EQUALIZER

Equalizer
Installation and
Administration Guide

Version 8.5
July 2009

CoyotePoint
Systems Inc

Contents

Configuring Equalizer Operation 43

Using Match Rules 159

Administering GeoClusters 187

Using Certificates in HTTPS Clusters 227

Equalizer VLB 241

Equalizer Installation and Administration Guide

Preface

The *Equalizer Installation and Administration Guide* tells you how to install, configure, and maintain Equalizer™ load balancers running Release 8 of the Equalizer software.

In This Guide

This guide contains the following chapters and appendices:

- Chapter 1*, Equalizer Overview,* contains detailed descriptions of Equalizer concepts and terminology. This chapter includes information to help you plan your Equalizer configuration. If you are setting up Equalizer for the first time, be sure to read the *Overview* chapter before attempting to install and configure your system.

- Chapter 2*, Installing and Configuring Equalizer Hardware,* provides comprehensive instructions for installing Equalizer hardware and setting up Equalizer to work with your networks and servers.

- Chapter 3, *Using the Administration Interface*, discusses how to use Equalizer's HTML-based administration interface, including adding administrative logins with distinct permissions.

- Chapter 4, *Configuring Equalizer Operation*, tells you how to configure through the Equalizer Administration Interface, including setting up a failover configuration.

- Chapter 5, *Administering Virtual Clusters*, tells you how to add and remove virtual clusters and servers, changing load balancing options, and shutting down servers.

- Chapter 6, *Monitoring Equalizer Operation*, describes how to view information, statistics, and graphical displays about Equalizer's operation.

- Chapter 7, *Using Match Rules*, shows you to create match rules that distribute requests based on a request's attributes.

- Chapter 8, *Administering GeoClusters*, shows you how to use the Envoy add-in to add and remove geographic clusters and sites and change geographic load balancing and targeting options.

- Appendix A, *Server Agent Probes*, describes how to develop custom server agents.

- Appendix B, *Timeout Configuration*, provides detailed desccriptions of all the timeout parameters used by Equalizer and the effect that each has on client/server connections and server probes.

- Appendix C, *Using Reserved IP Addresses* describes how to configure Equalizer to distribute requests to servers assigned IP addresses on reserved, non-routable networks.

- Appendix D, *Regular Expression Format* discusses Equalizer's regular expressions, components, formats, and usage.

- Appendix E, *Using Certificates in HTTPS Clusters* shows you how to obtain and install certificates for HTTPS clusters.

- Appendix G, *Troubleshooting* helps you to diagnose problems with Equalizer.

- Appendix H, *License and Warranty* contains the complete License and Warranty information.

- Appendix I, *Additional Requirements and Specifications* lists additional hardware related requirements for Equalizer installations.

- Appendix F, *Equalizer VLB* describes the optional Equalizer VLB product, which supports integration of Equalizer with VMware Infrastucture and ESX Server virtual machine configurations.

-

-

- The *Glossary* defines the technology-specific terms used throughout this book.

- Use the *Index* to help find specific information in this guide.

Typographical Conventions

The following typographical conventions appear throughout this guide:

Italics indicates the introduction of new terms, is used to emphasize text, and indicates variables and file names.

Boldface text highlights command names in instructions and text entered by the user. **Boldface** text highlights graphical administrative interface screen elements: labels, buttons, tabs, icons, etc.

`Courier` text is used to denote computer output: messages, commands, file names, directory names, keywords, and syntax exactly as displayed by the system.

Sequences such as "***Equalizer* > Status > Event Log**" are used to indicate the Administrative Interface click-path the user should follow to display the information or form relevant to the task at hand. In this example, the user would click on the Equalizer system name displayed on the left side of the Administrative Interface, then click on the **Status** tab on the right side of the screen, and then click on the **Event Log** sub-tab. Similarly, "***Cluster* > Probes**" starts by selecting a cluster name in the left frame tree, and ***"Server* > Reporting**" starts with a server name.

1. Numbered lists show steps that you must complete in the numbered order.

   - Bulleted lists identify items that you can address in any order.

Note – Highlights important information and special considerations.

Caution – Warns when an action could result in loss of data or damage to your equipment.

⚠ Emphasizes safety information or information critical to Equalizer operation.

Where to Go for More Help

Customer Support contact information is available from the **Support** link on our main web page at **http://www.coyotepoint.com**. Register today for access to the **Coyote Point Support Portal** at:

    http://support.coyotepoint.com

Registration provides you with a login so you can access these benefits:

- **Support FAQ**: answers to our customer's most common questions.

- **Moderated Customer Support Forum**: ask questions and get answers from our support staff and other

Chapter 1:

Equalizer Overview

Introducing Equalizer

Equalizer® is a high-performance content switch that features:

- Intelligent load balancing based on multiple, user-configurable criteria

- Non-stop availability with no single point of failure, through the use of redundant servers in a cluster and the optional addition of a failover (or backup) Equalizer

- Layer 7 content-sensitive routing

- Connection persistence using cookies or IP addresses

- Real-time server and cluster performance monitoring

- Server and cluster administration from a single interface

- SSL acceleration (on Equalizer models with Xcel SSL Hardware Acceleration)

- Data compression (on Equalizer models with Express Hardware GZIP Compression)

- Geographic load balancing (with the optional Envoy software add-on)

This document describes the features and capabilities of the Equalizer units available at the time this document was prepared. For a current list of products and features, please visit Coyote Point's website at `www.coyotepoint.com`.

Intelligent Load Balancing

Equalizer functions as a *gateway* to one or more sets of *servers* organized into *virtual clusters*. When a client submits a request to a site that Equalizer manages, Equalizer identifies the virtual cluster for which the request is intended, determines the server in the cluster that will be best able to handle the request, and forwards the request to that server for processing.

To route the request, Equalizer modifies the header of the request packet with the appropriate server information and forwards the modified packet to the selected server. Depending on the cluster options chosen, Equalizer may also modify the headers in server responses on the way back to the client.

Equalizer support clusters that route requests based on either *Layer 4* (TCP or UDP) or *Layer 7* (HTTP or HTTPS) protocols. Layer 4 is also referred to as the *Transport Layer*, while Layer 7 is referred to as the *Application Layer*. These terms come from the OSI and TCP/IP Reference Models, abstract models for network protocol design.

In general, Layer 4 clusters are intended for configurations where routing by the destination IP address of the request is sufficient and no examination of the request headers is required. Layer 7 clusters are intended for configurations where routing decisions need to be made based on the content of the request headers. Equalizer evaluates and can modify the content of request headers as it routes packets to servers; in some cases, it can also modify headers in server responses on their way back to the client.

The table below summarizes the basic capabilities of the cluster types supported by Equalizer.

Feature	Cluster Type			
	L4 UDP	L4 TCP	L7 HTTP	L7 HTTPS
Load balancing policies	round robin, static weight, adaptive, fastest response, least connections, server agent			
Server failure detection (probes)	ICMP, TCP, Server Agent	ICMP, TCP, ACV, Server Agent		
Persistence	Based on IP		Using Cookies	
Server selection by request content (i.e., Match Rules)	No; load is balanced according to current load balancing policy.		Yes; load is balanced according to decisions made by examining request content.	
Load balanced protocols	Ideal for stateless UDP-based protocols, such as DNS and RADIUS; WAP gateways; NFS server clusters that provide a single-system image.	Ideal for stateful TCP-based protocols, such as HTTP, HTTPS, SMTP, FTP, LDAP/LDAPS[a] and others.	HTTP	HTTPS
NAT and spoofing	Yes			

a. Note that some LDAP/LDAPS implementations are UDP-based.

Regardless of cluster type, Equalizer uses intelligent *load balancing algorithms* to determine the best server to receive a request. These algorithms take into account the configuration options set for the cluster and servers, real-time server status information, and information from the request itself. For Layer 7 clusters, user-defined match rules can also be used to determine the route a packet should take.

Load Balancing Configuration

When you configure a virtual cluster, you can select one of the following load-balancing algorithms to control how Equalizer balances the load across your servers: **round robin**, **static weight**, **adaptive**, **fastest response**, **least connections**, **server agent**, or **custom**.

When you configure the servers in a virtual cluster, you assign an *initial weight* between 20 and 200 for each server. When you select one of the adaptive load-balancing algorithms (i.e., any algorithm other than round robin), Equalizer uses the servers' initial weights as a starting point to determine the percentage of requests to route to each server. Each server handles a percentage of the total load based on its fraction of the total weights in the server cluster. Equalizer dynamically adjusts server weights according to real-time conditions to ensure that Equalizer routes requests to the server that is best able to respond. A server with a weight of zero (0) is considered down or unavailable, and Equalizer does not route new requests to servers in this state.

Real-Time Server Status Information

Equalizer gathers real-time information about a server's status using ICMP Probes, TCP Probes, Active Content Verification (*ACV),* and Server Agents. ICMP and TCP Probes are the default probing methods.

ICMP Probes uses the Internet Control Message Protocol to send an "Echo request" to the server, and then wait for the server to respond with an ICMP "Echo reply" message (like the Unix **ping** command). ICMP is a Layer 3 protocol. ICMP probes can be disabled via a global flag.

TCP Probes establish (and tear down) a TCP connection between Equalizer and the server, in a typical Layer 4 exchange of TCP SYN, ACK, and FIN packets. If the connection cannot be completed, Equalizer considers the server down and stops routing requests to it. TCP probes cannot be disabled.

Equalizer's *Active Content Verification (ACV)* provides an optional method for checking the validity of a server's response using Layer 7 network services that support a text-based request/response protocol, such as HTTP. When you enable ACV for a cluster, Equalizer requests data from each server in the cluster (using an *ACV Probe string*) and verifies the returned data (against an *ACV Response string)*. If Equalizer receives no response or the response string is not in the response, the verification fails and Equalizer stops routing new requests to that server. (Note that ACV is not supported for Layer 4 UDP clusters.) For more information, see "Using Active Content Verification (ACV)" on page 93.

Server Agent Probes are an optional feature that enable Equalizer to communicate with a user-written program (the *agent*) running on the server. A server agent is written to open a server port and, when Equalizer connects to the port, the server agent responds with an indication of the current server load and performance. This enables Equalizer to adjust the dynamic weights of the server according to detailed performance measurements performed by the agent, based on any metrics available on the server. If the server is overloaded and you have enabled **server agent** load balancing, Equalizer reduces the server's dynamic weight so that the server receives fewer requests. The interface between Equalizer and server agents is simple and well-defined. Agents can be written in any language supported on the server (e.g., perl, C, shell script, javascript, etc.). For more information see "Server Agent Probes" on page 205.

For those who have one or more VMware ESX Servers, *Equalizer VLB* can be configured to use VMware's status reporting to determine server status, and can also be configured to automatically manage VMware servers based on status information obtained from VMware. For more information, see Appendix F, "Equalizer VLB".

Network Address Translation and Spoofing

Equalizer's *Network Address Translation (NAT)* subsystem implements the *spoofing* feature for Layer 4 and Layer 7 clusters. When Equalizer receives a request destined for a cluster with the spoof option enabled, the NAT subsystem rewrites the TCP/UDP and IP headers of the request packet -- using the client IP address as the source IP address. For this reason, the servers in a cluster with spoofing enabled must be configured to use Equalizer as the default gateway, to ensure that all responses go through Equalizer (otherwise, the server would attempt to respond directly to the client IP). Equalizer keeps a record of the address translation performed, along with the cluster and server IP, before forwarding the translated packet to the selected server. When the server responds to the request, Equalizer performs the reverse translation on the response packets before forwarding them to the client -- using the cluster IP address as the source address in the packets. This is necessary since the client sent its original request to the cluster IP and will not recognize the server's IP address as a response to its request -- instead, it will drop the packet.

When IP spoofing is disabled, Equalizer uses it's internal IP address as the source IP for all packets it sends on to the servers in the cluster. The servers send responses directly back to Equalizer's internal IP. When Equalizer receives the packets, the NAT subsystem translates the source IP in the response packets (that is, the server IP) to the cluster IP to which the client originally sent the request.

NAT can also be used for outbound packets, that is packets going from servers to clients in response to client requests. When outbound NAT is enabled, Equalizer translates the source IP in the response packets from the server IP address to the Equalizer's external interface IP address. This is usually required in dual network mode when reserved IP addresses (e.g., 10.x.x.x, 192.168.x.x) are being used on the internal interface, so that clients do not see reserved IP addresses in server responses. Enabling outbound NAT, as a result, has a performance cost since Equalizer is modifying every outbound packet.

When Equalizer is in single network mode, outbound NAT should be disabled (the default) -- since Equalizer resides on a single subnet, outbound NAT is not needed, and may cause unexpected behavior.

For more information about NAT and spoofing options, see "Working with Virtual Clusters" on page 74.

Note that when Equalizer receives a packet that is not destined for a virtual cluster IP address, a failover IP address, a client IP address on an open connection, or one of its own network interface IP addresses, Equalizer passes the packet through to the network unaltered.

Maintaining Persistent Sessions and Connections

The *persistence* of *session data* is important when a client and server need to refer to data previously generated again and again as they interact over more than one transaction, possibly more than one connection. Whenever a client places an item in a shopping cart, for example, session data (the item in the cart, customer information, etc.) is created that potentially needs to persist across many individual TCP connections before the data is no longer needed and the session is complete.

It's important to note that *session persistence* is managed by the server application, not Equalizer. Equalizer provides *server persistence* so that a *persistent connection* between a particular client and a particular server can be maintained; this supports a client-server session where session data is being maintained on the server for the life of the connection. In other words, whether you need to enable persistence on Equalizer depends on the application you are load balancing.

Equalizers have no knowledge of the fact that the user has placed something in a shopping cart, logged into a web application, requested a file from shared storage, or made a "post" in a front end presentation server that has been written to a database. Basically, a "state" has been created in the load balanced application of which Equalizer is not aware. What Equalizer *does* know is that a specific client has been load balanced to a specific server in one of its virtual clusters. With this knowledge, Equalizer can track that information and send that client back to the same server they were connected the first time.

Equalizer provides server or connection persistence using cookies in Layer 7 HTTP and HTTPS clusters, and using the client IP address in Layer 4 TCP and UDP clusters. The following sections explain connection persistence provided by Equalizer, and its relationship to session persistence.

Cookie-Based Persistence (Layer 7)

Equalizer can use cookie-based persistent connections for Layer 7 HTTP and HTTPS clusters. In cookie-based persistence, Equalizer "stuffs" a cookie into the server's response header on its way back to the client. This cookie uniquely identifies the server to which the client was just connected. The client includes (sends) the cookie in subsequent requests to the Equalizer. Equalizer uses the information in the cookie to route the requests back to the same server.

