLB transparency to the terminating SBC. That is, when an SBC receives an encapsulated packet via the SLB, it can discard the outer encapsulation leaving behind an *identical* packet as transmitted

originally by the endpoint. Visibility into the actual packet transmitted by the endpoint is necessary to provide certain services in the SBC (for example, hosted NAT traversal, session-agent matching, and so on). A secondary goal achieved by using this encapsulation technique is that it provides a disassociation function between an SBC's connected network and its SIP reachability. That is, an SBC can be assigned any IP address it wants from a network topology standpoint, yet still process SIP packets as though it were logically situated elsewhere at Layer 5. In a larger sense, the *physicality* of the SBC is no longer important; like-configured, logically identical SBCs can be spread all over the globe.

Balancing and Rebalancing

The SLB performs two primary functions as the front-end to an SBC cluster: balancing traffic and rebalancing traffic. There are several key distinctions, which are described in the following two sections.

Balancing

Balancing is defined as the distribution of new endpoints (IP addresses) among the members of the SBC cluster. The SLB balances traffic based upon its configured policies (refer to "Load Balancer Policy Configuration" on page 16 for policy description and details), or, in the absence of configured policies, with a default round-robin procedure. Load balancer policies provide a flexible means of directing traffic to appropriate groups of SBCs. As initial packets arrive at the SLB from unknown (previously unseen) endpoints, they are passed to a resident software process that consults its policy engine to determine an appropriate destination (clustered SBC) for each endpoint. Regardless of the distribution algorithm, policy-based or round-robin, the SLB chooses an SBC from among all equally weighted candidates, giving preference to those with the lowest current *occupancy rate*, defined as the number of endpoints already present on that system relative to its maximum number of endpoints.

Even though each clustered SBC regularly reports CPU date to the SLB, the SBC's CPU utilization is not factored into the preference of one SBC over another. Rather, an SBC whose CPU utilization rate exceeds its load limit threshold (by default, 90%) is excluded from the list of candidates. For example, assuming that both SBCs are licensed for the same number of sessions, an SBC with a CPU load of 89% and a current occupancy of 10,000 endpoints will have equal footing with an SBC with a CPU load of 10% and a current occupancy of 10,000 endpoints. But an SBC with a CPU load of 90% and an occupancy of 0 endpoints will never receive new assignments from the SLB, until its CPU utilization rate falls below the 90% threshold.

Rebalancing

Rebalancing, as opposed to balancing, is taking some number of existing users from functioning SBCs and redistributing these existing users between current cluster members. Rebalancing occurs when a new SBC joins an existing cluster, or when invoked with the Acme Packet Command Line Interface (ACLI). When an SBC is removed from a cluster, whatever the reason, all of its endpoints are invalidated on the SLB and those users are essentially *balanced* when they revisit the SLB.)

A new SBC joins an existing cluster in response to a parameter contained in its start-up configuration that initiates the establishment of an IP-in-IP tunnel between the SBC and the SLB. During an initial handshake the SBC advertises which services it is configured to support. If there are existing SBCs offering this same service, and the SLB is configured to automatically rebalance, the SLB requests each of the SBCs

offering this service to divest themselves of a quantity of their subscribers. The SLB chooses the target quantity based upon the overall occupancy of that service relative to the SBC's contribution to that occupancy. It sends the requests using Cluster Control Protocol messaging to each SBC, instructing each SBC to *nominate* candidates for removal/redistribution. Each SBC in turn replies to the Cluster Control messages with a list of candidates; each Cluster Control reply will have approximately 80 candidates. The SLB removes existing forwarding rules associated with those users, and repeats this request/response process with each SBC until it has received the target quantity from each. When the affected users re-engage with the SLB (upon their next scheduled registration refresh, for example), the SLB lacks a forwarding rule that maps them to a specific SBC. Consequently, the message is passed up to the software processes running on the SLB's host, which choose a new destination for that user – presumably, the new SBC that has the most open capacity.

The SBC, when requested to nominate users for rebalancing, uses several criteria for choosing the most attractive candidates. As part of its standard session border control behavior the SBC *knows* the expiry times for all of the entries in its registration cache. Hence, the SBC can predict with a high degree of precision when any given user will be signaling back into the cluster. As the forwarding rules on the SLB are instantiated based upon the stimulus of a message from an endpoint, the SBC considers a user whose registration entry is most imminently due to expire a particularly attractive candidate. Note, however, that in many cases it is not prudent to nominate users whose registration cache entries are due to expire *immediately*, as this can cause a race condition between the removal message transmission from the SBC to the SLB and the refresh REGISTER message from the endpoint to the SLB. To avoid this potential dilemma, SBC are equipped with the ability to skip ahead to candidates whose expiry is not immediate.

Further, each SBC categorizes the endpoints stored in its cache based upon a priority value that is determined via the SLB's distribution policy (see "Distribution Policy Configuration" on page 18 for more details). It will choose endpoints from its lowest priority buckets first.

Lastly, the SBC does not rebalance a user that is actively in a phone conversation; it would be quite problematic to have in-dialog transactions for an active call arrive at a new SBC that did not handle the original call setup.

SLB Configuration

This section explains how to configure functionality specific to the SLB; it does not include configuration steps for elements that it shares in common with its corresponding SBCs (for example, *system-config*, *phy-interface*, *network-interface*, etc.) For information about general SBC configuration, refer to the Acme Packet documentation set as listed in *About This Guide*.

SLB configuration is quite simple; aside from basic network connectivity, the service interfaces, and the distribution policy, much of the configuration is otherwise learned dynamically learned from the SBCs that comprise the cluster.

Tunnel Configuration

The SLB sends and receives signaling messages to and from clustered SBCs through an IP-in-IP tunnel. The SLB requires one tunnel per interface per protocol. Currently SIP and H.248 are the only supported protocols. Consequently, running SIP and H.248 simultaneously on one SBC necessitates two parallel tunnels between the SLB and SBC, one tunnel reserved from SIP traffic, and the other for H.248 traffic.

Use the following procedure to perform required SLB-side *tunnel* configuration. Completion of tunnel configuration is accomplished on the clustered SBCs as described in "SBC Tunnel Configuration" on page 22.

1. From superuser mode, use the following ACLI command sequence to access *tunnel-config* configuration mode. While in this mode, you partially configure the *tunnel-config* configuration element.

```
alesmith# configure terminal
alesmith(configure)# system
alesmith(system)# network-interface
alesmith(network-interface)# tunnel-config
alesmith(tunnel-config)# ?

name                          tunnel name
local-address                 tunnel local IP address
port                          tunnel local and remote control ports
protocol                      tunnel control transport protocol
tls-profile                   tunnel control TLS profile
select                        select tunnel to edit
no                            delete tunnel
show                          show tunnel
done                          write tunnel information
exit                          return to previous menu

alesmith(tunnel-config)#
```

2. Ignore the **name** parameter.

   This setting will be provided by the SBC at the remote tunnel end.

3. Use the **local-address** parameter to specify the IP address at the SLB end of the tunnel.

   As the terminus for all tunnels from the clustered SBCs — and never the tunnel originator — only the local address is configured on the SLB.

```
alesmith(tunnel-config)# local-address 1.1.1.100
alesmith(tunnel-config)#
```

4. Use the **port** parameter to specify the port used to send and receive cluster control messages.

```
alesmith(tunnel-config)# port 4444
alesmith(tunnel-config)#
```

5. Use the **protocol** parameter to specify the transport protocol used in support of cluster control messages.

   Supported protocols are UDP (the default), TCP, or TLS.

   Note: Cluster control messages are exchanged quite frequently, and the overhead associated with encrypting and decrypting these messages is significant. Selection of TLS as the Transport Layer protocol degrades system performance.

```
alesmith(tunnel-config)# protocol UDP
alesmith(tunnel-config)#
```

6. If TLS is the selected transport protocol, use the **tls-profile** parameter to select the existing TLS profile that identifies the cryptographic resources used to secure the TLS connection.

```
alesmith(tunnel-config)# tls-profile TLS-LB
alesmith(tunnel-config)#
```

7. Use **done** and **exit** to complete configuration of this *tunnel-config* configuration element.
8. Repeat steps 1 through 7 to configure additional *tunnel-config* configuration elements.

Cluster Configuration

The *cluster-config* configuration element manages basic SLB interaction with clustered SBCs — it contains a set of global parameters that define the management of the RFC 2003 IP-in-IP tunnels that connect the SLB to clustered SBCs, and the details of rebalance operations. In addition, *cluster-config* provides for the creation of a list of *service interfaces* (signaling addresses) that are advertised to endpoints comprising the user access population.

Use the following procedure to perform required *cluster-config* configuration.

1. From superuser mode, use the following ACLI command sequence to access *cluster-config* configuration mode. While in this mode, you configure the *cluster-config* configuration element.

```
alesmith# configure terminal
alesmith(configure)# session-router
alesmith(session-router)# cluster-config
alesmith(cluster-config)# ?

state                          cluster control state
log-level                      configure log level
auto-rebalance                 Auto-rebalance cluster on new SD
                               availability
source-rebalance-threshold     Percentage of advertised registration
                               capacity
dest-rebalance-threshold       Percentage of advertised registration
                               capacity
dest-rebalance-max             Percentage of advertised registration
                               capacity
tunnel-check-interval          How often an SD's tunnels are checked
tunnel-fail-interval           Time for which no messages have been
                               received
rebalance-request-delay        Delay between subsequent rebalance
                               requests
session-multiplier             ratio of users (endpoints to sessions)
atom-limit-divisor             ratio of atoms (e.g. contacts to
                               endpoints)
rebalance-skip-ahead           Skip Endpoints refreshing sooner than
rebalance-max-refresh          Skip Endpoints refreshing later than
ignore-tgt-svcs-on-rebalance   When selecting source SDs during
                               rebalancing
rebalance-del-app-entries      Delete Application Endpoint Data
inactive-sd-limit              Duration no SD control messages received
                               (seconds)
red-port                       redundant mgcp sync port
red-max-trans                  max redundant transactions to keep
red-sync-start-time            redundant sync start timeout
red-sync-comp-time             redundant sync complete timeout
```

```
service-ports                 configure service ports
select                        select cluster config
no                            delete cluster config
show                          show cluster config
done                          save cluster config information
exit                          return to previous menu

alesmith(cluster-config)#
```

2. Use the **state** parameter to enable or disable the SLB software.

   The default setting, *enabled*, enables SLB functionality; *disabled* renders the SLB inoperable.

   ```
   alesmith(cluster-config)# state enabled
   alesmith(cluster-config)#
   ```

3. Use the **log-level** parameter to specify the contents of the SLB log.

   Log messages are listed below in descending order of severity.

emergency	— the most severe
critical	
major (error)	
minor (error)	
warning	
notice	
info	— (default) the least severe
trace	— (test/debug, not used in production environments)
debug	— (test/debug, not used in production environments)
detail	— (test/debug, not used in production environments)

   In the absence of an explicitly configured value, **log-level** defaults to *critical*, meaning that log messages with a severity of *critical* or greater (*emergency*) are written to the SLB log.

   ```
   alesmith(cluster-config)# log-level critical
   alesmith(cluster-config)#
   ```

4. Use the **auto-rebalance** parameter to specify SLB behavior when a new SBC joins an existing cluster.

   With this parameter *enabled*, the default setting, the SLB redistributes endpoints among cluster members when a new member joins the cluster. Refer to"Rebalancing" on page 8 for operational details.

   With this parameter *disabled*, the alternate setting, pre-existing SBCs retain their endpoint populations, and the SLB directs all new endpoints to the newly active SBC until that SBC reaches maximum occupancy.

   ```
   alesmith(cluster-config)# auto-rebalance enabled
   alesmith(cluster-config)#
   ```

5. If **auto-rebalance** is set to *enabled*, use the **source-rebalance-threshold** and **dest-rebalance-threshold** parameters to specify threshold settings that identify existing cluster SBCs as either endpoint sources or endpoint destinations during the rebalance operation. Use the **dest-rebalance-max** parameter to specify the occupancy rate for the new cluster member. Refer to "Balancing" on page 8 for details on occupancy and its calculation.

   If **auto-rebalance** is set to *disabled*, these three parameters can be ignored.

   Parameter values are numeric percentages within the range 0 through 100.

source-rebalance-threshold specifies the minimum occupancy rate that identifies a clustered SBC as a source of endpoints during a rebalance operation. For example, using the default value of *50* (percent), any clustered SBC with an occupancy rate of 50% or more sheds endpoints during a rebalance. The SLB assigns these endpoints to the new cluster member.

dest-rebalance-threshold specifies the maximum occupancy rate that identifies a clustered SBC as a destination of endpoints during a rebalance operation. Note that using the default setting of *0* (percent), ensures that no pre-existing SBC gains endpoints during a rebalance.

dest-rebalance-max specifies the maximum occupancy rate that the SLB transfers to the new cluster member during a rebalance operation. The default setting is *80* (percent). Should this threshold value be attained, the SLB distributes remaining endpoints to those SBCs identified as endpoint destinations by their **dest-rebalance-threshold** settings.

```
alesmith(cluster-config)# source-rebalance-threshold 50
alesmith(cluster-config)# dest-rebalance-threshold 40
alesmith(cluster-config)# dest-rebalance-max 75
```

6. If **auto-rebalance** is set to *enabled*, you can optionally use four additional parameters to fine-tune rebalance operational details.

   If **auto-rebalance** is set to *disabled*, these four parameters can be ignored.

   rebalance-request-delay specifies the interval (in milliseconds) between endpoint request messages sent from the SLB to a clustered SBC. As explained in "Rebalancing" on page 8 these messages request a list of endpoints that will be redistributed from the SBC to a new cluster member. By default, this parameter is set to 500 milliseconds.

   rebalance-skip-ahead restricts the target set of SBC endpoints registration eligible for rebalancing to those whose re-registration is not imminent — that is, the registration is not scheduled within the number of milliseconds specified by the parameter setting. Setting this parameter to a non-zero value mitigates against the possibility of a race condition precipitated by a simultaneous endpoint removal generated by the SBC and the arrival of endpoint signalling on an SLB service port. The default setting (*0* milliseconds) effectively makes the entire SBC endpoint set eligible for rebalancing.

   rebalance-max-refresh restricts the target set of SBC endpoints eligible for rebalancing to those whose re-registration is far in the future — that is, the registration is not scheduled until after the number of milliseconds specified by the parameter setting. The default setting (*0* milliseconds) effectively makes the entire SBC endpoint set eligible for rebalancing.

   rebalance-del-app-entries specifies when cached SIP entries for rebalanced endpoints are removed from the clustered SBC. The default setting (*disabled*) specifies that cached entries are retained after a rebalance operation, and subsequently removed from the cache by standard time-out procedures. When set to *enabled*, this parameter specifies that the SBC removes cached registration entries at the completion of the rebalance operation.

   ```
   alesmith(cluster-config)# rebalance-request-delay 750
   alesmith(cluster-config)# rebalance-skip-ahead 100
   alesmith(cluster-config)# rebalance-max-refresh 1000
   alesmith(cluster-config)# rebalance-del-app-entries enabled
   ```

7. Three parameters, **tunnel-fail-interval**, **tunnel-check-interval,** and **inactive-sd-limit** maintain and monitor the IP-in-IP tunnels established between the SLB and clustered SBCs.

   tunnel-fail-interval specifies the interval (in milliseconds) between periodic *keepalive* messages sent from a clustered SBC to the SLB. If the SLB fails to receive a *keepalive* message within the specified period, it flags the tunnel as *dead*. By default, this parameter is set to *10000* milliseconds.

   tunnel-check-interval specifies the interval (in milliseconds) between SLB tunnel audits. During a tunnel audit, the SLB checks the status of each tunnel and removes all tunnels flagged as *dead*. By default, this parameter is set to *15000* milliseconds.

   If you change default settings for either parameter, ensure that the setting for **tunnel-check-interval** is greater than the **tunnel-fail-interval** setting.

   inactive-sd-limit specifies the maximum silent interval (in seconds) before the SLB flags the SBC as *dead*, and removes it from the cluster. By default, this parameter is set to *1800* seconds (30 minutes). supported values are integers within the range 0 through 31556926 (365 days).

   ```
   alesmith(cluster-config)# tunnel-fail-interval 10000
   alesmith(cluster-config)# tunnel-check-interval 15000
   alesmith(cluster-config)# inactive-sd-limit 900
   ```

8. Use the **session-multiplier** and **atom-limit-divisor** parameters to specify a numeric factors used in *occupancy* and *occupancy rate* calculations.

   session-multiplier provides a factor that when multiplied by an SBC's licensed session limit, determines the maximum number of endpoints that the SBC can support (that is, its maximum *occupancy*).

   The default setting is *10*; valid settings include any integer values within the range 1 through 100.

   Using the default setting, an SBC licensed for 32,000 concurrent sessions has a maximum theoretical occupancy of 320,000 endpoints.

   atom-limit-divisor provides another factor that can be used in *occupancy* and *occupancy rate* calculations. By default, occupancy calculations are based on endpoints (IP addresses), and do not take into account the fact that the same IP address can represent multiple users.

   The default setting is *1*, which assumes a conservative 1-to-1 correlation between endpoints and users; valid settings include any integer values within the range 1 through 1000.

   Note: Once an SBC has reached its maximum number of endpoints, the SLB removes it from the load balancing algorithm. These parameter settings should be changed only after careful examination of network conditions and behavior.

   ```
   alesmith(cluster-config)# session-multiplier 10
   alesmith(cluster-config)# atom-limit-divisor 1
   ```

9. The **ignore-tgt-svc-on-rebalance** command is not currently supported.
10. Retain default settings for the **red-port**, **red-max-trans**, **red-sync-start-time**, and **red-sync-comp-time** parameters.
11. Use **done** and **exit** to complete cluster configuration.

Service Ports Configuration

A service port is an access-side network interface monitored by the SLB for incoming signaling from the user population. For virtually all network topologies, multiple service ports are expected on a typical SLB configuration. A *service-port* is a multiple instance configuration element; for each service port advertised to the access network(s), at least *one service-port* configuration element must be configured.

Use the following procedure to perform required *service-ports* configuration.

1. From superuser mode, use the following ACLI command sequence to access *service-port* configuration mode. While in this mode, you configure one or more *service-port* configuration elements.

```
alesmith# configure terminal
alesmith(configure)# session-router
alesmith(session-router)# cluster-config
alesmith(cluster-config)# service-ports
alesmith(service-port)# ?

address                       IP address
port                          port (default: 5060)
protocol                      transport protocol
network-interface             network interface for service port
select                        select cluster config
no                            delete cluster config
show                          show cluster config
done                          save cluster config information
exit                          return to previous menu

alesmith(service-port)#
```

2. Use the required **address** parameter to specify the IP address of this service port.

```
alesmith(service-port)# address 10.0.0.1
alesmith(service-port)#
```

3. Use the **port** parameter to specify the port monitored by the SLB for incoming signaling messages.

   In the absence of an explicitly configured port, the SLB provides a default value of 5060 (the registered SIP port).