Equalizer can direct requests from a particular client to the same server, even if the connection is to a different virtual cluster. For example, if a user switches from an HTTP cluster to an HTTPS cluster, the persistent cookie will still be valid if the HTTPS cluster contains a server with the same IP address.

If the server with which a client has a persistent session is unavailable, Equalizer automatically selects a different server. Then, the client must establish a new session; Equalizer stuffs a new cookie in the next response.

IP-Address Based Persistence (Layer 4)

For Layer 4 TCP and UDP clusters, Equalizer supports IPaddress based persistent connections. With the *sticky connection* feature enabled, Equalizer identifies clients by their IP addresses when they connect to a cluster. Equalizer then routes requests received from a particular client during a specified period of time to the same server in the cluster.

A *sticky timer* measures the amount of time that has passed since there was a connection from a particular IP address to a specific cluster. The sticky time period begins to expire as soon as there are no longer any active connections between the client and the selected cluster. Equalizer resets the timer whenever a new connection occurs. If the client does not establish any new connections to the same cluster, the timer continues to run until the sticky time period

expires. At expiration, Equalizer handles any new connection from that client like any other incoming connection and routes it to an available server based on the current load balancing policy.

To correctly handle sticky connections from ISPs that use multiple proxy servers to direct user connections, Equalizer supports *sticky network aggregation*, which uses only the network portion of a client's IP address to maintain a persistent connection. Sticky network aggregation directs the user to the same server no matter which proxy he or she connects through.

You can also configure Equalizer to ensure that it directs requests from a particular client to the same server even if the incoming connection is to a different virtual cluster. When you enable *intercluster stickiness* for a cluster, Equalizer checks the cluster for a sticky record as it receives each connection request, just like it does for ordinary sticky connections. If Equalizer does not find a sticky record, Equalizer proceeds to check all of the other clusters that have the same IP address. If Equalizer still does not find a sticky record, it connects the user based on the current load balancing policy.

Is Connection Persistence Always Needed With Session Persistence?

Session persistence is a function of the application and the state created when a user logs into a web site. If the session persistence is maintained in the front end server, then Equalizer cookie persistence should be enabled. The client must maintain the connection to the same front end server in order for the login to remain valid. For example, Windows Terminal Services maintains a session directory "database" when a user logs into a session. If that state or database is in the front end server, or even in a back end server that only associates the client connection to that front end server, then the client must "persist" to the front end server to which it is originally connected.

In other configurations, the session "state" is kept in shared storage in a backend server or database that is accessible to all the front end servers. If this is the case, then connection persistence may not be needed; if the user is balanced among servers, then the session can still be maintained across the front end server group via access to the shared storage.

It's therefore important to understand how the load balanced application provides session persistence when managing persistent connections on Equalizer.

Layer 7 Load Balancing and Server Selection

Equalizer's support for Layer 7 content-sensitive load balancing enables administrators to define rules for routing HTTP and HTTPS requests, depending on the content of the request. Layer 7 load balancing routes requests based on information from the application layer. This provides access to the actual data payloads of the TCP/UDP packets exchanged between a client and server. For example, by examining the payloads, a program can base load-balancing decisions for HTTP requests on information in client request headers and methods, server response headers, and page data.