   Allowable values are integers within the range 0 through 65535.

```
alesmith(service-port)# port 5060
alesmith(service-port)#
```

4. Use the **protocol** parameter to choose the transport protocol.

   Supported settings are *UDP* (the default) and TCP.

```
alesmith(service-port)# protocol udp
alesmith(service-port)#
```

5. Use the required **network-interface** parameter to identify the SLB network interface that supports this service port.

```
alesmith(service-port)# network-interface M00:0
alesmith(service-port)#
```

6. Use **done** and **exit** to complete configuration of this *service-port* configuration element.

7. Repeat steps 1 through 6 to configure additional *service-port* configuration elements.

Load Balancer Policy Configuration

The *lbp-config* configuration element manages the SLB endpoint table. It also creates and manages a list of *service interfaces* (signaling addresses) that are advertised to endpoints comprising the user access population.

Use the following procedure to perform required *lbp-config* configuration.

1. From superuser mode, use the following ACLI command sequence to access *lbp-config* configuration mode. While in this mode, you configure the *lbp-config* configuration element.

```
alesmith# configure terminal
alesmith(configure)# session-router
alesmith(session-router)# lbp-config
alesmith(lbp-config)#?

state                                lbp state
log-level                            configure log level
red-port                             lbp redundant sync port: 0 to
                                     disable and 2000 to enable
red-max-trans                        maximum redundancy transactions to
                                     keep on active
red-sync-start-time                  timeout for transitioning from
                                     standby to active
red-sync-comp-time                   sync request timeout after initial
                                     sync completion
untrusted-grace-period               Untrusted grace period
max-untrusted-percentage             Maximum untrusted endpoints
                                     percentage
max-untrusted-upper-threshold        Maximum untrusted endpoints upper
                                     threshold
max-untrusted-lower-threshold        Maximum untrusted endpoints upper
                                     threshold
endpoint-capacity-upper-threshold    Endpoint capacity upper threshold
endpoint-capacity-lower-threshold    Endpoint capacity lower threshold
rebalance-request-delay              Delay between subsequent rebalance
options                              optional features/parameters
service-ports                        configure service ports
select                               select cluster config
no                                   delete cluster config
show                                 show cluster config
done                                 save cluster config information
exit                                 return to previous menu

alesmith(lbp-config)#
```

2. Use the **state** parameter to enable or disable the LBP software.

   The default setting, *enabled*, enables SLB functionality; *disabled* renders the SLB inoperable.

```
alesmith(lbp-config)# state enable
alesmith(lbp-config)#
```

3. Use the **log-level** parameter to specify the contents of the LBP log.

   Log messages are listed below in descending order of severity.

   emergency — the most severe
   critical
   major (error)
   minor (error)
   warning

notice
info — (default) the least severe
trace — (test/debug, not used in production environments)
debug — (test/debug, not used in production environments)
detail — (test/debug, not used in production environments)

In the absence of an explicitly configured value, **log-level** defaults to *critical*, meaning that log messages with a severity of *critical* or greater (*emergency*) are written to the LBP log.

```
alesmith(lbp-config)# log-level critical
alesmith(lbp-config)#
```

4. Use the **untrusted-grace-period**, **max-untrusted-percentage**, **max-untrusted-upper-threshold**, and **max-untrusted-lower-threshold** parameters to implement percentage-based management and monitoring of untrusted endpoints in the SLB endpoint database. Management and monitoring of untrusted endpoints is instrumental in detecting and responding to Denial-of-Service (DOS) attacks aimed at the SLB.

   untrusted-grace-period specifies the maximum time, in seconds, that a forwarding rule is retained by the SLB before it is confirmed with a promotion message from the SBC which received the untrusted endpoint. Refer to "Balancing" on page 8 for message details.

   In the absence of an explicitly assigned value, the SLB provides a default setting of *30* (seconds).

   If this time period elapses without a promotion message arriving to confirm this user, the SLB deletes the entry.

   Setting this parameter to 0 allows untrusted/unconfirmed entries to exist indefinitely without aging out.

   max-untrusted-percentage specifies the percentage of the overall endpoint population that is reserved for untrusted users.

   The default setting is *20* (percent); supported values are integers within the range 1 through 100.

   This percentage is applied to the overall remaining occupancy of the SLB after trusted (confirmed) users are accounted for. For example, when empty, the SLB holds two million forwarding rules; assuming the default setting, at most 400,000 rules are reserved for untrusted rules. By the time one million users have been promoted, 20% of the remaining space means that up to 200,000 entries can be used for untrusted users.

   max-untrusted-upper-threshold specifies a threshold level at which the SLB (1) raises an alarm, and (2) issues an SNMP trap reporting an excessive number of untrusted endpoints within the entire endpoint population.

   This parameter, which has a default setting of *80* (percent), is calculated as a percent of **max-untrusted-percentage**. For example, assuming default settings for both parameters, the SLB raises an alarm and issues an SNMP trap when the percentage of untrusted endpoints attains 16%.

   max-untrusted-lower-threshold specifies a threshold level at which the SLB (1) clears the existing untrusted endpoint alarm, and (2) issues an SNMP trap reporting alarm clearance.

   This parameter, which has a default setting of *70* (percent), is calculated as a percent of **max-untrusted-percentage**. For example, assuming default settings for both parameters, the SLB clears an alarm and issues an SNMP trap when the percentage of untrusted endpoints falls to 14%.

```
alesmith(lbp-config)# untrusted-grace-period 30
alesmith(lbp-config)# max-untrusted-percentage 20
alesmith(lbp-config)# max-untrusted-upper-threshold 80
alesmith(lbp-config)# max-untrusted-lower-threshold 70
alesmith(lbp-config)#
```

5. Use the **endpoint-capacity-upper-threshold** and **endpoint-capacity-lower-threshold** parameters to implement license-based management and monitoring of the SLB endpoint counts.

   endpoint-capacity-upper-threshold specifies a threshold level at which the SLB (1) raises an alarm, and (2) issues an SNMP trap reporting an excessive number of active endpoints.

   This parameter, which has a default setting of *80* (percent), is calculated as a percentage of the endpoints allowed by the installed SLB license.

   endpoint-capacity-lower-threshold specifies a threshold level at which the SLB (1) clears the existing endpoint alarm, and (2) issues an SNMP trap reporting alarm clearance.

   This parameter, which has a default setting of *70* (percent), is calculated as a percentage of the endpoints allowed by the installed SLB license.

```
alesmith(lbp-config)# endpoint-capacity-upper-threshold 80
alesmith(lbp-config)# endpoint-capacity-lower-threshold 70
alesmith(lbp-config)#
```

6. Use **done** and **exit** to complete configuration of this *load-balancer-policy* configuration element.

Distribution Policy Configuration

Distributing endpoints equitably among the cluster members is the primary function of the SLB. The *lp-config* configuration element allows you to control the method of the SLB's distribution based on matching criteria. Using inbound packet matching criteria, you can control the assignment of users to SBCs. Matching is done by data available up to and including the transport layer of the packet: source IP address and port, destination IP address and port, and transport protocol. The IP addresses and ports may or may not include bit masks as well.

Conceptually, the load balancer policy table, with sample data, looks akin to the following.

Source IP/Mask	Source Port/ Mask	Destination IP/Mask	Destination Port/Mask	Transport Protocol Require-ments (list)	Realm Identi-fiers (list)
192.168.7.22/32	0/0	10.0.0.1/32	5060/16		West
192.168.1.0/24	0/0	10.0.0.1/32	5060/16	UDP, TCP	North, South, West
192.168.0.0/16	0/0	10.0.0.1/32	5060/16	UDP, TCP	East, West
0.0.0.0/0	0/0	0.0.0.0/0	0/0		

Policies are matched using a longest prefix match algorithm; the most specific policy is selected when comparing policies to received packets. One and only one policy is chosen per packet; if the next hops in that route are all unavailable, the next best route is not consulted (instead, the *default policy* may be consulted – see below). This is different than the local-policy behavior on the SBC.

Within each policy you may configure multiple next hops, where each next hop is a named group of SBCs. In the sample policy table, this is indicated in the second policy with a source IP range of 192.168.1.0/24. The realm identifier list for this policy indicates North, South, West. Each of these realm identifiers represents a collection of zero or more SBCs, in SBC parlance these are roughly analogous to *session-agent groups*. Each of these realm identifiers is also assigned a *priority* (a value between 1 and 31, with 1 representing the highest priority) in the configuration, and the SLB sorts the possible destinations with the highest priority first. Upon receipt of a packet matching a policy with multiple configured realm identifiers, the SLB gives preference to SBCs from the realm identifier with the highest priority. Should no SBCs be available in that priority level (due to saturation, unavailability, and so on.) the SLB moves on to investigate the next priority level, and so on. Should no SBCs become available after traversing the entire list of all SBCs within each priority level, the SBC either drops the packet or attempt to use the *default policy*.

The bottom row of the sample table shows this implicit, last resort default policy. When enabled, the SLB reverts to the default policy when all of the potential next hop realms referenced in the endpoint's distribution rule are unavailable. In that event, the default policy attempts to locate a clustered SBC that advertises support for the service-interface that the packet arrived on. The realm is not considered when matching to the default policy. If such an SBC is found, the SLB forwards the packet to that SBC; if such an SBC is not found, the SLB drops the packet.

It is not necessary to configure the default policy — it is simply intended as a catchall policy, and may be used when all that is required is a simple round-robin balancing scheme based on simple metrics (for example, CPU utilization and number of registrations currently hosted by an SBC). If no policies are configured on the SLB, the default policy is used. The default realm is implied in the above table as "*" and is enabled by default for policy records.

Use the following procedure to perform required *lb-config* configuration.

1. From superuser mode, use the following ACLI command sequence to access *lb-config* configuration mode. While in this mode, you configure the distribution rules used to implement policy-based load balancing on the SLB.

```
alesmith# configure terminal
alesmith(configure)# session-router
alesmith(session-router)# lb-config
alesmith(lb-config)#?

state                              lb policy state
default-realm                      use default realm
description                        load balancer policy description
protocols                          list of protocols
lb-realms                          list of realms
                                        name
                                        priority
source-addr                        source ip address
destination-addr                   destination ip address
select                             select lb policy
```

```
no                                  delete lb policy
show                                show lb policy
done                                save lb policy information
exit                                return to previous menu

alesmith(lb-config)#
```

2. Use the **state** parameter to enable or disable this distribution rule.

   The default setting, *enabled*, enables the distribution rule; *disabled* disables the rule.

   ```
   alesmith(lb-config)# state enabled
   alesmith(lb-config)#
   ```

3. Use the **default-realm** parameter to enable or disable the default distribution policy.

   The default setting, *enabled*, enables the default policy; *disabled* disables the policy.

   With **default-realm** enabled, the SLB provides a best-effort delivery model if the next-hop realms listed in this distribution rule are unavailable. With **default-realm** disabled, the orphaned packet is dropped.

   ```
   alesmith(lb-config)# default-realm enabled
   alesmith(lb-config)#
   ```

4. Optionally use the **description** parameter to provide a description of this distribution rule.

   ```
   alesmith(lb-config)# description “Local traffic to Los Angeles site”
   alesmith(lb-config)#
   ```

5. Use the **protocols** parameter to construct a list of protocols that must be supported by this distribution rule.

   ```
   alesmith(lb-config)# protocols “udp tcp”
   alesmith(lb-config)#
   ```

6. Use either the **source-addr** parameter or the **destination-address** parameter to specify matching criteria for this distribution rule.

   Use the **source-addr** parameter to specify source-address-based matching criteria.

   Packets whose source IP addresses match the criteria specified by this parameter are subject to this distribution rule.

   ```
   alesmith(lb-config)# source-addr 10.0.0.1
   alesmith(lb-config)#
   ```

   matches any port on the specified IP source address

   ```
   alesmith(lb-config)# source-addr 10.0.0.1:5060
   alesmith(lb-config)#
   ```

   matches the specified IP source address:port pair

   ```
   alesmith(lb-config)# source-addr 10.0.0.1/24
   alesmith(lb-config)#
   ```

   matches any IP source address, any port on the 10.0.0.x subnet

   ```
   alesmith(lb-config)# source-addr 10.0.0.240/28:5060
   alesmith(lb-config)#
   ```

   matches IP source addresses 10.0.0.240:5060 through 10.0.0.255:5060

Use the **destination-addr** parameter to specify destination-address-based matching criteria.

Packets whose destination IP addresses match the criteria specified by this parameter are subject to this distribution rule.

```
alesmith(lb-config)# destination-addr 10.0.0.1
alesmith(lb-config)#
```

matches any port on the specified IP destination address

```
alesmith(lb-config)# destination-addr 10.0.0.1:5060
alesmith(lb-config)#
```

matches the specified IP destination address:port pair

```
alesmith(lb-config)# destination-addr 10.0.0.1/24
alesmith(lb-config)#
```

matches any IP destination address, any port on the 10.0.0.x subnet

```
alesmith(lb-config)# destination-addr 10.0.0.240/28:5060
alesmith(lb-config)#
```

matches destination IP addresses 10.0.0.240:5060 through 10.0.0.255:5060

7. Use the **lb-realms** parameter to access *lb-realm* configuration mode.

   While in *lb-realm* configuration mode you identify one or more SBCs eligible to receive traffic that matches this distribution rule.

   ```
   alesmith(lb-config)# lb-realms
   alesmith(lb-realm)#
   ```

8. Use the **name** parameter to identify the realm.

   As previously discussed, the *name* field is roughly analogous to an SBC session-agent group. SBCs configured to communicate within a cluster hosted by an SLB advertise offered services to the SLB. These services (for example, SIP support or H.248 support) exist in realms, whose names are sent to the SLB as part of the SBC advertisement. The SLB, upon receipt of these advertisements, joins each SBC into one or more *realm identifier groups* based upon the realm name(s) the SBC has offered up. The **name** command of the *lb-realm* configuration element matches this distribution rule to a supporting SBC that has offered that realm name for cluster membership.

   ```
   alesmith(lb-realm)# name LosAngeles
   alesmith(lb-realm)#
   ```

9. Use the **priority** parameter to specify the realm priority.

   Priority is expressed as an integer value within the range 0 to 31 — the lower the integer, the greater the priority.

   The default value, 0, specifies use of the default routing policy, and should not be used when policy-based distribution is enabled.

   Priority values are considered when multiple SBCs offer the same service to matched packets.

   ```
   alesmith(lb-realm)# priority 1
   alesmith(lb-realm)#
   ```

10. Use **done** and **exit** to complete configuration of this *lb-realm* configuration element.

11. To specify other eligible SBCs, repeat steps 7 through 10. For example.

```
alesmith(lb-config)# lb-realms
alesmith(lb-realm)# name LasVegas
alesmith(lb-realm)# priority 5
alesmith(lb-realm)# done
alesmith(lb-realm)# exit
alesmith(lb-realm)# verify-config
```

12. Use **done** and **exit** to complete configuration of this distribution rule.
13. To specify additional distribution rules, repeat steps 1 through 12 as often as necessary.

SBC Configuration

This section describes the configuration necessary to allow an SBC to join a cluster. Configuration is simplified to allow for an easy and seamless migration from a deployed standalone SBC to a deployed clustered SBC. There are only two places where new configuration is required: in the *network-interface* configuration element, where tunnel information is defined; and in the signaling application's interface, either the *sip-interface* configuration element for clustering the SIP protocol or the *h248-mgc-config* configuration element for clustering the H.248 protocol.

SBC Tunnel Configuration

Configuring the properties of the IP-in-IP tunnel on the SBC is a matter of configuring the local address and specifying transport layer and application layer protocol support.

The following example uses a tunnel named *sipSignaling*, which was initially and partially configured on the SLB. Note in the following configuration that the **remote-address** parameter is not specified — that value having been previously set with the **local-address** parameter on the SLB. The complementary configuration performed on the SBC enables tunnel establishment between the SBC and the SLB.

1. From superuser mode, use the following ACLI command sequence to access *tunnel-config* configuration mode. While in this mode, you perform required SBC tunnel configuration.

```
westy# configure terminal
westy(configure)# system
westy(system)# network-interface
westy(network-interface)# tunnel-config
westy(tunnel-config)# ?

name            tunnel name
local-address   tunnel local IP address
remote-address  tunnel remote IP address
application     application protocol for this tunnel
port            tunnel local & remote control ports
protocol        tunnel control transport protocol
tls-profile     tunnel control TLS profile
select          select tunnel to edit
no              delete tunnel
show            show tunnel
done            write tunnel information
exit            return to previous menu

westy(tunnel-config)#
```

2. Use the **name** command to provide a unique identifier for this tunnel instance.

```
westy(tunnel-config)# name sipSignaling
westy(tunnel-config)#
```

3. Use the **local-address** parameter to specify the IP address at the SBC end of the tunnel.

```
westy(tunnel-config)# local-address 1.1.1.100
westy(tunnel-config)#
```

4. Use the **port** parameter to specify the port used to send and receive cluster control messages.

```
westy(tunnel-config)# port 4444
westy(tunnel-config)#
```

5. Use the **application** parameter to specify the application protocol supported by this tunnel.

   Supported application protocols are SIP or H.248.

```
westy(tunnel-config)# application SIP
westy(tunnel-config)#
```

6. Use the **protocol** parameter to specify the transport protocol used in support of cluster control messages.

   Supported transport protocols are UDP (the default), TCP, or TLS.

   Note: Cluster control messages are exchanged quite frequently, and the overhead associated with encrypting and decrypting these messages is significant. Selection of TLS as the Transport Layer protocol degrades system performance.

```
westy(tunnel-config)# protocol UDP
westy(tunnel-config)#
```

7. If TLS is the selected transport protocol, use the **tls-profile** parameter to select the existing TLS profile that identifies the cryptographic resources used to secure the TLS connection.

   TLS usage is not recommended!

```
westy(tunnel-config)# tls-profile TLS-ClusterMember
westy(tunnel-config)#
```

8. Use **done** and **exit** to complete configuration of this *tunnel-config* configuration element.
9. Repeat steps 1 through 8 to complete tunnel configuration on other SIP interfaces as required.

SIP Configuration

In a traditional SBC configuration the IP address assigned to a *sip-port* configuration element is contained within the address space defined by the network interface netmask. This is not be the case for clustered SBCs. Rather, the IP address assigned to the sip-port is identical to the address of an SLB service-port advertised on the access network. The process of encapsulating the packets between the SLB and SBC masks the fact that the IP address the SBC expects to receive IP packets on is different than the Layer 5 address the SBC expects the SIP address on.

Consistency of realm identification is vital to successful and predictable policy-based load balancing. Take particular care to ensure that the **realm-id** of the *sip-interface* configuration element mirrors the **lb-realm** assignments made while configuring distribution rules. See “Distribution Policy Configuration” on page 18.

In the following configuration example, the **realm-id** is *LosAngeles*. This SBC, when booted, will detect that it is a member of an SLB cluster and register the service port 10.0.0.1:5060/UDP as the realm *LosAngeles* with the SLB. The SLB will automatically create the SBC group *LosAngeles* (if it doesn't exist) or join the SBC to the group *LosAngeles* (if it is not the first to advertise *LosAngeles*). Policy statements that direct packets to *LosAngeles* now consider this SBC as a potential destination, assuming the address:port/protocol also are consistent with the policy's matching criteria.