Equalizer's Layer 7 load balancing allows administrators to define rules in the administration interface for routing HTTP and HTTPS requests according to the request content. These rules are called *match rules*. A match rule might, for example, route requests based on whether the request is for a text file or a graphics file:

- ```load balance all requests for text files (html, etc.) across servers A and B```
- ```load balance all requests for graphics files across servers C, D, and E```
- ```load balance all other requests across all of the servers```

Match Rules are constructed using match functions that make decisions based on the following:

- HTTP protocol version; for HTTPS connections, the SSL protocol level the client uses to connect.

- Client IP address

- Request method (GET, POST, etc.)

- All elements of the request URI (host name, path, filename, query, etc.)

- Pattern matches against request headers

Match functions can be combined using logical constructs (AND, OR, NOT, etc.) to create extremely flexible cluster configurations. Please see "Using Match Rules" on page 159 for an overview of Match Rules, a complete list of match functions, and usage examples.

Geographic Load Balancing

The optional Envoy add-on supports , which enables requests to be automatically distributed across Equalizer sites in different physical locations. An Equalizer *site* is a cluster of servers under a single Equalizer's control. A is a collection of sites that provide a common service, such as Web sites. The various sites in a geographic cluster can be hundreds or even thousands of miles apart. For example, a geographic cluster might contain two sites, one in the eastern U.S. and one on the U.S.'s west coast (Figure 1).

Geographic load balancing can dramatically improve reliability by ensuring that your service remains available even if a site-wide failure occurs. Equalizer can also improve performance by routing requests to the location with the least network latency.
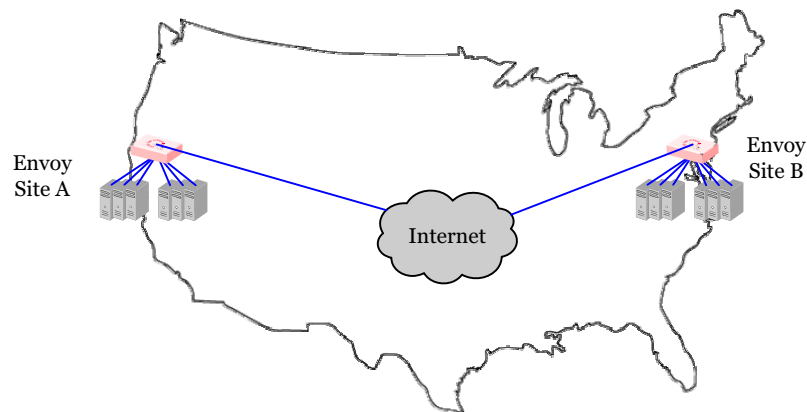
Figure 1   Geographic cluster with two sites

Geographic Load Balancing Routing

Envoy routes each incoming request to the site best able to handle it. If a site is unavailable or overloaded, Envoy routes requests to the other sites in the geographic cluster. When you enable geographic load balancing, Envoy directs incoming client requests to one of the sites in the geographic cluster based on the following criteria:

- **Availability: I**f a site is unavailable due to network outage, server failure, or any other reason, Equalizer stops directing requests to that site.

- **Performance:** Envoy tracks the load and performance at each site and uses this information to determine the site that can process the request most efficiently.

- **Distance:** Envoy notes the site that is *closest* to the client (in network terms) and offers the least network latency.

Distributing the Geographic Load

Envoy uses the Domain Name System (DNS) protocol[1] to perform its geographic load distribution. DNS translates fully-qualified domain names such as `www.coyotepoint.com` into the IP addresses that identify hosts on the Internet. For Envoy, the authoritative name server for the domain is configured to query the Equalizers in the geographic cluster to resolve the domain name. When Envoy receives a resolution request, it uses the load-balancing

---

1. For more information about DNS, see Paul Albitz and Cricket Liu, DNS and BIND, 3rd ed. (O'Reilly & Associates, 1998).

algorithms configured for the geographic cluster to determine the site that is best able to process the request and then returns the address of the selected site.

For example, the geographic cluster `www.coyotepoint.com` might have three sites (see Figure 2): one on the east coast of the U.S., one on the west coast of the U.S., and one in Europe. The servers at each site are connected to an Equalizer with the Envoy add-on installed.
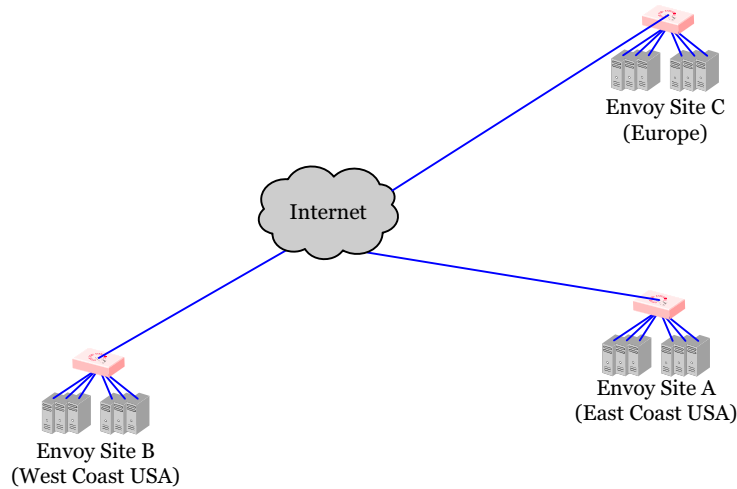
Figure 2   Three-site geographic cluster configuration

When a client in California attempts to connect to `coyotepoint.com`:

1.   The client queries its local DNS server to resolve the domain name (see Figure 3).
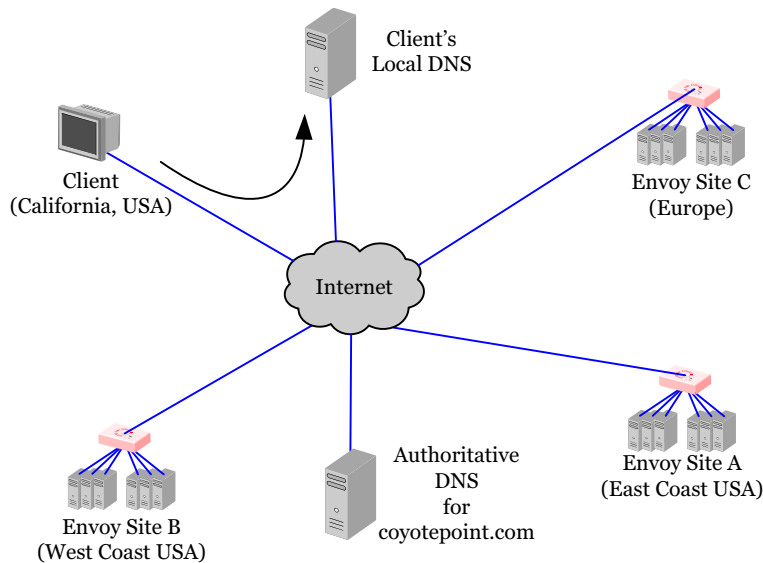
Figure 3   Client queries its local DNS for coyotepoint.com

2. The local DNS server queries the authoritative name server for `coyotepoint.com` (see Figure 4).
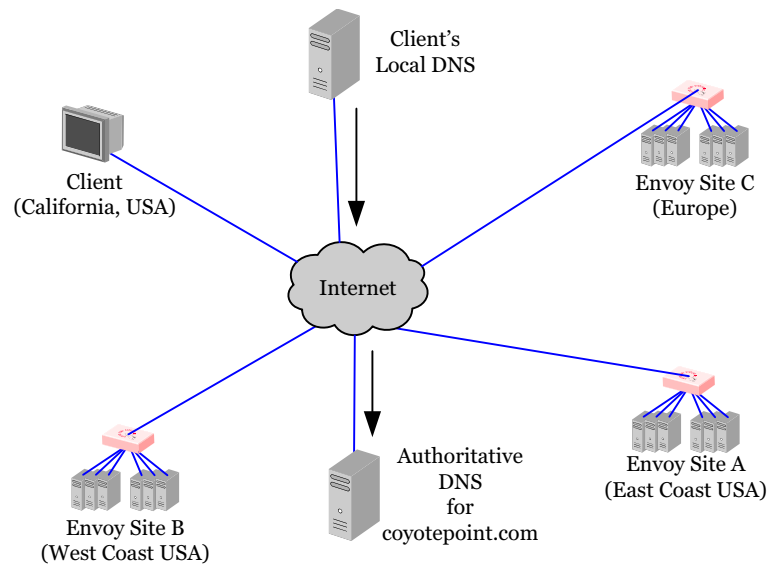
Figure 4   Client's local DNS queries the authoritative name server for coyotepoint.com

3. The authoritative name server provides a list of Envoy-enabled Equalizer sites and returns this list to the client's local DNS server (see Figure 5).
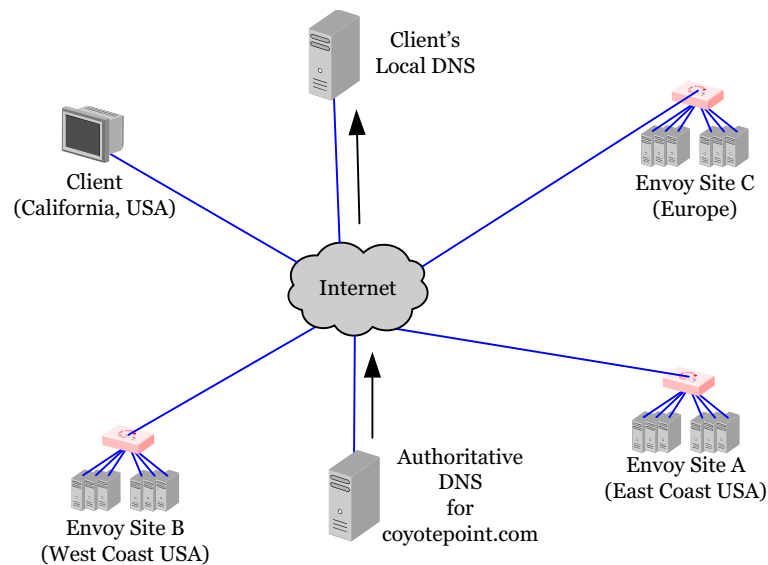
Figure 5   The authoritative name server for coyotepoint.com returns a list of delegates

4. The client's DNS server sends a request for the IP address of `coyotepoint.com` to each Envoy site in the list until one of them responds.

5. The Envoy site contacted returns the IP Address of the virtual cluster best able to handle the client's request. (For an overview of how Envoy chooses the virtual cluster IP to return to the client's DNS, see "Administering GeoClusters" on page 187.)

6. Finally, the client's local DNS server returns the virtual cluster IP to the client, which then sends the request to the virtual cluster.

Configuring the Equalizer Network

Equalizer is a versatile traffic management and application acceleration solution.

You can use Equalizer in a number of configurations -- use the information in this section to determine where it will fit into your network and how you will configure Equalizer's networking interfaces.

The following section provides some worksheets to help you plan your cluster configuration.

Equalizer's Network Interfaces and Ports

All Equalizer GX models have two network interfaces and a set of gigabit-speed ports on the front panel. Two port-based VLANs are configured with the following default port assignments:

- ports 1 and 2 (labelled in red on the front panel) are configured into a VLAN on one network interface (the *external* interface or VLAN)

- ports 3 and above are configured into another VLAN on the other network interface (the *internal* interface or VLAN).

Serial Port

External VLAN Ports (1 & 2)

Internal VLAN Ports (3 and above)
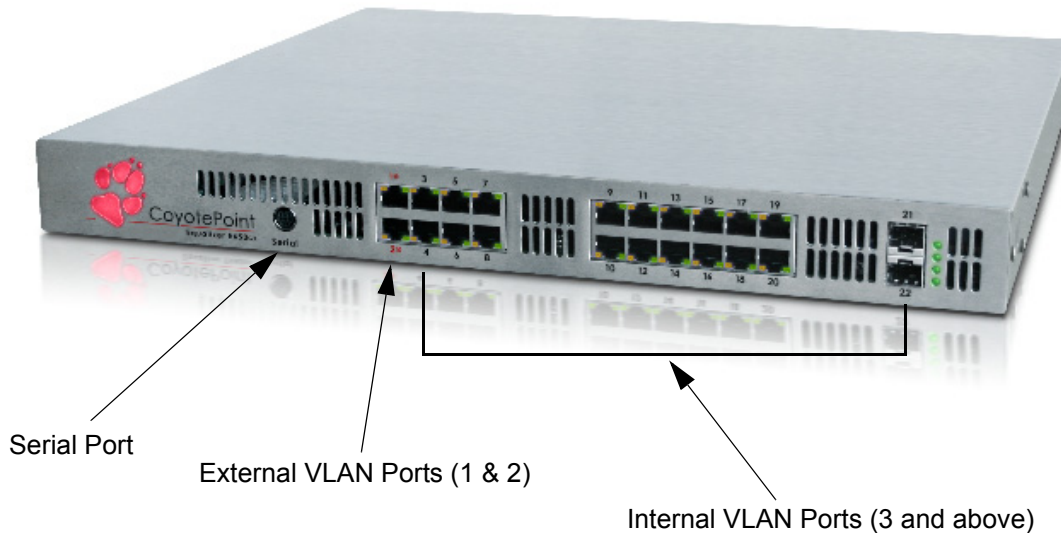
Figure 6   Equalizer's network ports on an E650GX

The default port configuration shown above can be updated as desired by reassigning individual ports to either of the configured VLANs using a graphical interface.

> **Note –** The E650GX, pictured above, has two fiber-optic ports labelled 21 and 22 at the far right of the front panel. These ports are both configured by default on the internal VLAN.

Equalizer's External VLAN

The external interface ports are usually connected to the network to which the client machines and possibly the Internet or an Intranet are connected. This *external network* receives the client request packets that Equalizer distributes across the available servers. Equalizer also uses the external network to transmit response packets to clients. This port is only used for dual network (external and internal) configurations. It is not used for single network configurations; see "Using Equalizer in a Single Network Environment" on page 12 for more information.

Hosts or routers on the external network can have routes to the internal network that are gatewayed through Equalizer's external address. Equalizer's external address is also used as an *administration address*, the IP address used to connect to Equalizer's browser-based administration interface.

Equalizer's Internal VLAN

Servers that process the incoming requests connect to the Internal VLAN ports, either directly or through a network device such as a switch, router, or other network device. In other words, the servers don't necessarily need to be connected directly to Equalizer, but must be *accessible* to Equalizer on the Internal VLAN's subnet. This can be accomplished either by having an IP address on the internal interface subnet, or through a network route defined either on Equalizer or on the gateway device for the configured subnet. An example of a configuration with both directly connected servers and remotely accessible servers is illustrated in the diagram below.
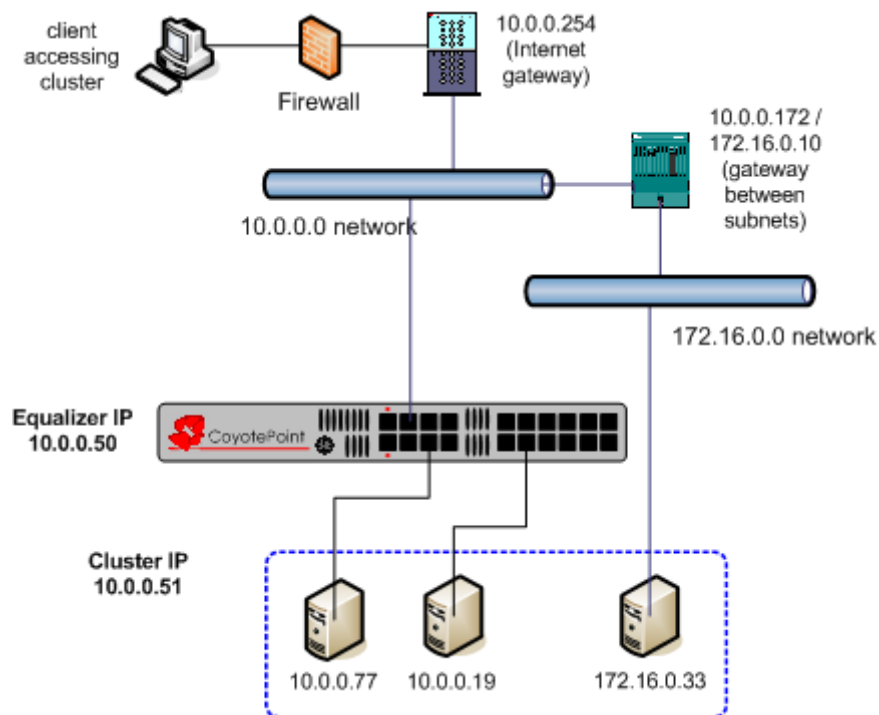
Figure 7   Example dual network configuration with local and remote servers

Using Equalizer in a Dual Network Environment

The example configuration in Figure 7 is an example of a *dual network* configuration, where Equalizer functions as the gateway between two separate networks -- the internal network where the servers reside and the external network on which clients and the Internet or an Intranet reside.

The example cluster shown above contains three servers, two on the local 10.0.0.0 subnet, and another on another subnet. In this example, a static route would be needed on Equalizer to forward all packets for the 172.16.0.0 network to the gateway at 10.0.0.172. Similarly, the server at 172.16.0.33 would need a static route that forwards all traffic for the 10.0.0.0 network through 172.16.0.10, the gateway address for the 10.0.0.0 network. As long as Equalizer and the server can communicate via the internal interface subnet, the server can be used in a cluster.

The servers available via the internal interface subnet provide services on specific IP addresses and ports and are organized into clusters. Equalizer's load-balancing subsystem translates client request packets to use a server IP as the destination IP, and then forwards the packets. When a server machine sends a response packet back to a client, in most cases it must send the response through Equalizer, which processes it and forwards it to the appropriate client

across the external network. In some configurations (such as Direct Server Return, or DSR), the server responds directly to the client without going through Equalizer.

When using Equalizer with spoofing, you must configure the routing tables on all servers so that Equalizer is the gateway for any outbound packets that leave the internal network. If the servers do not use Equalizer's internal address as the gateway when they send responses to clients, the reply packets will not be translated on their way to the client, causing the clients to reject the reply packets because they do not belong to an established connection. (From the client side, it would look like the server was not responding.) If you are using Equalizer without spoofing, you usually do not need to configure your servers to use Equalizer as a gateway.

Figure 8 shows a generalized picture of a dual network configuration:
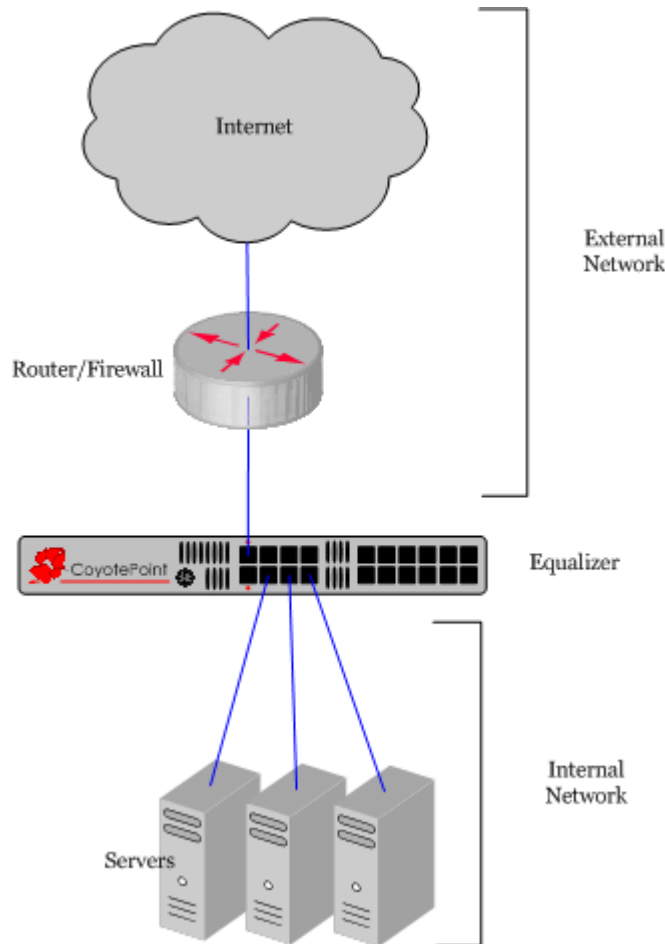
Figure 8   example dual network configuration

In some cases, implementing dual network mode might require reorganizing the current network configuration into two subnet. In such cases, a single network configuration as described in the following section may be more appropriate.

Using Equalizer in a Single Network Environment

If you do not want to split your network into internal and external networks, you can configure Equalizer to use a *single network* configuration, effectively placing both the clients and servers on the same network. When using Equalizer in single network mode, the client machines, servers, Intranet and/or Internet must all connect to Equalizer through one of the server ports.

Figure 9 on page 13 shows this configuration in detail. Certain protocols that use dynamic port mapping or multiple TCP/UDP ports work best in a single network environment. For example, use a single network configuration if you need your servers on your internal network to communicate with a Windows file server or a machine running pcAnywhere™.

In a single network configuration, you use only one of Equalizer's network interfaces, the internal interface. This means that you do not use the default external VLAN server ports (1 and 2) on the front panel. You can, if you want, reconfigure ports 1 and 2 on the Internal VLAN using the Switch management Interface ().

Connect one of Equalizer's server ports to the network and connect the servers to the other server ports as usual, as shown in the diagram below.
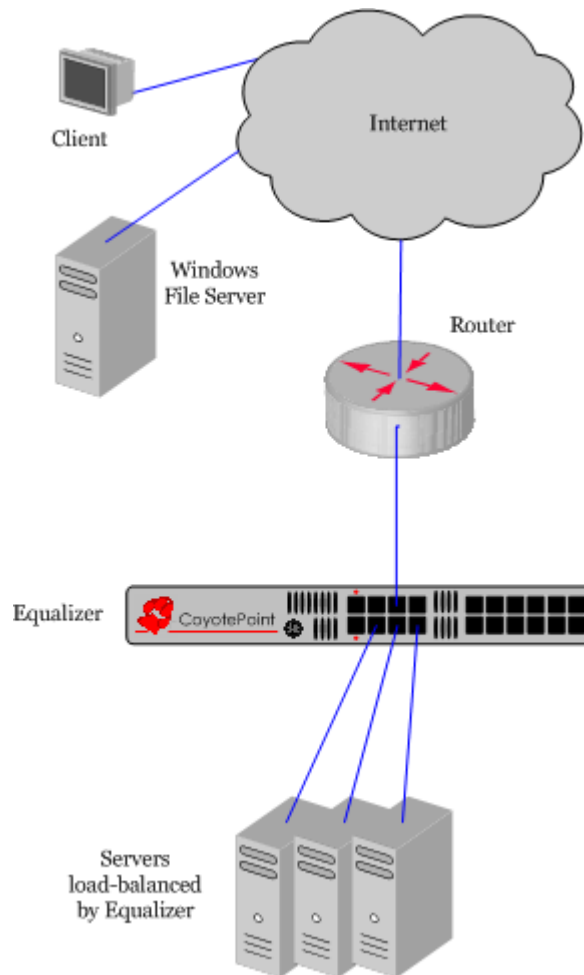
Figure 9   Sample single network configuration for a switch-based Equalizer

You must configure servers, which must have valid network addresses on the external network, to use Equalizer's internal address as the gateway for outbound packets (unless you are using Direct Server Return or DSR).

Most operating systems allow you to specify a host route (gateway) for packets destined for specific hosts. If you want your virtual clusters to accept connections from clients on the same network as the servers, you must configure the servers to route packets destined for these clients through Equalizer. The clients on the local network must also be configured to use the Equalizer as their gateway; clients that do not have such routes configured connect to the server's IP address directly and not through a virtual cluster (that is, they are not routed through Equalizer).

Using a Second Equalizer as a Backup Unit

You can configure a second Equalizer as a backup unit that will take over in case of failure. This is known as a *failover* or *hot-backup* configuration. The two Equalizers are defined as *peers*, the *primary* unit and the *backup* unit. If the primary Equalizer stops functioning, the backup unit adopts the primary unit's IP addresses (clusters) and begins servicing connections. In a failover configuration, the servers in a virtual cluster use a separate *failover IP alias* as their default gateway, rather than the IP address of the cluster or external port on a particular Equalizer. The failover alias migrates between the primary and backup unit as needed, automatically ensuring that the servers have a valid gateway in the event of a failure.

In a failover configuration, both the primary and backup Equalizers are connected to the same networks; the backup unit's cluster and external ports must be connected to the same hubs or switches to which the primary Equalizer's ports are connected. Figure 10 on page 14 shows a sample failover configuration.
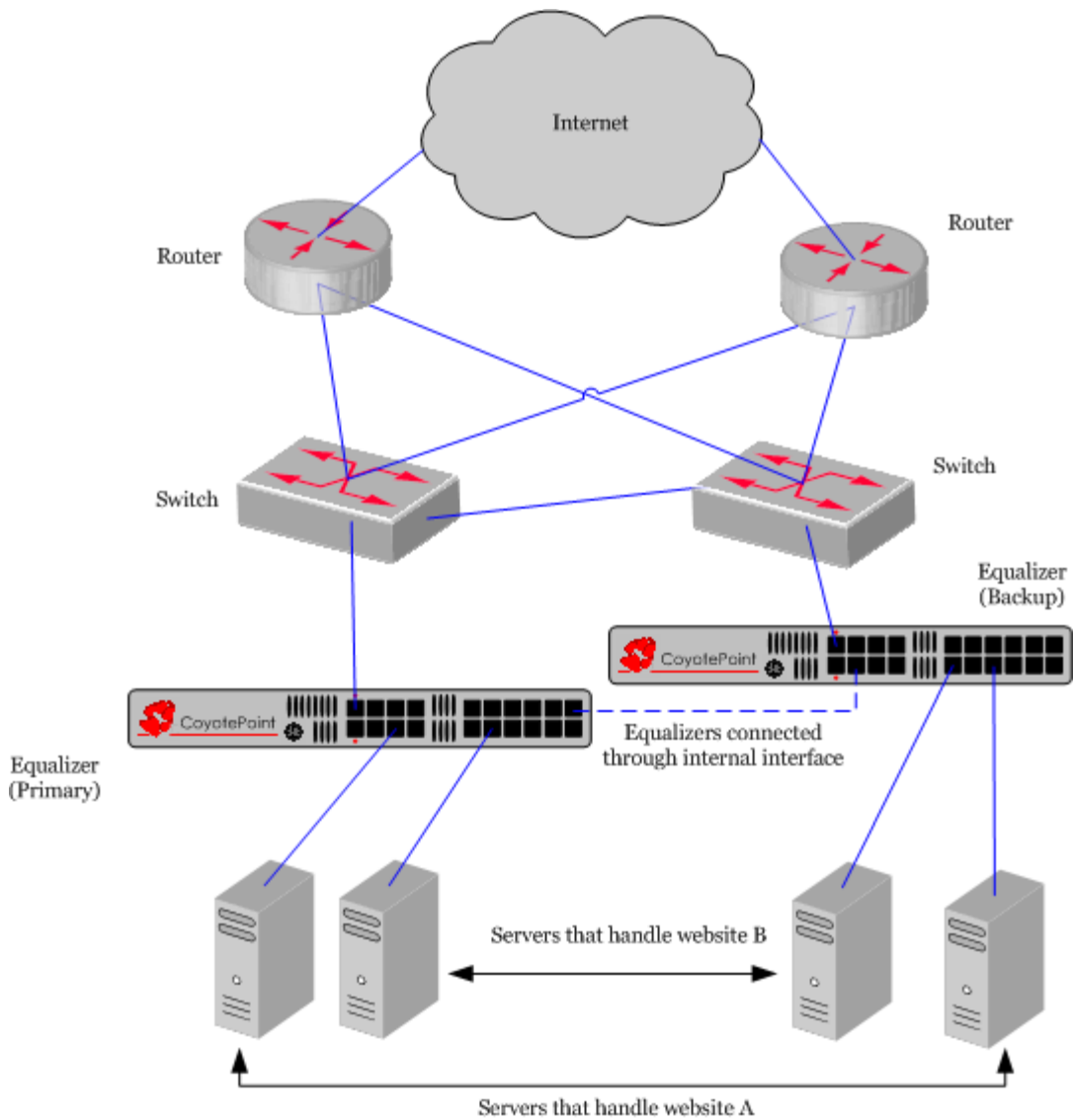
Figure 10   Sample failover configuration

In the sample failover configuration above, there is no single point of failure. If a router goes down, the other router takes over; if a link fails, requests are routed through another link. In this dual network configuration, the Equalizers

Equalizer Installation and Administration Guide

communicate over both the internal and external subnets. The connection shown between the Equalizers on the internal interface might not be necessary in a single network configuration, or if the Equalizers can communicate over the internal interface through an existing route.

The backup Equalizer monitors all traffic to and from the primary unit; both Equalizers periodically exchange status messages over the local area network. The Equalizers also exchange current configuration information. When you update the configuration on either machine, the configuration on its peer is automatically updated.

Should either Equalizer fail to respond to a status message probe, the other Equalizer begins a diagnostic cycle and attempts to contact its peer via the other network ports. If these attempts fail, the peer is considered to be *down*.

When the backup Equalizer determines that its failover peer is down, it initiates a failover process:

1. The backup Equalizer configures the virtual cluster aliases on the external port and sends out "gratuitous ARP" packets that instruct any external-network routers to replace ARP table entries that point to the physical address of the failed Equalizer with the physical address of the backup unit.

2. The backup Equalizer configures a *failover gateway alias* on the port that is local to the servers.
   - With no backup configuration, the servers use the IP address of the cluster or external port as their default gateway.
   - In a hot-backup environment, the gateway address can migrate between the primary and backup unit. This requires an additional address.

3. The Equalizer kernel changes from BACKUP mode to PRIMARY mode. The PRIMARY-mode Equalizer performs gateway routing of packets between its cluster and external ports, address translation, and load balancing.

When a failed unit is brought back online, it begins to exchange status messages with its failover peer. Once both Equalizers have synchronized, the newly-started unit assumes the backup role.

Using Reserved IP Addresses

RFC 1597 defines blocks of internet IP addresses that will never be officially assigned to any entity, and will not be routed through the Internet. This means that any site can use them in a local Intranet. These reserved or private IP addresses are:

```
10.0.0.0 to 10.255.255.255
172.16.0.0 to 172.31.255.255
192.168.0.0 to 192.168.255.255
```

In environments where conserving IP addresses is important, using reserved IP addresses can minimize the number of routeable IP addresses needed. Equalizer supports placing servers and clients on these reserved networks.

For example, an ISP hosting several hundred unique web sites replicated on three servers might not want to assign real IP addresses for all of them because each virtual cluster would consume four addresses: three on the back-end servers and one for the virtual cluster. In this case, the ISP might use 10.0.0.0 (the now-defunct Arpanet) as the internal network and assign virtual server addresses out of this network for the servers.

Figure 11 shows an example of a reserved network configuration.
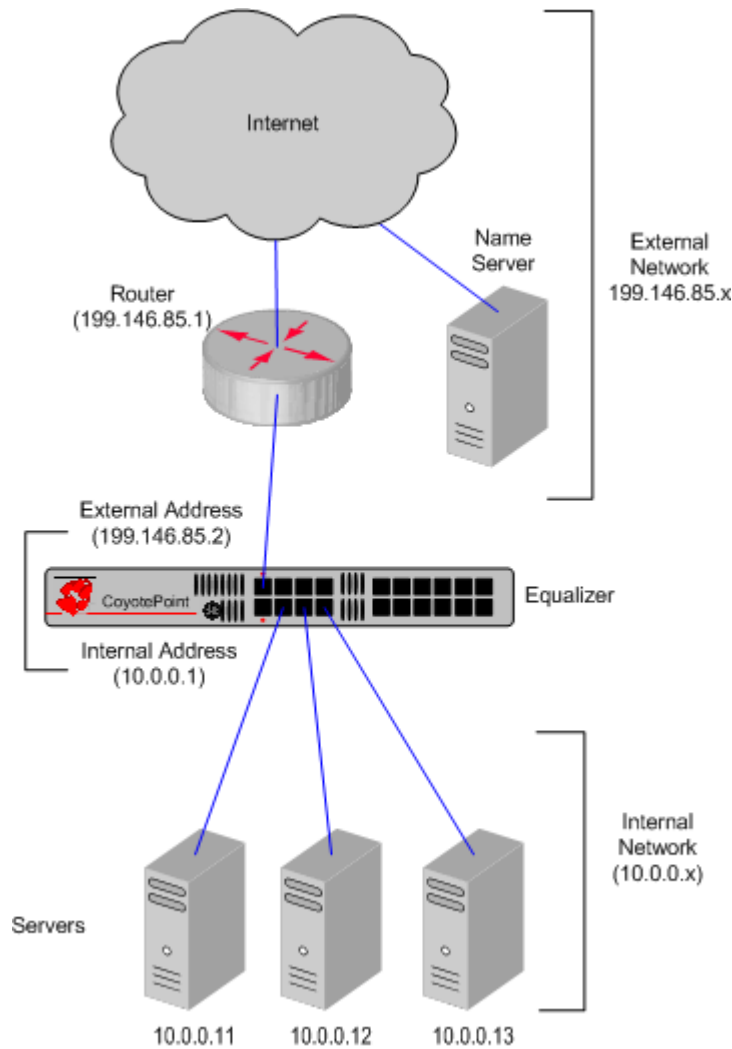
Figure 11 Reserved internal network configuration

If servers placed on a non-routable network need to communicate with hosts on the Internet for any reason (such as performing DNS resolution or sending e-mail), you need to configure Equalizer to perform *outbound NAT*. When you enable outbound NAT, Equalizer translates connections originating from the servers on the reserved network so that external hosts will not see packets originating from non-routable addresses; specifically, it substitues the Equalizer's external interface IP address for the server address in the server reponse. If you use a failover configuration, you must enable outbound NAT on both Equalizers. For more information, see "Setting Up a Failover Configuration" on page 53.

Note – Because the external interface address is used when outbound NAT is enabled, you should *disable* outbound NAT if your system is in single network mode (which only uses the internal interface).

Sample Configuration Worksheets

Network Configuration

Before you begin to install and configure Equalizer, make a detailed diagram of your network configuration. Include in it all the IP addresses that you'll need to configure Equalizer. The following figure shows an example of such a worksheet:
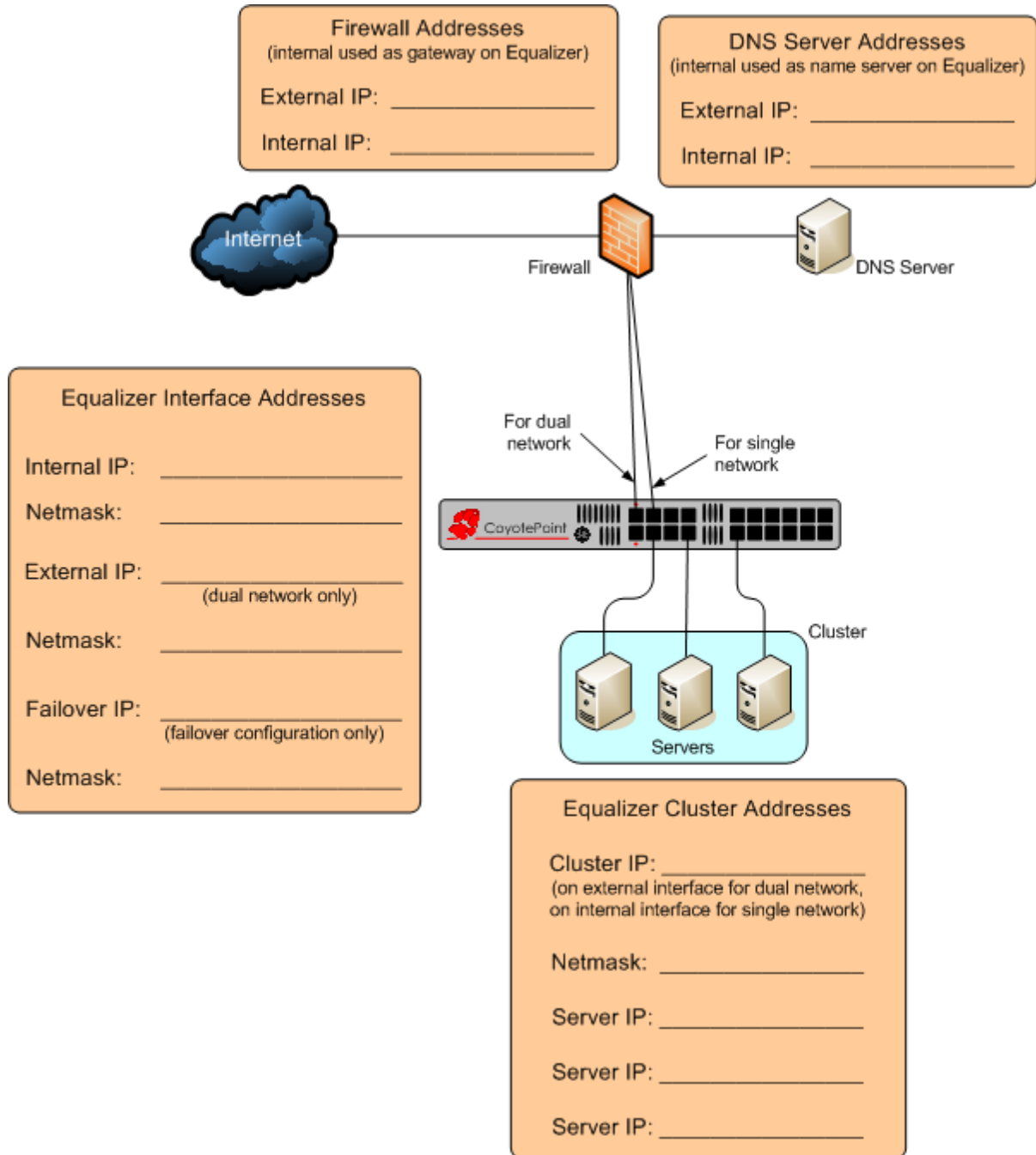
Figure 12 Network configuration worksheet

Cluster Configuration

When you start adding clusters to Equalizer, you'll need to configure parameters and options for each cluster that you create, and these depend on the cluster type chosen. The following table shows the most important options to set for each cluster type when you first create the cluster. Additional cluster settings are available to fine tune the performance of your cluster after you get it working.

L4 TCP / L4 UDP Clusters	HTTP / HTTPS Clusters
Name: _____	Name: _____
IP Address: _____	IP Address: _____
Port or Port Range: _____ The port or range of ports for incoming client connections to the cluster IP.	Port: _____ The port for incoming client connections to the cluster IP.
Sticky Time: _____ Time that client connections to this cluster will continue to be routed to the same server (default: 0; connections do not persist). A "sticky record" is created for each connection.	Persist (check box): _____ Insert a cookie into the client request header so that it gets redirected back to the same server for the age of the cookie (default: on). See "Enabling Persistent Server Connections" on page 88.
Inter-cluster Sticky (check box): _____ If you have two clusters with the same IP address, different ports, and the same servers defined, enabling inter-cluster sticky routes traffic from one port to another on the same server when a client that has a sticky record (see above) to one server port attempts to reconnect and that port cannot be reached. Equalizer will attempt to connect the client to the same server on the other port. (default: off).	Once Only (check box): _____ Examine only the first client request header in a connection and load balance subsequent packets in the same way (default: on). Clusters with Persist enabled and/or non-default Match Rules should have Once Only disabled. See "Enabling the Once Only and Persist Options" on page 90 for more information.
Spoof (check box): _____ When Equalizer forwards a client request packet to a server, use the client IP address in the packet, so that the server sees the client IP as the packet source IP (default: on). When disabled, Equalizer's internal IP address is the source IP in the packets sent to the servers.	Spoof (check box): _____ When Equalizer forwards a client request packet to a server, use the client IP address in the packet, so that the server sees the client IP as the packet source IP (default: on). When disabled, Equalizer's internal IP address is the source IP in the packets sent to the servers.
Outbound NAT (check box): _____ This is a global parameter that affects all cluster connections. When enabled, this option translates the source IP in outbound packets from the server IP to the Equalizer's external interface IP (default: off).	

Figure 13 Cluster settings worksheet

Equalizer Installation and Administration Guide

Server Configuration

The initial server parameters that you should configure for each load balanced server in a cluster are shown below. If this server/port combination is in more than one cluster, include all clusters on the **Cluster Name** line. [Note that the same IP/port combination cannot be used in more than one Layer 4 UDP cluster.] Additional server settings are available that allow you to modify the default behavior.

Server Name: _____
Cluster Name: _____
IP Address: _____
Port: _____ 0 means use the port or start port defined for the cluster. Specify another port to map the cluster port or start port to the specified server port.
Weight: _____ A number from 20 to 200 that indicates the capacity of the server relative to the other servers in the cluster. For example: specifying weights of 100, 100, and 200 for three servers means that you expect the third server to be able to handle about twice as much traffic as the other two servers.

Figure 14   Server settings worksheet

For a complete description of cluster and server settings, please see Chapter 5, "Administering Virtual Clusters." on page 73.

Chapter 2:

Installing and Configuring Equalizer Hardware

This chapter contains all the information you need to get your Equalizer out of the box and onto your network:

Before You Turn Equalizer On for the First Time

The first step in setting up Equalizer is to connect it to the local area network and a power source. Once you have installed Equalizer, you need to configure it as described in Chapter 3, "Configuring Equalizer Hardware".

Please review the warnings located in Appendix I , *Additional Requirements and Specifications*, on page 263 for precautions you must take before installing your Equalizer hardware.

Stepping Through the Hardware Installation

To install Equalizer, follow these steps:

1. Carefully remove the Equalizer rack-mount enclosure and cables from the shipping container.

    Save the original packaging in case you need to ship the Equalizer for any reason, such as sending it in for warranty service. The Equalizer chassis does not contain any parts that you can service. If you open the chassis or attempt to make repairs, you may void your warranty. See Appendix H , *License and Warranty*, on page 259.

2. Place the Equalizer in its intended position in an EIA equipment rack or on a flat surface. Please see Appendix I , *Additional Requirements and Specifications*, on page 263, for a list of environmental limits and power requirements for your Equalizer.

3. Connect a console to Equalizer. Do one of the following:

    a. Connect a serial terminal or a workstation running terminal emulator software to the serial port on the front panel of the Equalizer (see Figure 6 on page 10). Use the serial cable supplied with Equalizer.

    b. Some Equalizer models also have a USB keyboard connector and VGA display adapter at the back of the unit. You can connect a USB keyboard and VGA display to these ports as a console, instead of the serial port. Use the cables supplied with the keyboard and display that you choose.

4. Connect Equalizer to the network with a quality category 5 network cable:

    a. To use Equalizer as an intermediary between an external and internal network, connect Equalizer to the external network using one of the RJ-45 ports labelled 1 or 2 on the front panel. Connect Equalizer to the internal network using one or more of the ports numbered 3 and above.

    b. For a single-network (one subnet) topology, connect Equalizer to the network and the servers using one of the numbered RJ-45 ports numbered 3 and above on the front panel of the Equalizer.

5. Connect Equalizer to an appropriate power source using the supplied power cord, which plugs into the 3-pin connector on the rear of the Equalizer enclosure. This system uses an auto-sensing power supply that can operate at 50Hz or 60Hz, 110-240 VAC input.

6. Turn on the power using the switch on the rear panel.

Setting Up a Terminal or Terminal Emulator

After the Equalizer hardware, you need to directly connect a terminal to Equalizer to complete the hardware configuration.

Serial Connection

When you set up Equalizer for the first time, you must use a serial connection in order to configure Equalizer's network with the **eqadmin** interface. Connect the serial port on the Equalizer (see Figure 6) to the serial port on a terminal, or any system (such as a Windows or Unix PC) running terminal emulation software.

Configure your terminal or terminal emulator software to use the following settings:

- 9600 baud
- 8 data bits
- no parity
- one stop bit
- VT100 terminal emulation
- ignore hang-ups (if supported); this allows a single terminal session to continue running even if Equalizer restarts

On Windows systems, you can use the Windows built-in terminal emulator, **HyperTerminal**, or the **Tera Term Pro** terminal emulator to log in to Equalizer over the serial port. On Unix systems, you can use the **cu**(1) command or any other Unix serial communication program.

If you use **HyperTerminal**, in addition to the settings shown above, select **File > Properties > Settings** from HyperTerminal's menu, select **VT100** in the **Emulation** drop-down box, and then **Terminal Setup** to enable these options:

- keyboard application mode
- cursor keypad mode

Tera Term Pro version 2.3 is freely available at:

```
http://hp.vector.co.jp/authors/VA002416/teraterm.html
```

Performing Basic Equalizer Configuration

The first time you configure Equalizer, you'll need to use the Equalizer Configuration Utility (**eqadmin**) to specify at least the following:

- **Network Interfaces**: Equalizer's external and internal network interfaces and the netmasks associated with these networks. You must at least configure one of the network interfaces with an IP address in order to access the Equalizer browser based Administration Interface (described in the next chapter).
- **Hostname/IP Address**: The DNS hostname or IP address that is assigned to Equalizer.
- **Default Gateway**: The IP address of the router or other network device that Equalizer will use to forward packets to the Internet or Intranet.
- **DNS Server**: The Domain Name Server Equalizer will use.

Starting to Configure Equalizer

As Equalizer boots, the terminal displays a series of device probe and startup messages. Normally, you can ignore these diagnostic messages. However, if you do not configure the terminal emulation software to ignore hang-ups, the terminal session might exit twice during the boot process. If this happens, restart the terminal session.

To begin configuration, follow these steps:

1. When the boot process is complete, press **Enter** on the terminal keyboard to display the login prompt.

2. When the login prompt appears, type **eqadmin** and press **Enter**.

3. When the password prompt appears, enter the **eqadmin** password and press **Enter**. Equalizer automatically launches the **Equalizer Configuration Utility**, which provides a character-based interface for setting and changing Equalizer configuration parameters.

4. If the terminal display is not readable or not formatted properly, press **Esc** and make sure that your terminal emulator is set for VT100 emulation. Start over at Step 2.

5. To select a menu item within the configuration utility, press one or more arrow keys until you highlight the desired item. If the arrow keys do not operate within your terminal emulator, you can use **Ctrl-n** to select the *next* menu item or **Ctrl-p** to select the *previous* menu item. Press the **Tab** key to highlight one of the menu actions (such as Select or Cancel) displayed at the bottom of the window. Then press **Enter** to continue.

   Continue with "Configuring Equalizer's Network Interfaces" on page 24.

Configuring Equalizer's Network Interfaces

To configure the Hostname, Network Interfaces, Default Router, and DNS, use the following steps. Even if you are using your Equalizer in a single network configuration, you need to enter information for both the external and internal (server) interfaces.

6. In the Equalizer Configuration Menu window, select option 1, **Interfaces**, and press **Enter**. Equalizer displays the **Configure network interfaces** window:

7. Press one or more arrow keys until you highlight **External Ethernet interface**; then press **Enter**. The Equalizer Configuration Utility displays the Network Configuration window.

   a. In the **Host** field (required), enter the name for the Equalizer on your network. This can be the system node name (such as "eq-ext"), or the fully qualified domain name (FQDN, such as "eq-ext.customer.com"). If you supply the FQDN in the **Host** field, the **Domain** field will automatically be filled in using the domain of the FQDN.

   b. In the **Domain** field (required), enter the domain name for the Equalizer. (For example, for the fully qualified domain name, `eq-ext.customer.com`, you would enter "customer.com" in the **Domain** field.

   c. In the **Gateway** field (required), enter the IP address of the router on the external network. This router is the gateway for all the packets Equalizer sends to the outside world through the external network. For example, if your external network router is located at IP address `192.22.33.1`, enter "**192.22.33.1**" in the Gateway field.

   d. In the **Name Server** field, enter the IP address of the domain name server that Equalizer will use. To indicate that no name server is available, you can enter "**NONE**" or "**0.0.0.0**"; we recommend you use "**0.0.0.0**", which will disable DNS lookups on Equalizer.

      Only one DNS server can be specified via eqadmin. If you want to specify additional alternate DNS servers, see the section .

   e. If you will be using a **dual-network configuration**, you need to assign an IP address to the external interface. In the **IP address** and **Netmask** fields, respectively, specify the IP address and netmask for the external interface. For **single network configurations** using a switch-based Equalizer, leave the IP address for the external interface blank (or, set to NONE) to disable the port.

     f.   When you're finished, highlight **OK**. Then press **Enter**.

8.   To specify the internal interface parameters, select **Internal Ethernet interface**. Then press **Enter**.

     a.   Specify the **IP Address** and **Netmask**. For example, if the internal interface will have the address `192.22.34.2`, enter `192.22.34.2` in the **IP Address** field. Leave the **IP address** field blank or type `NONE` to disable the server ports. The **Netmask** used will depend on how your network is configured.

     b.   Highlight **OK. Then** press **Enter**.

9.   Highlight **Back**. Then press **Enter** to return to the main configuration menu.

10. In the **Equalizer Configuration Menu** window, select option 6, **Commit**; then press **Enter**. The system commits your changes and automatically reboots.

After rebooting, you can continue to configure Equalizer as described in the remainder of this chapter, and test your initial setup as shown in the section "Testing Your Basic Configuration" on page 30.

Setting the Time Zone

The time zone can be set using the browser-based Administration Interface, which also supports setting up a Network Time protocol (NTP) server, as shown in "Managing System Time and NTP" on page 61. To set the system time zone using **eqadmin**, follow these steps:

1.   In the **Equalizer Configuration Menu** window, select option 2, **Time Zone**, and press **Enter**.

2.   Use the menus to specify your time zone.

3.   Highlight **OK; then** press **Enter**.

Setting the Date and Time

The current date and time can be set using the browser-based Administration Interface, which also supports setting up a Network Time protocol (NTP) server, as shown in "Managing System Time and NTP" on page 61. To set the system date and time using **eqadmin**, follow these steps:

1.   In the **Equalizer Configuration Menu** window, select option 3, **Time**; then press **Enter**.

2.   Specify the current date and time, based on a 24-hour clock, in the format MM/DD/YY HH:MM.

3.   Highlight **OK; then** press **Enter**.

Adding Administrative Interface Logins

The browser-based Administrative Interface by default supports two logins: **touch** and **look**. The **touch** login has control access over Equalizer's configuration, while the **look** login has read access only to the interface. Additional logins can be created with custom permissions on clusters and global configuration. See "Managing Multiple Interface Users" on page 36 for a description of the user management interface.

Option **4 Manage users** allows you to create a full access or read only user login for the Administrative Interface in the event you cannot log in, either because all logins have been accidentally deleted or all administrative passwords lost. To add a user login, do the following:

1.   In the **Equalizer Configuration Menu** window, select option 4, **Manage users**, and press **Enter**.

2.   Select either **Full access login** or **Read-only login**.

3.   Type in a name for the new login. Then, type the password. The password can include any combination of printable characters (except spaces) and can be no more than 20 characters in length (*note that spaces are accepted by the interface, but will not work when attempting to log in*).

4. Select **OK** to create the login and return to the main menu.

Changing Equalizer's Console Password

The console password is the password for the **eqadmin** account, which automatically displays the Equalizer Configuration Utility when you log in via **ssh** or the serial port. The factory-installed password for this account is **equalizer**. To change Equalizer's console password, follow these steps:

1. In the **Equalizer Configuration Menu** window, select option 5, **Console**. Then press **Enter**.

2. Type the new password; it can include any combination of printable characters (except spaces) up to 20 characters.

3. When prompted, enter the password again to confirm the change. The new password takes effect immediately.

Upgrading Equalizer Software

After you have finished setting up your Equalizer to access the Internet, you can use the Equalizer Configuration Utility to install the latest Equalizer software upgrade from Coyote Point.

> **Note –** Before you can upgrade your Equalizer, you must first license it See "Licensing Equalizer" on page 44 for more information.

The procedure below contains the basic upgrade instructions for the current Equalizer software release. Please visit the **Support Portal** at **support.coyotepoint.com** for detailed upgrade instructions, release notes, and version compatibility charts for all releases.

1. In the **Equalizer Configuration Menu** window, select option 8, **Upgrade**, and press **Enter**. You are prompted to choose the location of the upgrade image:
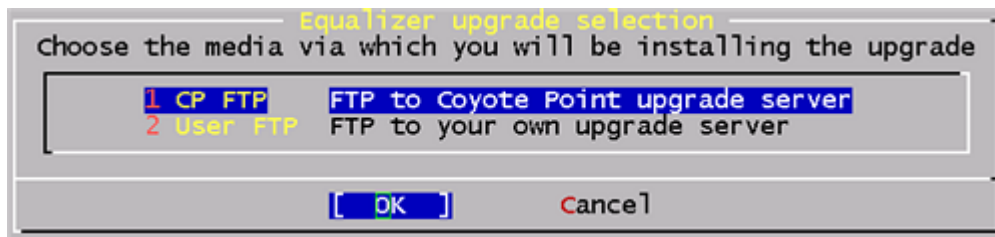
- Use the first option to connect to the Coyote Point FTP server to download the upgrade image.

- Use the second opton to specify a local FTP server, to which you have already downloaded the upgrade image from the Coyote Point FTP server.

Use the arrow or number keys to choose the appropriate location and then press **Enter**.

2. The upgrade utility prompts you to enter the upgrade URL:
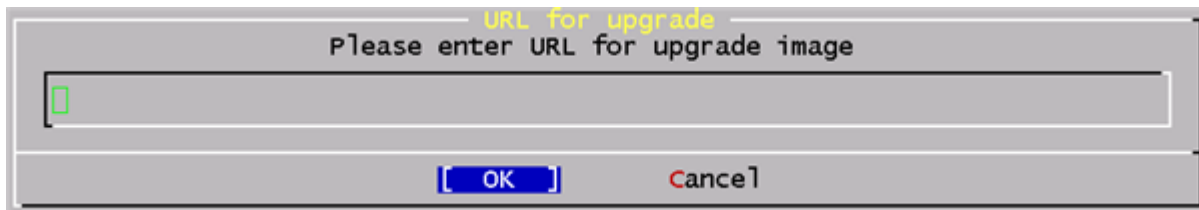
Enter the URL appropriate for the option you selected in **Step 1**:

- If you chose **Option 1 CP FTP**: Enter the upgrade image URL provided to you by Coyote Point. The latest release of Equalizer software is always located at the following URL:

  ```
  ftp://ftp.coyotepoint.com/pub/patches/upgrades/latest/upgrade.tgz
  ```

- If you chose **Option 2 User FTP**: Enter the upgrade image URL appropriate for your local FTP server, as provided by your local network administrator.

After entering the URL, select **OK**, and press **Enter**. Equalizer downloads the upgrade file and runs the upgrade script.

3. When prompted, confirm that you want to upgrade the Equalizer software. The script then installs the software upgrade. Upgrades may take as long as five minutes. After the upgrade is installed, you will be prompted to reboot the system.

Shutting Down Equalizer

You can shut down Equalizer from the configuration utility. *Note that shutting down Equalizer does not automatically commit changes made to the configuration.* To shut down, follow these steps:

1. In the **Equalizer Configuration Menu** window, select option 7, **Shutdown**; then press **Enter**.

2. After the shutdown process completes, power off the system.

Adding Alternate DNS Servers

Only one DNS server can be specified via **eqadmin**, as shown in the section "Configuring Equalizer's Network Interfaces" on page 24. If you want to add additional, alternate DNS servers, follow the procedure below.

> **Caution –** Do not attempt this procedure unless you are comfortable working at the Equalizer command line and editing system files using standard BSD system text editors. Or, contact **Coyote Point Support** for assistance.

1. Log in to Equalizer via **ssh** (as *eqsupport*) or the serial interface (as *root*). If you use **ssh**, enter the following command to switch to the root login:

   ```
   su root
   ```

2. Enter the following two commands:

   ```
   mount -w /
   cd /var/etc
   ```

3. Edit the file *resolv.conf* using either of the text editors supplied with Equalizer: **ee** or **vi**. For example:

   ```
   ee resolv.conf
   ```

   Up to three name servers are permitted in the file, as in the following example:

   ```
   domain          mydomain.com
   nameserver      10.0.0.120
   nameserver      10.0.0.122
   nameserver      10.0.0.124
   ```

   For instructions on using the **ee** and **vi** editors, see their manual pages on the FreeBSD Hypertext Manual Pages website: **www.freebsd.org/cgi/man.cgi**.

4. When you are done updating the file, save your changes and exit the editor.

5. Enter the following two commands:

   ```
   shadow /var/etc/resolv.conf
   mount -r /
   ```

The new DNS settings take effect for all subsequent DNS queries, and will persist across system reboots.

Managing Remote Access to the Equalizer

Remote access, when enabled, provides a user account (**eqsupport**) which allows you to log into Equalizer over a Secure Shell (SSH) connection.

> **Note –** By default, the password for the **eqsupport** account is blank. If you enable the account, change the password when you enable it.

Managing the Remote Access Account

To enable, disable, or change the password for the **eqsupport** account, do the following:

1. Log into the Equalizer hardware configuration utility using a terminal or terminal emulator (see "Setting Up a Terminal or Terminal Emulator" on page 23 and "Starting to Configure Equalizer" on page 24.

2. In the **Equalizer Configuration Menu**, select option 9, **Manage 'eqsupport'**, and press **Enter**. Equalizer displays the **Equalizer CLI eqsupport account selection** window.

```
┌──────────── Equalizer CLI eqsupport account selection ────────────┐
 Select whether you would the 'eqsupport' account enabled or disabled.
 Changes are applied immediately. Account starts WITHOUT a password.

 ┌──────────────────────────────────────────────────────────────┐
 │  1 Enable      Enable 'eqsupport' account                     │
 │  2 Disable     Disable 'eqsupport' account                    │
 │  3 Password    Set 'eqsupport' password                       │
 └──────────────────────────────────────────────────────────────┘

            [  OK  ]          Cancel
```
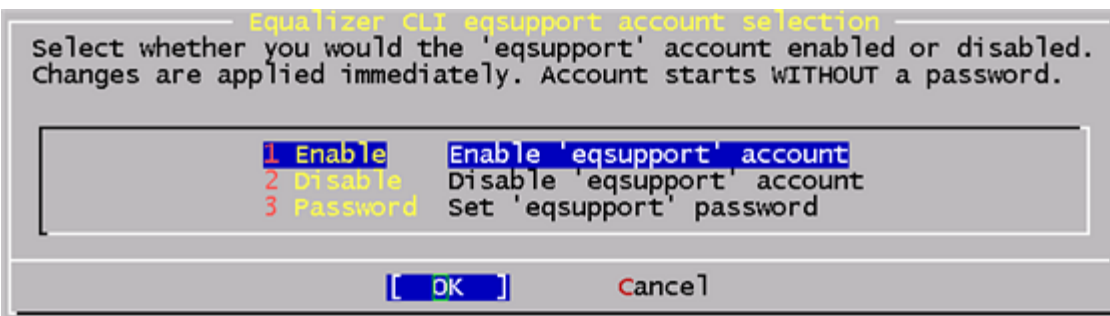
3. The following selections are available:

   a. To enable the remote access account, use the arrow keys to highlight **Enable** and press **Enter**. The account is now enabled.

   b. To disable the remote access account, use the arrow keys to highlight **Disable** and press **Enter**. The account is now disabled.

   c. To change the password, use the arrow keys to highlight **Password** and press **Enter**. Follow the prompts to change the password.

   If you modify the password for the account when it is disabled, Equalizer will display a reminder that the account must be enabled before you can use it.

4. When you are done, highlight OK on the account selection window and press **Enter** to return to the **Equalizer Configuration Menu**.

Using the Remote Access Account

Use the Secure Shell Client (SSH) to log in with the remote access account user name (**eqsupport**) and password, using Equalizer's external or internal interface IP address. The account is not enabled by default, and must first be enabled (see the previous section) in order for this to work. For the best visual output when using **eqadmin** over **ssh**, the following are recommended:

- The PuTTY terminal emulator, freely available from

```
http://www.chiark.greenend.org.uk/~sgtatham/putty/
```

- An SSH client running from a Windows Command window; for example, OpenSSH, which is freely available from:

```
http://sshwindows.sourceforge.net/download/
```

- An SSH client running from a Cygwin window. Cygwin is a UNIX shell environment that includes versions of many UNIX utilities, including SSH; it is freely available from:

```
http://cygwin.com/
```

When you run the Setup program to install, make sure that SSH (under "Net"), and the Xorg Server and xterm (under "X11") are selected for installation. To run, open a Cygwin window and enter 'startx'; once the Xterm window opens, enter 'ssh eqsupport@*equalizer-ip*'.

Configuring a Second Equalizer As a Backup (Failover)

You can configure a second Equalizer as a hot backup (or hot spare) so that if the Equalizer that currently handles requests (the *primary unit*) fails, the *backup unit* automatically takes over. This is called a *failover configuration*.

Both Equalizers are configured to default to either the primary or backup role. When a failed unit comes back online, it assumes the backup role, even if it is designated the default primary.

If you are going to use two Equalizers in a failover configuration, perform the basic configuration for both units now as described in the previous section.

Additional configuration for failover is performed through the Equalizer Administration Interface, as described in the section "Setting Up a Failover Configuration" on page 53.

Configuring Routing on Servers

In most configurations, you should configure your servers so that Equalizer gateways the packets the servers send to clients; this means that each server should have an Equalizer IP address as its default gateway in its routing tables. If you do not adjust the routing on your servers, a client may not receive a response when it attempts to contact a virtual cluster, and the connection will time out.

One exception is Direct Server Return (DSR); in DSR configurations, client requests coming through Equalizer are routed to servers, which then respond directly back to the clients without going through Equalizer. Servers in DSR configuration typically have a default gateway other than Equalizer.

When you configure servers, the *default* route gateway depends on your Equalizer configuration:

- If you have one standalone Equalizer in **single network** configuration, then the default gateway on each server should be Equalizer's internal address. If the single network Equalizer is also in a **failover** configuration with another Equalizer, then set the default gateway on each server to the failover alias IP address. (For more information, see "Setting Up a Failover Configuration" on page 53.)

- If you have one standalone Equalizer in **dual-network** configuration with the **spoof** option enabled on one or more clusters, then the default gateway on each server in the clusters with **spoof** enabled should be Equalizer's internal address. If you have two dual network Equalizers in a **failover configuration**, set the default gateway on each server in the clusters to the failover alias IP address. (For more information, see "Setting Up a Failover Configuration" on page 53.)

- In other configurations, the default gateway to use on servers depends on your network topology rather than on the parameters set on Equalizer.

The way that you configure routing on a server depends on the server's operating system, but usually involves some form of the `route` command; check your operating system documentation. To verify that you have configured a server's routing correctly, trace the route from the server to a destination address outside the internal network to ensure that Equalizer gets used as a gateway. On UNIX systems, use the `traceroute` utility; on Windows, use `tracert`.

Configure routing on each server from the system console, not through a telnet session. This will avoid any disconnects that might otherwise occur as you change the network settings on the server.

Configuring DNS and Firewalls for Envoy

If you are configuring Equalizer to use Envoy for geographic load balancing, you need to configure your authoritative domain name server to delegate authority to the Envoy sites. If you will use Envoy across firewalled networks, you also need to configure the firewalls to allow traffic between Envoy sites and between the Equalizer and clients.

Configuring the Authoritative Name Server to Query Envoy

To delegate authority to the Envoy sites, you must configure the authoritative name server(s) for the domains that are to be geographically load-balanced. You also must delegate each of the fully-qualified subdomains to be balanced.

For example, assume that you want to balance `www.coyotepoint.com` across a geographical cluster with two Envoy sites, `east.coyotepoint.com` and `west.coyotepoint.com`. In this case, you configure the name servers that handle the `coyotepoint.com` domain to delegate authority for `www.coyotepoint.com` to both `east.coyotepoint.com` and `west.coyotepoint.com`. When a client asks to resolve `www.coyotepoint.com`, the name servers should return name server (NS) and alias (A) records for both sites.

Using Geographic Load Balancing with Firewalls

Envoy sites communicate with each other using Coyote Point's UDP-based Geographic Query Protocol. Similarly, Equalizer sites communicate with clients using the DNS protocol. If a network firewall protects one or more of your sites, you must configure the firewall to permit Equalizer packets to pass through.

To use geographic load balancing with firewalled networks, you need to configure the firewalls so that the following occurs:

- Equalizer sites communicate with each other on UDP ports 5300 and 5301. The firewall must allow traffic on these ports to pass between Envoy sites.

- Equalizer sites and clients can exchange packets on UDP port 53. The firewall must allow traffic on this port to flow freely between an Equalizer server and any Internet clients so that clients trying to resolve hostnames via the Equalizer DNS server can exchange packets with Equalizer sites.

Equalizer sites can send ICMP echo request packets (i.e., a 'ping') through the firewall and receive ICMP echo response packets from clients outside the firewall. (When a client attempts a DNS resolution, Equalizer sites send an ICMP echo request packet to the client; the client might respond with an ICMP echo response packet.)

Testing Your Basic Configuration

Once you have installed and configured Equalizer and your servers, perform tests to verify that Equalizer is working properly.

To perform these tests, you need the following:

- A test machine on the internal network (the same physical network as the servers; one of the server machines can be used for this purpose).

- If you have a two-network configuration, a test machine on the external network.

- A client machine somewhere on the Internet, to simulate a "real-world" client. This machine should be set up so that the only way it can communicate with your servers or Equalizer is through your Internet router.

Then follow these steps:

1. Ping Equalizer's external address (if configured) from a host on the external network interface address.

2. Ping Equalizer's internal address from a host on the internal network interface address.

3. If DNS is configured, ping a host on the Internet (e.g., `www.coyotepoint.com`) from Equalizer to ensure that DNS and the network gateway are functioning properly.

4. From the internal-network test machine, ping the physical IP address of each server. You should be able to successfully ping all of the servers from the test machine.

5. From the internal-network test machine, ping the server aliases on each of the servers. You should be able to successfully ping all of the servers from the test machine using their aliases.

6. From the internal test machine and each of the servers, ping the Equalizer address that you use as the default gateway on your servers. (If you use a two-network topology, this will be Equalizer's internal address or failover alias.)

7. From the internal-network test machine, connect to the server aliases on service ports of running daemons (you may need to configure telnet or ssh services on Windows servers). You should be able to connect successfully to the server aliases.

8. If you use a two-network configuration: From the external-network test machine, ping a physical server IP address using `ping -R` to trace the route of the ping. The Equalizer IP address should appear in the list of interfaces that the ping packet traverses. You can also use the `traceroute` (UNIX) or `tracert` (Windows) tools to perform this test.

9. Log into the Administrative Interface on either the external (if configured) or internal interfaces, as described in "Logging In and Navigating the Administrative Interface" on page 31.

Using the Administration Interface

Use Equalizer's HTML-based Administration Interface to perform the monitoring and administrative tasks described in the subsequent chapters of this guide. This chapter contains the following sections that show you how to log in and configure access to the interface:

Logging In and Navigating the Administrative Interface

The Equalizer Administration Interface can be opened in any Javascript-enabled browser. Two default logins are provided: the **look** login provides read-only access to the interface, and the **touch** login lets you view and edit the configuration. (The section "Managing Multiple Interface Users" on page 36 shows you how to add additional logins as well as define the resource that any login can view or edit.)

Logging In

To log into Equalizer, follow these steps:

1. Open a Javascript-enabled web browser. We recommend you use one of these browsers:

   • Internet Explorer Version 6 or later

   • Firefox Version 2 or later

2. From the browser, load the URL that corresponds to Equalizer's internal or external network interface address, using either the `http` or `https` protocols. If you are using a pair of Equalizers in a failover configuration, you can also use the failover IP alias to ensure that you log into the Equalizer that has the primary role.

   For example, if the one of the internal, external, or failover IP addresses is `199.146.85.2`, open the Equalizer Administration Interface by typing `http://199.146.85.2` or `https://199.146.85.2` in the appropriate location in the browser. Use the `https` protocol to access the interface using SSL and a server certificate. This is recommended when accessing Equalizer over a public network (such as the Internet).

Equalizer displays the login screen:
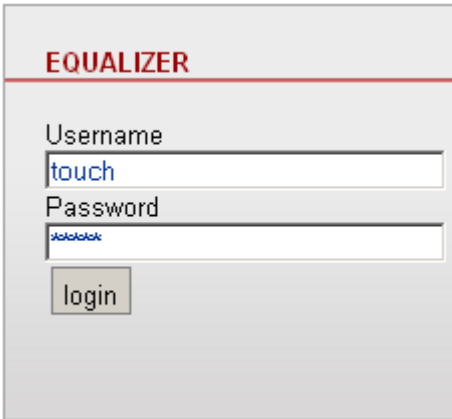
Figure 15   The login screen

Note that you can restrict the Administrative Interface to accept logins on a subset of the available interfaces and protocols; see "Managing Equalizer Access" on page 34.

3. Enter **touch** (administrator), **look** (read-only), or another defined login as **Username**. Enter the **Password** for the login. Click the **login** button to log into Equalizer.

> **Note –** Initial passwords for the **touch** and **look** logins are "touch" and "look", respectively. These passwords can be changed and additional user logins defined as shown in "Managing Multiple Interface Users" on page 36.

4. The **Home** screen of the Administrative Interface is displayed:
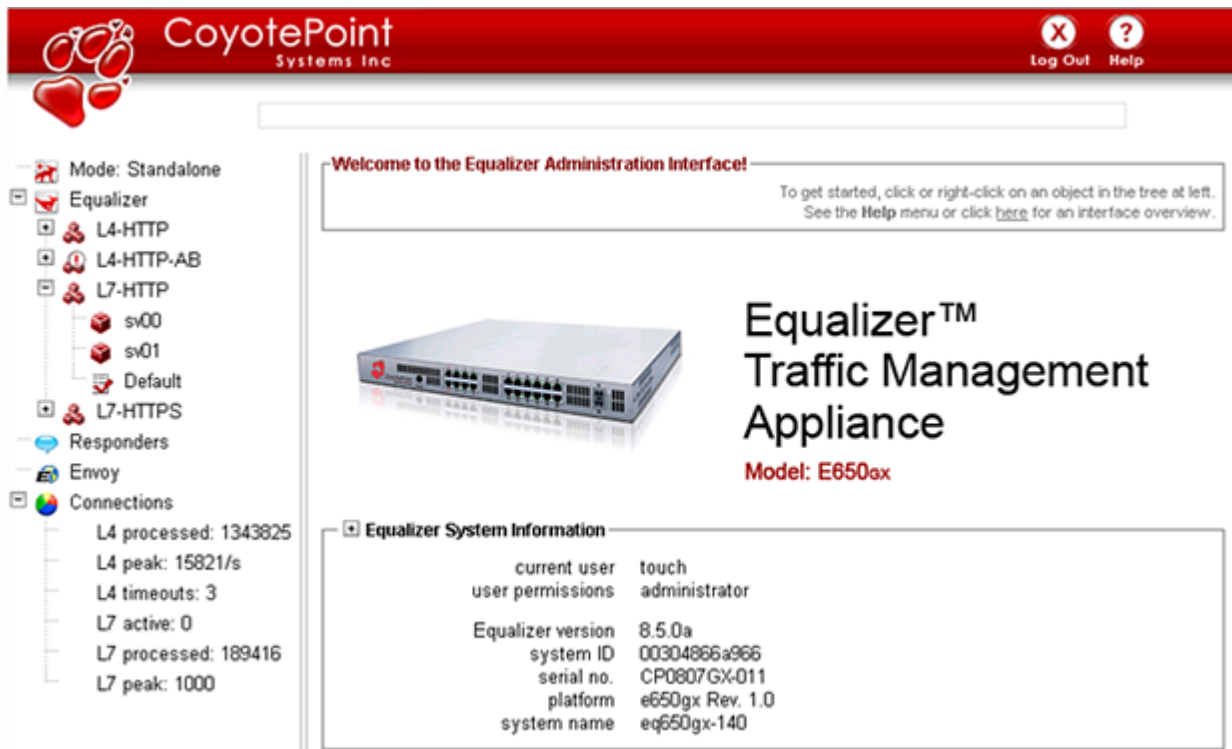
Figure 16   The Home Screen of Equalizer's Administration Interface

Navigating Through the Interface

The Equalizer Administration Interface (see Figure 16) is divided into three major sections:

1. The left side of the screen displays a hierarchical list of objects, as explained below.

   - **Mode: Standalone** or, if failover is configured, the *Failover Peer Name* defined for the other Equalizer in the failover pair.
     - The Coyote Icon indicates the failover status of the peer: a sitting Coyote means the peer is in backup mode (or failover is not enabled); a running Coyote means the peer is in primary mode.
     - Click this item to open the **Failover** configuration tab on the right hand side of the screen.
     - Right-click this item to open a menu that displays the currently configured IP addresses and a menu of failover commands.

   - **Equalizer** or, if failover is configured, the *Failover Peer Name* defined for this Equalizer:
     - The Coyote Icon indicates the failover status of this Equalizer: a sitting Coyote means this Equalizer is in backup mode; a running Coyote means this Equalizer is in primary (or standalone) mode.
     - Click this item to open the **Clusters > General** tab in the right frame -- a summary of the configuration and status of all currently defined clusters and servers.
     - Click the plus sign (+) next to **Equalizer** to open a list of currently defined clusters.
     - Click the plus sign next to a cluster name to open a list of currently defined servers and (for Layer 7 clusters) a list of Match Rules.
     - Click a cluster, server, or match rule name to open the management tabs for that object.
     - Right-click on this item to display the internal and external IP addresses, whether the system is in dual or single network mode, and the **Add Cluster** command.

   - The **Responders** item:
     - Click **Responders** to open the **Responders** management tab, where you can add, modify and delete Responders. Right-click to open a menu containing the **Add New Responder** command.
     - Click the plus sign next to **Responders** to display a list of currently defined Responders in the tree.
     - Click a **Responder** name to open that Responder's configuration tab.

   - The **Envoy** item (if the optional Envoy geographic load balancing software is installed):
     - Click the plus sign next to **Envoy** to display a list of currently defined GeoClusters in the tree.
     - Click the plus sign next to a GeoCluster name to display a list of the sites defined for the GeoCluster.
     - Click a GeoCluster or Site name to open the configuration tabs for that object.

   - The **Connections** item:
     - Click on **Connections** to open the *Equalizer* **> Status > Statistics** screen.
     - Click on the plus sign to display L7 Statistics and L4 Statistics.

     See "Displaying Global Connection Statistics" on page 141.

2. The top of the Administrative Interface screen displays the following buttons:

   - **Alert**: Displayed when a critical system message has been logged. Clicking on this icon displays the text of the system message.

   - **Log Out**: Logs you out of the Administrative Interface.

   - **Help**: Displays a sub-menu of commands**:**
     - **View Guide**: opens the *Equalizer Installation and Administration Guide* (this book) in PDF.
     - **View Release Notes**: opens the *Release Notes* for the currently installed version of Equalizer in PDF.
     - **View Transition Guide**: opens the *Equalizer Version 8 Transition Guide*, written to help Version 7 users locate Version 7 functionality in the Version 8 Administrative Interface.

- **Context Help**: displays the section in the *Equalizer Installation and Administration Guide* PDF file corresponding to the screen currently displayed in the right frame.
- **About**: the Equalizer **Home** screen displayed when you first log into the Administrative Interface.

3. The right hand side of the Administrative Interface initially displays the **Home** screen as shown in Figure 16 on page 32. For a description of the information contained on the **Home** screen, see "Displaying Equalizer System Information" on page 136.

- Click on any item in the left frame, or right click to choose a command for that object. The right frame will display the management tabs for the object or the appropriate command dialog.

- The easy-to-use management tabs organize configuration information into forms and tables that make configuring Equalizer simple. Sub-tabs provide a second level of organization within top-level tabs.

The following section shows you how to enable and disable access to the Administrative Interface over the available IP addresses and protocols, using the **Permissions** tab.

Managing Equalizer Access

You can control the IP addresses and protocols on which the web-based Administrative Interface runs, and the IP addresses over which SSH (Secure Shell) access to Equalizer is permitted.

GUI Access

By default, the Administrative Interface is available via HTTP and HTTPS protocols, on the Equalizer's:

- internal IP address
- external IP address (if configured)
- failover IP address (if configured)

To view or change the Web Interface Access settings, do the following:

1. Log into the Administrative Interface over one of the currently configured IP addresses. Use a login that has read or write access to global parameters (see "Objects and Permissions" on page 37).

2. Select *Equalizer* > **Permissions** > **GUI Access**. The following form is displayed in the right frame:

Figure 17   The web interface access flags

The flags have the following functions:

- **external** enables access via the external interface IP address
- **internal** enables access via the internal interface IP address

- **failover** enables access via the internal and/or external failover IP address, depending on whether one or both of the **external** or **internal** flags are also enabled; if neither flag is enabled, then neither failover address is enabled
- **http** enables access via the **http://** protocol.
- **https** enables access via the **https://** protocol.

You can enable or disable access for an interface that is not currently configured. You can check the currently configured IP addresses by right-clicking on the *Equalizer* in the left frame Configuration Tree.

The interface returns an error if you attempt to disable access for the IP address or protocol you are using for the current browser session; in this case, you need to log in using another IP address or protocol.