This technique allows you to configure the same IP:port/protocol on multiple SBCs, with different realm-id labels, to indicate priority of one SBC or group of SBCs over another. As an example, consider several SBCs geographically situated together with the label *LosAngeles*, and several other SBCs geographically situated elsewhere with the label *NewYork*, all with the identical SIP interface and SIP port configuration. A policy can be easily defined to give preference to a source subnet of users in California to the *LosAngeles* member SBCs, with *NewYork* as a second priority. This provides flexibility in network design without undue burden in the configuration: SBCs' tagged with the same realm name are joined in dynamically created SBC groups by the SLB, with no explicit configuration required on the SLB whatsoever.

1. From superuser mode, use the following ACLI command sequence to access *sip-interface* configuration mode. While in this mode, you verify the **realm-id** and assign the newly created IP-in-IP tunnel to a SIP interface.

```
westy# configure terminal
westy(configure)# session-router
westy(session-router)# sip-interface
westy(sip-interface)# select
<realm-id>: LosAngeles
1: LosAngeles  172.192.1.15:5060

selection: 1
westy(sip-interface)# show
sip-interface
        state                   enabled
        realm-id                LosAngeles
        ...
        ...
        ...
westy(sip-interface)#
```

2. Use the **tunnel-name** parameter to assign the IP-in-IP tunnel to the current SIP interface.

```
westy(sip-interface)# tunnel-name sipSignaling
westy(sip-interface)# ?
```

3. Use the **sip-port** command to move to *sip-port* configuration mode.

```
westy(sip-interface)# sip-port
westy(sip-port)# ?

address            IP Address
port               port (default: 5060)
transport-protocol transport protocol
tls-profile        the profile name
ims-aka-profile    ims-aka profile name
select             select a sip port to edit
no                 delete a selected sip port
```

```
show                show sip port information
done                write sip port information
exit                return to previous menu

westy(sip-port)#
```

4. Use the **address**, **port**, and **transport-protocol** parameters to mirror the address of an existing SLB service port.

```
westy(sip-port)# address 10.0.0.1
westy(sip-port)# port 5060
westy(sip-port)# transport-protocol udp
westy(sip-port)#
```

5. Use **done**, **exit**, and **verify-config** to complete configuration of this *sip-port* configuration element.
6. Repeat steps 1 through 5 as necessary to verify **realm-id**s, assign IP-in-IP tunnels, and create mirrored service ports on additional SIP interfaces.

H.248 Configuration

As with SIP, the H.248 application configuration is straightforward. For a given H.248 virtual media gateway controller configuration object (*h248-mgc-config*), verify realm consistency and addressing information, and assign an IP-in-IP tunnel to the H.248 interface.

1. From superuser mode, use the following command ACLI sequence to access *h248-mgc-config* configuration mode. While in this mode, you verify the **realm-id**, ensure that the **ip-address**, **port**, and **transport-method** parameters settings mirror those of the associated SLB service port, and assign the newly created IP-in-IP tunnel to the H.248 interface.

```
westy# configure terminal
westy(configure)# session-router
westy(session-router)# h248-config
westy(h248-config)# h248-mgc-config
westy(h248-mgc-config)# select
name: slbService
1: slbService

selection: 1
westy(h248-mgc-config)# show
h248-mgc-config
        name                            slbService
        state                           enabled
        realm-id                        NewYork
        ip-address                      10.0.0.25
        port                            2944
        transport-method                udp
        ...
        ...
        ...
westy(h248-mgc-config)#
```

2. Verify the consistency of the **realm-id** parameter value.
3. Verify that **ip-address**, **port**, and t**ransport-method** parameter values mirror an SLB service port.

4. Use the **options** parameter to assign an IP-in-IP tunnel to the current H.248 interface.

```
westy(h248-mgc-config)# options +tunnel=h248Signaling
westy(h248-mgc-config)# ?
```

5. Use **done**, **exit**, and **verify-config** to complete configuration of this *h248-mgc-config* configuration element.
6. Repeat steps 1 through 5 as necessary to verify **realm-id**s, assign IP-in-IP tunnels, and create mirrored service ports on additional H.248 interfaces.

Forced Rebalance

The **notify ccd rebalance** ACLI command initiates an immediate forced rebalance operation. A forced rebalance operation is identical to the one described in "Rebalancing" on page 8.

```
westy# notify ccd rebalance
westy#
```

Online/Offline Configuration

The **set-system-state** ACLI command provides the ability to temporarily place a clustered SBC in the *offline* state. The offline setting puts the SBC into a state where it is powered on and available only for administrative purposes.

The transition to the offline state is graceful in that existing calls are not affected by the state transition. The SBC informs the SLB of the impending status change via a Cluster Control Protocol message. Upon receiving such a message, the SLB ceases to forward new endpoints to the SBC, and places the SBC in the *Shutdown* state. The SBC, for its part, enters a state that results in the rejection of any incoming out-of-dialog SIP requests. Eventually all calls compete, registrations expire and are removed by the SLB, and returning endpoints are allotted to active SBCs.

Use the **set-system-state offline** ACLI command to place an SBC in the offline state.

```
westy# set-system-state offline
Are you sure you want to bring the system offline? [y/n]?: y
Setting system state to going-offline, process will complete when all
current calls have completed
westy#
```

Note: An SBC in the offline state plays no role in a balance or rebalance operation.

In a similar fashion use the **set-system-state online** ACLI command to place an SBC in the online state.

```
westy# set-system-state online
Are you sure you want to bring the system online? [y/n]?: y
Setting system state to online
westy#
```

SLB Statistical Output

The SLB provides the operator with a full set of statistical data for troubleshooting and diagnostic purposes. This section describes current statistical outputs and defines displayed values. It is important to become familiar with the data and the collection process when opening trouble tickets with the Acme Packet Technical Assistance Center, as they will rely upon this information to assist you in diagnosing hardware, software, and/or network issues.

The **show balancer** command is the root of all statistical data pertinent to SLB operation. Below is a list of valid arguments, which are described in further detail in the following sections:

```
alesmith# show balancer ?

endpoints   show session load balancer endpoints
groups      show session load balancer service groups
lookup      show load balancer policy lookup table
members     show session load balancer cluster member summary
metrics     show load balancer metrics
policies    show load balancer policies
realms      show load balancer realms
statistics  show session load balancer statistics
tunnels     show session load balancer IP-in-IP tunnel info

alesmith#
```

show balancer endpoints

The **show balancer endpoints** command displays a full list of all IP-to-SBC mappings resident in the SLB. As the SLB can hold up to two million entries, the output of this command can and will grow very large, and extreme caution should be exercised when executing this command on a heavily trafficked SLB system.

```
alesmith# show balancer endpoints
IP address       Port Index    Address  Flags      SBC Handle
---------------- ----- -------- -------- ---------- ----------
14.0.134.236     5060 001e81e7 001e81e7 c0000000           94
14.0.134.232     5060 001e81eb 001e81eb c0000000           93
14.0.134.228     5060 001e81ec 001e81ec c0000000           92
14.0.134.224     5060 001e81ea 001e81ea c0000000           98
14.0.134.220     5060 001e81ee 001e81ee c0000000           97
14.0.134.216     5060 001e81ef 001e81ef c0000000           96
14.0.134.212     5060 001e81f1 001e81f1 c0000000           95
14.0.134.208     5060 001e81f4 001e81f4 c0000000           94
14.0.134.204     5060 001e81f0 001e81f0 c0000000           93
14.0.134.200     5060 001e81f3 001e81f3 c0000000           92
14.0.134.196     5060 001e81e7 001e81e7 c0000000           98
14.0.134.192     5060 001e81f9 001e81f9 c0000000           97
14.0.134.188     5060 001e81f6 001e81f6 c0000000           96
alesmith#
```

The table provided by **show balancer endpoints** displays every endpoint mapping. In the above example, note that IP addresses in the 14.0.134.0/24 space are being distributed among a number of SBCs. The *IP address* and *Port* columns pinpoint a specific endpoint. The *Index*, *Address*, and *Flags* columns contain SLB internal reference identifiers for locating that specific endpoint in memory. The *SBC Handle* column identifies which SBC serves that endpoint; use the **show balancer members** command to display a mapping of SBC names to SBC handles.

The **notify lbp save-stats** command saves the entire endpoint table to flash in a compressed (*xz*) format.

> **Note: notify lbp save-stats** may take several minutes to run, and should not be executed while the SLB is performing initial ramp-up of endpoint traffic, nor while the SLB is actively rebalancing.

You can use optional command arguments to filter/restrict command output.

show balancer endpoints <*ip-address*> restricts the display to one endpoint.

For example:

show balancer endpoints address 14.0.134.232

displays data for the specified IP endpoint

show balancer endpoints <*ip-address*>/<*:port_num*>> restricts the display to a specific port on a specific IP address.

For example:

show balancer endpoints address 14.0.134.232:5060

displays data for port 5060 on the specified endpoint

show balancer endpoints <*ip-address*>/<*bit-mask-len*> restricts the display to a contiguous range of endpoint addresses.

For example:

show balancer endpoints address 14.0.134.0/24

displays data for the 14.0.134.0 subnet

show balancer endpoints address 14.0.134.240/28

displays data for endpoint addresses 14.0.134.240 through 14.0.134.255

show balancer endpoints <*ip-address*>/<*bit-mask-len*><*:port_num*>> restricts the display to a specific port on a contiguous range of endpoint addresses.

001: 192.169.203.83:5060 [ref count = 4]

002: 192.169.203.83:5050 [ref count = 2]

alesmith# show balancer groups

show balancer endpoints address 14.0.134.0/24:5060

displays data for port 5060 on the specified subnet

show balancer groups

The **show balancer groups** command displays all of the configured *service groups* (groups of *service ports*, for example, a set of SIP ports on a SIP interface).

```
alesmith# show balancer groups
001: 192.169.203.83:5060 [ref count = 4]
002: 192.169.203.83:5050 [ref count = 2]
alesmith#
```

From left-to-right, the command displays:

- the group index
- the remote SBC IP address
- the remote port

show balancer members

The **show balancer members** command provides a list of all SBCs that have registered with the SLB.

```
alesmith# show balancer members
SBC Name                      Source IP  Dest IP  Slot Port Vlan Endpts
--- ------------------------- ---------- -------- ---- ---- ---- ------
 93 samadams@172.30.68.31     1.1.1.100  1.1.1.11    1    0    0  76592
 94 sixtus@172.30.68.36       1.1.1.100  1.1.1.16    1    0    0  76592
 95 bass@172.30.68.35         1.1.1.100  1.1.1.15    1    0    0  76592
 96 newcastel@172.30.68.37    1.1.1.100  1.1.1.12    1    0    0  76592
 97 magichat@172.30.68.34     1.1.1.100  1.1.1.14    1    0    0  76592
 98 guinness@172.30.68.33     1.1.1.100  1.1.1.13    1    0    0  76591
 99 westy                     1.1.1.100  1.1.1.19    1    0    0  76591

              max endpoints: 2000000
    max untrusted endpoints: 292777
          current endpoints: 526142
current untrusted endpoints: 186074
               surrent SBCs: 7
alesmith#
```

SBC contains the SBC handle, an internal shorthand that identifies a specific SBC. The **show balancer members** command provides a handle-to-hostname mapping.

Name contains the SBC hostname. Standalone SBCs are displayed as *hostname@eth0 IP address*, and highly available SBCs (*westy* in the above display) are displayed as *hostname*.

Source IP contains the local (SLB) tunnel address.

Destination IP contains the remote (SBC) tunnel address.

Slot, *Port*, and *Vlan* identify the local interface that supports the SLB-to-SBC tunnel.

Endpoints contains the number of endpoint-SBC associations that the SLB created for each specific SBC,

max endpoints contains the licensed capacity of the SLB.

max untrusted endpoints contains the maximum allowed number of untrusted endpoints.

current endpoints contains the current number of endpoints, trusted and untrusted

current untrusted endpoints contains the current number of untrusted endpoints.

show balancer metrics

The **show balancer metrics** command displays a comparison between the number of *local endpoints* (that is, the associations between source addresses and each SBC) and the number of *remote endpoints* (that is, what the SBC reports to the SLB as the number of endpoints it has received via the tunneled interface). Note that in the example output below those two numbers are the same; this is true if and only if there are no users in the access network that have multiple phone lines sourced from the same IP address. Were that the case, the number of remote endpoints would be higher than the number of local endpoints.

This table is populated with the data received in the periodic heartbeats from the SBC to the SLB. As these heartbeats are somewhat infrequent (every two seconds by default), the data in this table should only be considered accurate within two seconds.

```
alesmith# show balancer metrics
                              local  remote                   max
SBC Name                      epts   epts    max reg   CPU    CPU
--- ------------------------ ------ ------- -------- ----  ----
 93 magichat@172.30.68.34     0      0       480000    2.7  90.0
 94 westy                     0      0       480000    2.7  90.0
 95 samadams@172.30.68.33     0      0       480000    2.8  90.0
 96 bass@172.30.68.35         0      0       480000    4.3  90.0
 97 sixtus@172.30.68.36       0      0       480000    2.9  90.0
 98 newcastle@172.30.68.37    0      0       480000    2.9  90.0
 99 guiness@172.30.68.33      0      0       480000    3.6  90.0
alesmith#
```

SBC contains the SBC handle.

Name contains the SBC hostname.

max reg contains the maximum number of endpoints the SLB will send to this specific SBC. Its value is derived from the product of the **session-multiplier** parameter in the *cluster-config* configuration element and the SBC's licensed session capacity. The SBC passes this value to the SLB during the SBC's registration process into the cluster.

CPU contains the last received information on the CPU percentage from this SBC.

max CPU contains the threshold percentage at which the SBC is removed from consideration for the assignment of new endpoints. The default value is 90%, and may be changed on an SBC by setting the load-limit value as a SIP configuration option.

show balancer realms

The **show balancer realms** command displays a composite list of realms that all member SBCs have registered with the SLB.

```
alesmith# show balancer realms
Realm        SBC Tunnel Name                    ref count endpoints
---------- --- ------ ----------------------- --------- ---------
access       99   4092 newcastle@172.30.68.37           1     53535
access       98   4091 magichat@172.30.68.34            1     53535
access       97   4090 augustiner@172.30.68.41          1     53535
access       94   4086 bass@172.30.68.35                1     53535
access       93   4085 westy@172.30.68.42               1     53535
access       92   4084 sixtus@172.30.68.36              1     53535
access       96   4089 guiness@172.30.68.33             1     53535
net-13       99   4088 samadams@172.30.68.31            1     62550
net-13       91   4087 stbernie@172.30.68.43            1     62550
alesmith#
```

In this example, seven of the nine SBCs have registered the realm *access* and two have registered the realm *net-13*. The total number of endpoints for each of these services is indicated in the rightmost column. *ref count* is reserved for future use.

show balancer tunnels

The **show balancer tunnels** command generates a list of data for each tunnel between the SLB and its member SBCs. It includes the tunnel source and destination addresses, as well as an internal switch ID (swid) for this tunnel.

```
alesmith# show balancer tunnels
4090(32752)::
outer src ip = 182.16.203.83
outer dst ip = 182.16.203.87
slot/port/vlan = 0/1/0
trusted swid/pipe = 40/1213
untrusted swid/pipe = 41/1212
host swid/pipe = 42/1211
  service: 192.169.203.83:5050 [Realm192p1] protocols: 6/21588
  service: 192.169.203.83:5060 [Realm192p1] protocols: 17/21588

4091(32753)::
outer src ip = 182.16.203.83
outer dst ip = 182.16.203.86
slot/port/vlan = 0/1/0
trusted swid/pipe = 36/1216
untrusted swid/pipe = 37/1215
host swid/pipe = 38/1214
  service: 192.169.203.83:5060 [Realm192p1] protocols: 17/21588

alesmith#
```

Use the error argument for error reporting and troubleshooting.

```
alesmith# show balancer tunnels errors
src ip 182.16.203.83 / dst ip 182.16.203.87 / slot 0 / port 1 / vlan 0:
  IP:Port:  192.169.203.83:5050
  Proto Encaps Errors Decaps Errors
  ----- ------------ ------------
  6                0            0
  IP:Port:  192.169.203.83:5060
  Proto Encaps Errors Decaps Errors
  ----- ------------ ------------
  17               0            0

src ip 182.16.203.83 / dst ip 182.16.203.86 / slot 0 / port 1 / vlan 0:
  IP:Port:  192.169.203.83:5060
  Proto Encaps Errors Decaps Errors
  ----- ------------ ------------
  17               0            0


unknown protocol:      0
do not fragment drops: 0
no matching tunnel:    0
service lookup failed: 0
IP frag msg failure:   0
mblk alloc failues:    0
alesmith#
```

Use the fragments argument for information related to packet fragmentation/ reassembly details.

```
alesmith# show balancer tunnels fragments
src ip 182.16.203.83 / dst ip 182.16.203.87 / slot 0 / port 1 / vlan 0:
  IP:Port:  192.169.203.83:5050
  Proto Encaps Fragmented Decaps Reassembled
  ----- ---------------- ----------------
  6                    0                0
  IP:Port:  192.169.203.83:5060
  Proto Encaps Fragmented Decaps Reassembled
  ----- ---------------- ----------------
  17                   0                0

src ip 182.16.203.83 / dst ip 182.16.203.86 / slot 0 / port 1 / vlan 0:
  IP:Port:  192.169.203.83:5060
  Proto Encaps Fragmented Decaps Reassembled
  ----- ---------------- ----------------
  17                   0                0

MTU = 1500

packets reassembled:    0
fragments too small:    0
fragment len not mod 8: 0
too many fragments:     0
reassembly timeouts:    0
duplicate fragments:    0
mbuf alloc failed:      0
gap in reassembly:      0
IP hdr checksum errors: 0
overlapping frags:      0
packet too long:        0
multi-buffer frames:    0
alesmith#
```

Use the statistics argument for information related to packet counts.

```
alesmith# show balancer tunnels statistics
src ip 182.16.203.83 / dst ip 182.16.203.87 / slot 0 / port 1 / vlan 0:
  IP:Port:  192.169.203.83:5050
  Proto Encap Pkts Encap Octets Decap Pkts Decap Octets
  ----- ---------- ------------ ---------- ------------
  6              0            0          0            0
  IP:Port:  192.169.203.83:5060
  Proto Encap Pkts Encap Octets Decap Pkts Decap Octets
  ----- ---------- ------------ ---------- ------------
  17         48011     24213914          0            0

src ip 182.16.203.83 / dst ip 182.16.203.86 / slot 0 / port 1 / vlan 0:
  IP:Port:  192.169.203.83:5060
  Proto Encap Pkts Encap Octets Decap Pkts Decap Octets
  ----- ---------- ------------ ---------- ------------
  17         48017     24217918          0            0


alesmith#
```

show balancer statistics

The **show balancer statistics** command displays statistical output pertinent to low-level events on the SLB. The contents and output of this command are subject to change, and will be documented in a subsequent document release.

```
alesmith# shower-balancer-statistics
tLBP not initialized drops   0
max capacity reached drops   2
endpoint SBC mismatch errors 0
endpoint table read errors   0
Tx packet failed count       0
service not found count      0
duplicate ept packet drops   0
msgq drops                   0
forwarded duplicates         0
policy miss                  40508
realm miss                   0
throttle drops               0
throttle skips               0
total packets processed      40508
EPT delete errors            0
EPT add errors               0
EPT update errors            0
group not found              0
LBP agent not found          0
invalid endpoint             0
insert error                 0
untrusted dropped            0
untrusted age outs           171
packets dropped in balance   0
packets dropped by standby   0
-----------------------------
trusted endpoints (EPT db)   0
untrusted endpoints (EPT db) 337
-----------------------------
total trusted endpoints      0
total untrusted endpoints    337
total endpoints              337
alesmith#
```

CCP Statistical Output

The Cluster Control Protocol (CCP) also provides the operator with a full set of statistical data for troubleshooting and diagnostic purposes.

The **show ccd** command is the root of all statistical data pertinent to CCP operation. Below is a list of valid arguments, which are described in further detail in the following sections:

```
alesmith# show ccd ?

ccp         Cluster Control Protocol Stats
reset       Reset Stats
sds         Controlled SDs
stats       Cluster Control Stats
alesmith#
```

show ccd ccp

The **show ccd ccp** command displays data about specific Cluster Control Protocol operations.

```
alesmith# show ccd ccp
--------------------
M01:0
--------------------

Svc Add               Recent      Total  PerMax
==================   ======  =========  ======
Ops Recvd                  0          8       4
Op Replies Sent            0          8       4

                  -------- Received -------   ---------- Sent --------
Status Code         Recent     Total  PerMax   Recent     Total  PerMax
-----------           ------  ---------  ------   ------  ---------  -
-----
200 OK                   0         0      0       0         8      4

EP Del                Recent      Total  PerMax
==================   ======  =========  ======
Ops Recvd                  0       3984    1013
Duplicate Ops              0         12       6
Op Replies Sent            0       3984    1013

                  -------- Received -------   ---------- Sent --------
Status Code         Recent     Total  PerMax   Recent     Total  PerMax
-----------           ------  ---------  ------   ------  ---------  -
-----
200 OK                   0         0      0       0      3984    1013

EP Promo              Recent      Total  PerMax
==================   ======  =========  ======
Ops Recvd                  0     115601    8187
Duplicate Ops              0        329      23
Op Replies Sent            0     115601    8187
```

```
                  -------- Received -------  ---------- Sent --------
Status Code        Recent     Total  PerMax   Recent     Total  PerMax
-----------          ------  ---------  ------   ------  ---------  -
200 OK                  0         0      0       0     115601    8187

Metrics              Recent      Total  PerMax
==================   ======  =========  ======
Ops Recvd                57      16330      75
Op Replies Sent          57      16331      75

                  -------- Received -------  ---------- Sent --------
Status Code        Recent     Total  PerMax   Recent     Total  PerMax
-----------          ------  ---------  ------   ------  ---------  -
200 OK                  0         0      0      57      16328     75
406 Not Accept           0         0      0       0          3      3

Prov Done            Recent      Total  PerMax
==================   ======  =========  ======
Ops Recvd                0          6       3
Op Replies Sent          0          6       3

                  -------- Received -------  ---------- Sent --------
Status Code        Recent     Total  PerMax   Recent     Total  PerMax
-----------          ------  ---------  ------   ------  ---------  -
200 OK                  0         0      0       0          6      3

Going Down           Recent      Total  PerMax
==================   ======  =========  ======
Ops Recvd                0          1       1
Op Replies Sent          0          1       1

                  -------- Received -------  ---------- Sent --------
Status Code        Recent     Total  PerMax   Recent     Total  PerMax
-----------          ------  ---------  ------   ------  ---------  -
200 OK                  0         0      0       0          1      1

Stop Down            Recent      Total  PerMax
==================   ======  =========  ======
Ops Recvd                0          8       6
Op Replies Sent          0          9       6

                  -------- Received -------  ---------- Sent --------
Status Code        Recent     Total  PerMax   Recent     Total  PerMax
-----------          ------  ---------  ------   ------  ---------  -
200 OK                  0         0      0       0          6      3
406 Not Accept           0         0      0       0          3      3

alesmith#
```

show ccd sds

The **show ccd sds** command displays a table containing an overview of all of the data gleaned from the CCP from each SBC.

```
alesmith# show ccd sds
Session Director        Hdl  State     Tunnel Svcs Version     HW
----------------------- ---- --------- ------ ---- ---- ------ -----
augustiner@172.30.68.41   95 InService  2/2     2  6.2.0.30b8  SD3
bass@172.30.68.35         94 InService  2/2     2  6.2.0.30b8  SD3
guinness@172.30.68.33     96 InService  2/2     2  6.2.0.30b8  SD3
magichat@172.30.68.34     97 InService  2/2     2  6.2.0.30b8  SD3
newcastel@172.30.68.37    98 InService  2/2     2  6.2.0.30b8  SD3
samadams@172.30.68.31     99 InService  2/2     2  6.2.0.30b8  SD3
sixtus@172.30.68.36       92 InService  2/2     2  6.2.0.30b8  SD3
stbernie@172.30.68.43     91 InService  2/2     2  6.2.0.30b8  SD3
westy@172.30.68.42        93 InService  2/2     2  6.2.0.30b8  SD3
alesmith#
```

Session Director contains the hostname of the cluster SBCs that are connected to the SLB. As with all of the similar statistical output, standalone SBCs are displayed as *hostname@eth0 IP address*, and highly available SBCs are displayed as *hostname*.

Hdl contains the clustered SBC handle, an internal shorthand that identifies a specific cluster member. The **show balancer members** command provides a handle to hostname mapping.

State contains the current SBC state. Valid states are:

- *Init* — during initial handshaking with the SLB
- *InService* — healthy and operating normally
- *Rebalance* — during a cluster expansion/contraction operation
- *LostControl* — no longer communicating with the SLB

Tunnel contains the number of tunnels between the SBC and SLB.

Svcs contains the number of advertised services (protocols) that the SBC has negotiated with the SLB.

Version contains the software version running on that SBC.

HW identifies the hardware platform (in this case, SD3 identifies an Acme Packet Net-Net 4500 SBC).

LastPing is not currently used.

When issued with an optional *hostname* argument, the **show ccd sds** command provides a detailed report for the target hostname.

```
alesmith# show ccd sds bass
Session Director: bass@172.30.68.35
+--------------------------------------------------------------------
|State     : InService       Handle       : 0x5f
|Tunnels   : 1               ServicePorts : 1
|HW Type   : SD3             SW Version   : 6.2.0.30b8
|Last Ping : 1080ms          App Count    : 1
|
|Service:        App  SvcPorts Tunnels Endpoints DropCount
+--------------- ---- -------- ------- --------- ---------
access           SIP         1       1    285714         0
|
|#   Tunnel                             App  Handle     Svcs LastHB
+--- ---------------------------------- ---- ---------- ---- -----
|0   (1.1.1.100|1.1.1.15)               SIP  0xffb         1  375ms
|
|#   CPU   MAX   CurReg    RegLimit  CurSess   MaxSess
+--- ----- ----- --------- --------- --------- ---------
|0    4.1% 90.0%    285714         0      7336     64000
|     4.1% 90.0%    285714    960000      7736     64000
|
|Service Port                        App  Handle      TunNdx Avail
+----------------------------------- ---- ---------- ------ -----
|access::192.168.168.100:5060<17>       H248     513(1)      0   yes
alesmith#
```

SBC State

State — the current SBC state

Handle — the SBC handle

Tunnels — the current number of SBC tunnels

ServicePorts — the current number of SBC service ports

HW Type — the hardware platform (in this case, SD3 identifies an Acme Packet Net-Net 4500 SBC)

SW Version — the installed software revision level

Last Ping — the number of elapsed milliseconds, since a ping/keepalive was received from this SBC

App Count — the number of applications supported by the SBC

Services State

Service — the realm advertised by the SBC in the Service Port ID

App — the supported protocol: SIP or H.248

SVCPorts — the current number of service ports

Tunnels — the current number of tunnels

Endpoints — the cumulative number of endpoints for this service

DropCount — the number of elements to drop when rebalancing this SBC

Tunnel State

#	—	the tunnel index (0 or 1)
Tunnel	—	the SLB and SBC tunnel IP address
App	—	the supported protocol (SIP orH.248)
Handle	—	the handle for the tunnel
Svcs	—	the number if service ports supporting the tunnel
LastHB	—	the number of elapsed milliseconds since a heartbeat was received from the remote end of this tunnel

Tunnel Metrics

#	—	the tunnel number (0 or 1)
CPU	—	the current CPU utilization rate
Max	—	the maximum supported CPU utilization rate, if this value is exceeded, the tunnel implements a load limit algorithm
CurReg	—	the current number of registrations supported by the SBC
regLimit	—	the maximum number of registrations supported by the SBC
CurSess	—	the current call count reported by the SBC
MaxSess	—	the maximum sessions for which the SBC is licensed

Service Port Data

Service Port	—	the service path (the concatenation of realm, IP address, port number, and IP Level 4 protocol number — 17 for UDP, 6 for TCP)
App	—	the supported protocol (SIP orH.248)
Handle	—	the handle for the service port
TunNdx	—	the tunnel the service port is registered for
Avail	—	current availability (yes or no) determined by the presence of heartbeats

show ccd stats

The **show ccd stats** command displays endpoint statistics for the SBC members of the cluster.

```
alesmith# show ccd stats
17:10:09-54
                          -------- Period -------    ---- LifeTime ----
 SD                      Active  Rate   High Total  Total PerMax    High
bass@172.30.68.35       I285714   0.0 285714    0 285.71K 13.76K 285.71K
guinness@172.30.68.33   I285714   0.0 285714    0 285.71K 13.76K 285.71K
magichat@172.30.68.34   I285714   0.0 285714    0 285.71K 13.76K 285.71K
newcastel@172.30.68.37  I285714   0.0 285714    0 285.71K 13.76K 285.71K
samadams@172.30.68.31   I285714   0.0 285714    0 285.71K 13.76K 285.71K
sixtus@172.30.68.36     I285714   0.0 285714    0 285.71K 13.76K 285.71K
westy                   I285714   0.0 285714    0 285.71K 13.76K 285.71K

Total Endpoints: 153908
Total SDs      : 9
alesmith#
```

The *Period* stats provided represent an accumulation of data for the amount of time specified after the dash separator in the timestamp printed in the first line of output (in this example, the period represents 54 seconds).

The single ASCII character between the *SD* column and the *Active* column is the state of that SBC; the letter I represents *InService*.

The *Rate* column displays the transmission rate of *new* endpoint associations to that particular SBC. (In the sample, no new endpoints are arriving in the cluster, so all of the SBCs show a rate of 0.0.) The High field indicates the highest number of active endpoint associations for the current period.

When issued with an optional *hostname* argument, the **show ccd stats** command provides a detailed report for the target hostname.

```
alesmith# show ccd stats bass
15:09:25-59
SD bass@172.30.68.33 [InService]
State                 -- Period -- -------- Lifetime --------
            Active    High   Total      Total Permax     High
Tunnels          1       1       0          2      1        1
Service Ports    2       2       0          2      1        2
Endpoints    53571   53571       0      53571  14399    53571
Contacts     53571   53571       0      53571  14399    53571
Sessions         0       0       0          0      0        0
Remote CPU       0       0       0          0      0        0
SD Something        ---- Lifetime ----
                      Recent      Total  PerMax
Heartbeats rcvd         30      27426      15
Heartbeats Missed        0          1       1
Tunnel Adds              0          2       1
Tunnel Removes           0          1       1
Service Adds             0          2       1
Service Removes          0          0       0
Endpoint Removes         0          0       0
Endpoint Promotes        0      53571    13561
Endpoints Skipped        0          0        0
Rebalance Source         0          0        0
Rebalance Target         0          0        0
Rebalance Request        0          0        0
Rebalance Replies        0          0        0
CPU Above Limit          0          0        0
CPU Above Threshold      0          0        0
Online Transitions       0          1        1
Offline transitions      0          1        1
Tunnel Add Fails         0          0        0
CCD Tunnel Add Fails     0          0        0
Tunnel Remove Fails      0          0        0
CCD Tunnel Remove Fails  0          0        0
Service Add Fails        0          0        0
CCD Svc Add Fails        0          0        0
Service Remove Fails     0          0        0
CCD Svc Remove Fails     0          0        0
Service Adds No Cfg      0          0        0
Bad Service Handle       0          0        0
Endpoint Remove Fails    0          0        0
Endpoint Prom Fails      0          0        0
alesmith#
```

The *Period* stats provided represent an accumulation of data for the amount of time specified after the dash separator in the timestamp printed in the first line of output (in this example, the period represents 59 seconds).

Tunnels contains the number of tunnels between the SLB and the target SBC, in this case, *bass*.

Service Ports contains the number of Service Ports advertised by the target SBC when it joined the cluster.

Endpoints and *Contacts* contain the number of endpoint associations the SLB has assigned to the target SBC. If there is only one registering device at a given endpoint, a one-to-one correlation between endpoints and contacts is expected. However, if the **atom-limit-divisor** parameter has been set to a non-default value, the number of contacts exceeds the number of endpoints.

Sessions contains the number of active calls.

The table below the overview data displays specific Cluster Control Protocol message statistics.

HeartBeats rcvd contains the number of heartbeat/keepalive messages received from the target SBC. Heartbeats are sent every two seconds by the SBC.

HeartBeats Missed contains the number of scheduled heartbeat/keepalive messages not received from the target SBC.

The *Tunnel Adds* and *Tunnel Removes* counters are incremented when an SBC joins the cluster and leaves the cluster, respectively.

The *Service Adds* and *Service Removes* counters are incremented when an SBC advertises support for a service and withdraws support for a service, respectively. This generally happens only when an SBC first joins the cluster, or if the configuration on a clustered SBC is changed, saved, and activated.

The *Endpoint Removes* counter tracks the number of SBC-originated Cluster Control messages that request the SLB to delete a forwarding rule. Such a request can be the result of (1) a rebalance operation (when the SLB asks for the SBC to nominate candidates for rebalancing), (2) an endpoint de-registration with the SBC, or (3) an endpoint is power down. Generally, whenever a registration cache entry on a clustered endpoint is removed by the SBC, it notifies the SLB to remove that binding.

The *Endpoint Promotes* counter tracks the number of promotion messages the SBC sends to the SLB to validate an untrusted forwarding rule. When the SLB first creates a forwarding rule for a new endpoint, it treats it as untrusted. When the SBC receives a 200 OK for a REGISTER message from that endpoint's registrar, the SBC sends a Promote Cluster Control message to the SLB. At this point, the SLB modifies the particular forwarding rule and assigns it *trusted* status. If this Promote message is not received within the time configured as the untrusted-grace-time in the *lbp-config*, the SLB deletes the untrusted entry.

Endpoints Skipped contains the number of endpoints in its registration cache that the SBC has skipped over during a rebalance request. Skipping may be done for one of two reasons: either the most appropriate user for rebalancing was in an active phone call (and **rebalance-skip-calls** was enabled in *cluster-config*), or the **rebalance-skip-ahead** value in *cluster-config* was set to a nonzero value. In this case, when the SBC is asked to nominate users for rebalance, it will skip over any users whose registration cache entry is due to expire within the number of milliseconds set as the **rebalance-skip-ahead** value.

Rebalance Source contains the number of times the target SBC was used as a source of endpoints during a rebalance operation (that is, it supplied endpoints to a cluster member that was added to the cluster after itself).

Rebalance Target contains the opposite: the number of times that SBC was the recipient of endpoints from other sources during a rebalance operation.

The *Rebalance Requests* and *Rebalance Replies* counters increment upon receipt of a Cluster Control message from the SLB to the SBC asking it to divest itself of endpoints, and the responsive Cluster Control message from the SBC that indicates the endpoints the SBC has chosen.

The *CPU Above Limit* and *CPU Above Threshold* counters increment whenever an SBC has reported a high CPU value, and has been taken out of consideration for new endpoint assignments. Generally, the CPU limit and threshold are the same value (90%). However, it is possible to configure the threshold to be lower using the sip-config **option load-limit**.

Session Load Balancer SNMP Reference

Overview

This chapter provides an overview of SNMP support for Session Load Balancer (SLB) features.

Enterprise Traps

The following table identifies the SLB proprietary traps that the Net-Net system supports.

apSLBEndpointCapacityThresholdTrap	Generated when the number of endpoints on the SLB exceeds the configured threshold.
apSLBEndpointCapacityThresholdClearTrap	Generated when the number of endpoints on the SLB falls below the configured threshold.
apSLBUntrustedEndpointCapacityThresholdTrap	Generated when the number of untrusted endpoints on the SLB exceeds the configured threshold.
apSLBUntrustedEndpointCapacityThresholdClearTrap	Generated when the number of untrusted endpoints on the SLB falls below the configured threshold.

Acme Packet License MIB (ap-license.mib)

SNMP GET Query Name	Object Identifier Name: Number	Description
Object Identifier Name: apLicenseEntry (1.3.6.1.4.1.9148.3.5.1.1.1)		
apLicenseSLBEndpointCap	apLicenseEntry: 1.3.6.1.4.1.9148.3.5.1.1.1.23	SLB endpoint capacity

Acme Packet Session Load Balancer MIB (ap-slb.mib)

SNMP GET Query Name	Object Identifier Name: Number	Description
Object Identifier Name: apSLBMIBObjects (1.3.6.1.4.1.9148.3.11.1)		
Object Identifier Name: apSLBMIBGeneralObjects (1.3.6.1.4.1.9148.3.11.1.1)		
apSLBStatsEndpointsCurrent	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.1	Number of endpoints currently on the SLB.
apSLBStatsEndpointsDenied	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.2	Number of endpoints denied by the SLB because the system has reached the maximum endpoint capacity.
apSLBEndpointCapacity	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.3	Maximum number of endpoints allowed on the SLB. This value is based on the installed SLB license(s).
apSLBEndpointCapacityUpperThresh	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.4	Percentage of endpoints relative to maximum threshold capacity.
apSLBEndpointCapacityLowerThresh	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.5	Percentage of endpoints relative to minimum threshold capacity.
apSLBStatsUntrustedEndpointsCurrent	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.6	Number of untrusted endpoints currently on the SLB.
apSLBStatsTrustedEndpointsCurrent	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.7	Number of trusted endpoints currently on the SLB.
apSLBStatsUntrustedEndpointsDenied	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.8	The number of untrusted endpoints denied by the SLB due to the total number of untrusted endpoints exceeding the configured maximum threshold.
apSLBStatsUntrustedEndpointsAgedOut	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.9	The number of untrusted endpoints aged out of the system because they were not authenticated within the configured grace period.
apSLBUntrustedEndpointCapacity	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.10	Maximum number of untrusted endpoints allowed on the SLB. This value is a configured percentage of the maximum endpoint capacity of the system.
apSLBUntrustedEndpointCapacityUpperThresh	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.11	Percentage of untrusted endpoint maximum threshold capacity in use.
apSLBUntrustedEndpointCapacityLowerThresh	apSLBMIBGeneralObjects: 1.3.6.1.4.1.9148.3.11.1.1.12	Percentage of untrusted enpoint minumum threshold capacity percentage.