3. Click **Commit** to save your changes.

It is possible to disable all access to the Administrative Interface only by manually editing the Equalizer configuration file (*eq.conf*). To re-enable access in this case, see "Restoring IP Access to the Administrative Interface" on page 256.

SSH Access

If you enable the **eqsupport** login (see "Managing the Remote Access Account" on page 28), you can use SSH (Secure Shell) to log into the Equalizer console. By default, SSH login is permitted on the Equalizer's:

- internal IP address
- external IP address (if configured)
- failover IP address (if configured)

To view or change the SSH access settings, do the following:

1. Log into the Administrative Interface over one of the currently configured IP addresses. Use a login that has read or write access to global parameters (see "Objects and Permissions" on page 37).

2. Select *Equalizer > Permissions > SSH Access*. The following form is displayed in the right frame:

Figure 18   The SSH access flags

The flags have the following functions:

- **internal** enables access via the internal interface IP address
- **external** enables access via the external interface IP address
- **failover** enables access via the internal and/or external failover IP address, depending on whether one or both of the **external** or **internal** flags are also enabled; if neither flag is enabled, then neither failover address is enabled

You can enable or disable access for an interface that is not currently configured. You can check the currently configured IP addresses by right-clicking on the *Equalizer* in the left frame Configuration Tree.

3. Click **Commit** to save your changes.

Updating the Administration Interface Certificate

The Administration Interface is delivered with a default SSL certificate for **https//:** connections. Clients use this certificate to authenticate a connection with the interface. You can replace this certificate by doing the following:

1. Log in to Equalizer using a login that has **add/del** access on global parameters (see "Logging In" on page 31).

2. Select *Equalizer* **> Maintenance > Certificates**.

3. Use the **Browse** button to select the certificate file from your local file system. The certificate file must be in PEM (*.pem*) or PKCS12 (*.pfx*) format, and must contain the private key and the entire certificate chain. (For more information on certificates, see Appendix E, "Using Certificates in HTTPS Clusters".)

4. Select **upload** to install the new certificate on Equalizer. You'll be prompted for a password, which is the password (or pass phrase) you provided when you generated the CSR for the certificate (or created the self-signed certificate).

   Note: If you select a file that is not in PEM or PKCS12 format (or select no file at all), the following error message is displayed:

   ```
   Certificate must be in PEM or PKCS12 format.
   ```

   Following the error message is the output the SSL tools returned after they were run on the uploaded file. The output may be unreadable and poorly formatted; this is normal, because the file you uploaded was not in the correct format. Click **dismiss** on the error popup and then go back to the previous step to select a file that is in PEM or PKCS12 format.

Once the certificate file is successfully uploaded to Equalizer, the tab displays the certificate information at the bottom of the screen.

Managing Multiple Interface Users

Equalizer is shipped with two logins for the browser based Administrative Interface: **look** (read-only mode) and **touch** (administrator or edit mode). The definitions of these users and any additional users you create specify the permissions each has on Equalizer objects.

On Equalizer, there are two object types that are assigned permissions: clusters and global parameters. Cluster parameters include all cluster settings and the settings for the servers in the cluster. System parameters include network interface settings and user definitions. As installed, the **touch** user can add, modify, and delete global parameters, as well as add, modify, and delete clusters. The **look** user has read-only access to all global parameters and clusters.

These two logins are usually sufficient for sites that have a small number of system administrators. For sites where multiple administrators with different responsibilities exist, you can create additional logins that reflect the administrative roles assigned to each user who logs in to Equalizer.

Let's say, for example, that your site has one person who is responsible for the overall administration of Equalizer's clusters, users, and operating parameters (the Equalizer Administrator), and several junior system administrators, each of which is responsible for maintaining a single cluster (the Cluster Administrators). The Equalizer Administrator could use the **touch** login to create additional logins for each Cluster Administrator, and give each login permission to modify the configuration of a single cluster only.

Objects and Permissions

The following table shows the permissions and objects defined on Equalizer:

Permissions	Objects
none read write add/del	global parameters ALL cluster parameters

The permission set on the **ALL** object specifies the user's permission on all clusters with their permission set to **none** (the default), unless a different permission is set on the cluster. The table on the following page explains the permissions used on objects.

none	The user cannot view, modify, or delete the object. **For global parameters**: the user cannot view any of the global parameter tabs displayed when you click on *Equalizer* in the left frame. **For clusters:** the left frame and all global tabs display only clusters that the user has been given explicit permission to view by assigning a higher permission to those clusters. When **none** is set for the **ALL** object, the user cannot view any clusters on the system. This can be overridden by setting a higher permission on individual clusters.
read	The user can only view the object's definition. **For global parameters**: the user can open all of the global parameter tabs displayed when you click on *Equalizer* in the left frame, but cannot use the **commit** button to make any changes. **For clusters**: cluster definitions for which the user has **read** permission are displayed in the left frame and all global tabs. The user can select clusters and view their definitions. A user with the **read** permission set for the **ALL** object has **read** permission on all clusters, unless **write** or **add/del** is set on an individual cluster.
write	In addition to **read** permission, the user can modify existing objects, but cannot add new objects or delete existing objects. **For global parameters**: the user can update all global parameters (including parameters that are not already assigned a value). The user cannot, however, add or delete global objects (for example: logins, clusters, and responders). **For clusters**: the user can modify the values assigned to all cluster parameters (including parameters that are not already assigned a value). The user cannot add or delete a cluster object (for example, a server or match rule.) A user with the **write** permission set for the **ALL** object has **write** permission on all clusters, unless **add/del** is set on an individual cluster.
add/del	In addition to **write** permission, the user can add new objects and delete existing objects. **For global parameters**: the user can add and delete global objects (for example: logins, clusters, and responders). **For clusters**: the user can add and delete a cluster object (for example, a server or match rule.) A user with the **add/del** permission set for the **ALL** object has **add/del** permission on all clusters with their permission set to **none**; if a permission other than **none** is set on the cluster, then the cluster permission applies.

Figure 19   Permissions in user definitions

Viewing or Modifying Login Permissions

To view or modify the permissions for a login, do the following:

1. Log into the Administrative Interface over one of the currently configured IP addresses. Use a login that has at least read access for global parameters (see "Logging In" on page 31).

2. Select **Equalizer > Permissions > Users**. The following table is displayed:

Figure 20   Users table

3. To view or modify login details, select the modify icon in the **Actions** column in the same row as the login name you want to view. The user definition appears, as shown in this example for the default **touch** login.

This screen contains the following information about the login:
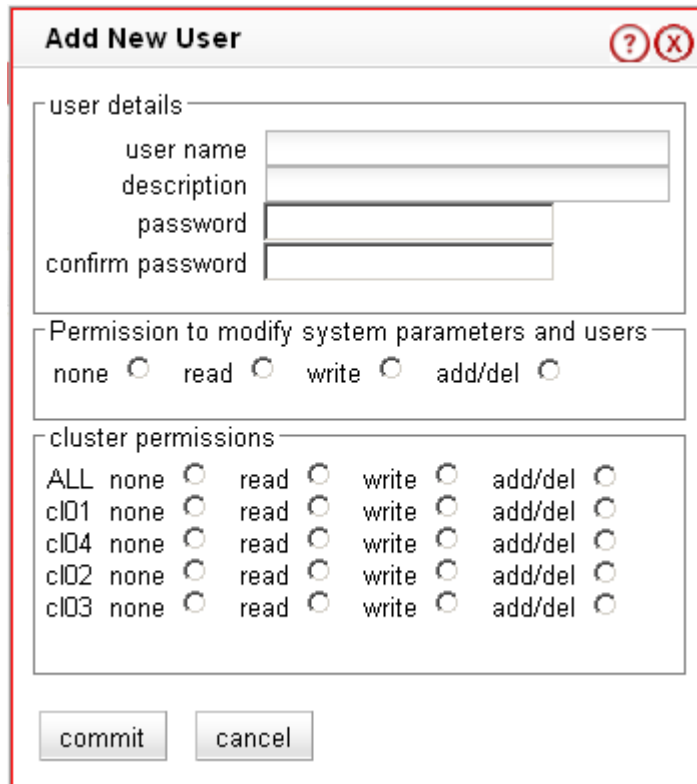
user details	The **description** field contains a text description of the purpose of the login. The **password** field is empty when viewing a login definition.
permission to modify system parameters and users	Specifies the permission the user has on the global system parameters (displayed when you select **Equalizer > Global Configuration**).
cluster permissions	**ALL** specifies the minimum permission the user has on all clusters below. Below **ALL** is a line for each cluster that specifies the permission that the user has on that cluster. If the permission on the cluster is to the left of the permission given for **ALL**, then the **ALL** permission applies to the cluster instead.

The screen above shows that the default user **touch** has **add/del** permission on system parameters and user definitions, and **add/del** on **ALL**. This means that **touch** has complete control over the system parameters, the user definitions, and all the clusters on Equalizer.

4. If you make any updates to the login password or permissions, click **Commit** to save your changes. Otherwise, click **Cancel** to return to the **Users** table.

Adding a Login

1. Log into the Administrative Interface over one of the currently configured IP addresses. Use a login that has at least read access for global parameters (see "Logging In" on page 31).

2. Select *Equalizer > Permissions > Users*.

3. Select the add icon ✚ . The **Add New User** screen like the following is displayed:

Figure 21 The add user screen

4. Type a **user name** and a **description** for the login. User names may only contain alphanumeric characters, periods (.), dashes (-), and underscores (_).

5. Type a **password** for the login and re-type it into the **confirm password** text box. Passwords must be between 6 and 128 characters long and should contain a mix of letters (uppercase and lowercase), numbers, and metacharacters. A blank password is not permitted.

6. Select the desired **permission to modify system parameters and users**. See the section "Objects and Permissions" on page 37 for help.

7. Select the desired **cluster permissions**. See the section "Objects and Permissions" on page 37 for help.

8. Select **commit** to save the user definition.

Deleting a Login

The Administrative Interface prevents you from deleting the login that you are currently using. For example, you cannot log in as **touch** and delete the **touch** login; to do this, you must log in using a different user name that has the **add/del** permission on users. This also prevents you from deleting all logins via the interface. However, it is possible that all user logins could be deleted by manually editing the configuration file, or in the unlikely event the configuration file becomes corrupted. If this occurs, the **eqadmin** utility can be used to create a new Administrators Read-Only login; see

1. Log in to Equalizer using a login other than the one you want to delete; the login you use must have the **add/del** permission on users (see "Logging In" on page 31).

2. Select *Equalizer* **> Permissions > Users**.

3. Select the delete icon 🗑 on the same row as the name of the user login you want to delete.

4. A confirmation box appears. Select **Commit** to delete the login.

Entering Names for Equalizer Objects

Equalizer identifies administrative objects, such as clusters and servers, by name. For example, object names and icons are displayed in a hierarchy in the Administrative Interface's left frame as described earlier in this chapter. Keep in mind the following guidelines when typing in a name for an Equalizer object:

- The characters used in names are limited to standard ASCII letters ('A' through 'Z' and 'a' through 'z'), numbers (0 through 9), and the characters '**.**' (period), '**-**' (dash) and '**_**' (underscore).

- The first character in a name must be a letter.

- Names can be at most 63 characters long.

- The readability of lists presented in the interface is increased by using short names that use as many unique characters at the beginning of the name as possible.

Chapter 4:

Configuring Equalizer Operation

This chapter describes the global parameters, resources, and procedures that you can use to specify Equalizer's operating characteristics and perform system maintenance tasks:

> **Note –** This chapter assumes that you have already initially configured Equalizer according to the instructions found in Chapter 2, "Installing and Configuring Equalizer Hardware." on page 21. See Chapter 3, "Using the Administration Interface." on page 31 for how to log into and use the web-based Administration Interface.

Licensing Equalizer

You must register and license your Equalizer before performing any other configuration using the Equalizer Administration Interface (described in Chapter 3, "Using the Administration Interface"). The License Manager helps you register and request a license, as well as view your current license information.
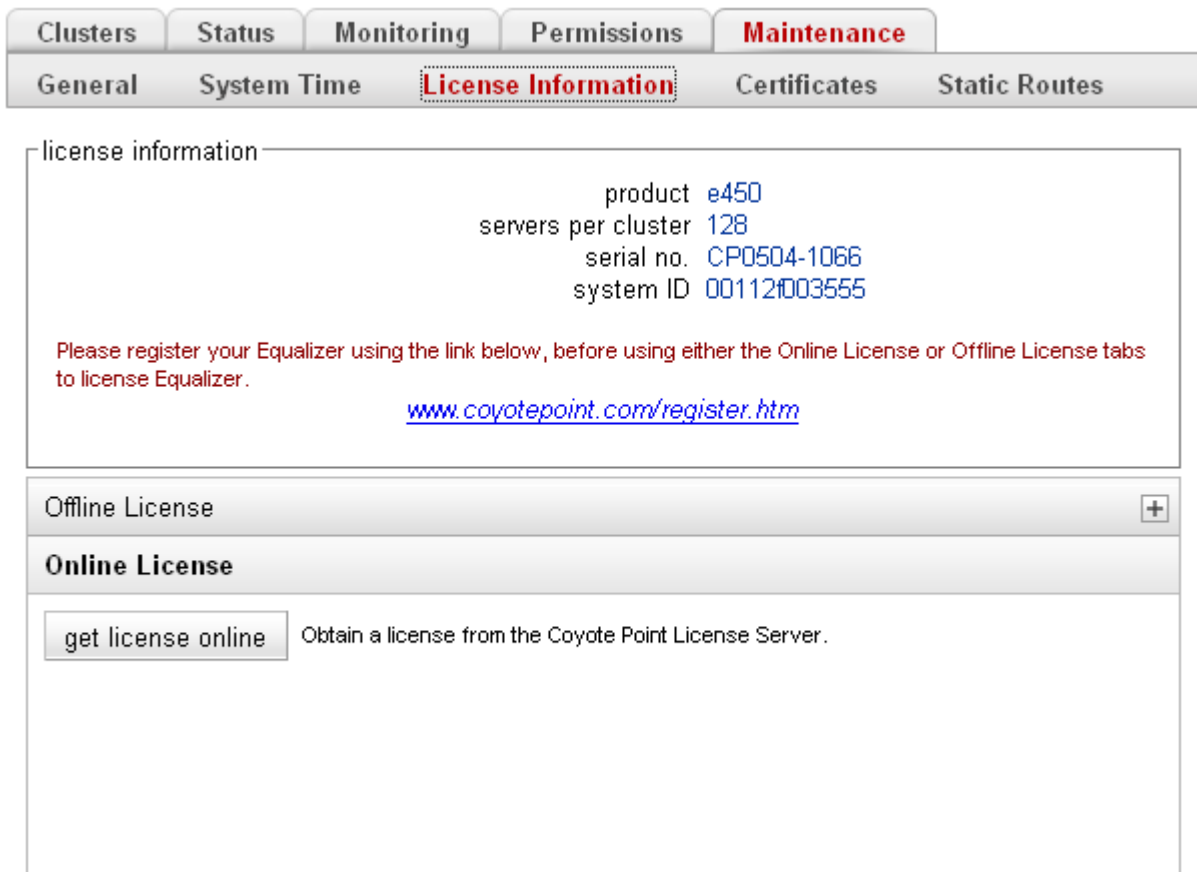
You'll need to request a license if:

- The left frame of the Equalizer Administrative Interface displays an unlicensed system error.

- You've purchased the Envoy Geographic Clustering product after previously licensing Equalizer.

- You want to upgrade to a new release that requires a new license.

You can register and license your Equalizer either online (using Coyote Point's Licensing Server, or offline using a license file obtained from Coyote Point Support. **In order to request a license online, your Equalizer needs to be able to connect to the internet on port 127.** You may need to work with your local network administrator to ensure that Equalizer can do this through any firewalls or other network devices on your network. If this is not possible, you can use the offline method.

Follow the procedure below to get a license:

1. Log into the Administrative Interface using the default **touch** login, or another login that has **add/del** access for global parameters to request a license; **read** access or greater to view (see "Logging In" on page 31).

2. Select *Equalizer* in the left frame and open the **Maintenance > License Information** tab in the right frame. The following screen is displayed:.

Figure 22   The License Information screen -- Online License

The top section of **license status** screen shows the following information for an already licensed system:

product	Equalizer product model number. Displays "unlicensed" if the system is not licensed or the current license is invalid.
servers per cluster	The number of servers per cluster allowed, as specified by your license.
serial no.	The serial number of the Equalizer unit (also printed on the back or bottom of the unit).
system ID	The internal system ID. [Note: in previous releases, the system ID was shown with a colon (:) separating each pair of numbers.]

If you don't need to license Equalizer, stop now. Otherwise, continue with the next step.

3. If your Equalizer is already registered with Coyote Point, skip this step.

    You must register your Equalizer before you can license it. Click on the link shown in the screen above to register Equalizer. Follow the prompts displayed by the Registration Web Site. You will need to copy the **system ID** and the **system serial number** into the registration form (see Figure 22 on page 44).

4. Do *one* of the following:

    a. If Equalizer is connected to the Internet and a DNS server is configured, click on the **get license online** button to request a license online. The license server will download your license automatically, and ask you if you want to reboot to apply the license. Select **Yes** to reboot.

    b. If Equalizer is not connected to the Internet or DNS is not configured, then see the section "Requesting a License Offline" on page 46, below.

After the system comes back up, there should be no unlicensed error in the left frame or on the **Help > About** screen. If you licensed Envoy, the **Help > About** screen should show **Envoy geographic load balancing enabled** when the **Equalizer System Information** box is expanded.

Requesting a License Offline

If your Equalizer is not currently connected to the Internet or if DNS is not configured for Equalizer, then you will need to request a license offline. To do this, follow this procedure:

1. Follow Steps 1 through 3 of the procedure above.

2. Select **Offline License** on the **License Information** screen:
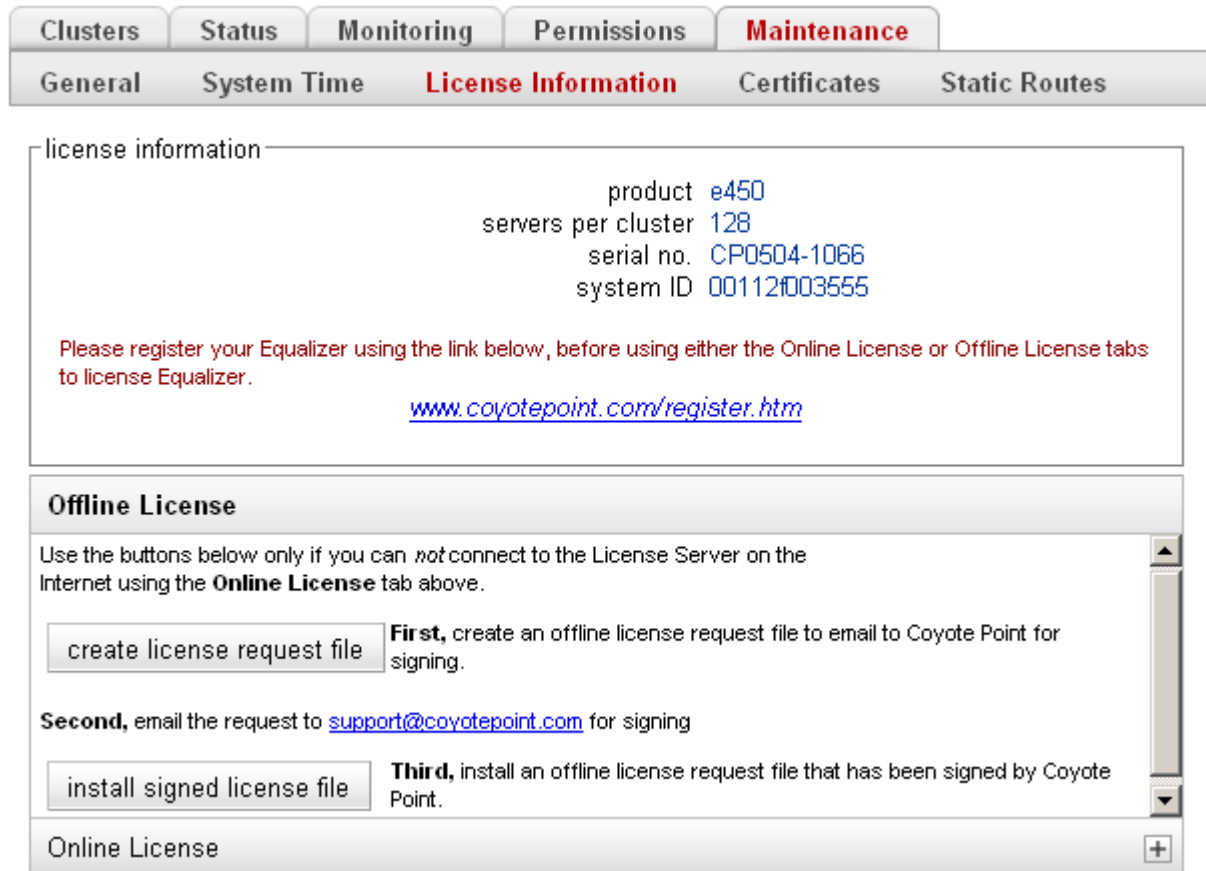
Figure 23 The License Information screen -- Offline License

3. Select **create license request file** and save the file to an appropriate location on your local system.

4. Select the **support@coyotepoint.com** link to open your browser's mail client, or open your email client manually and specify this address in the **To:** field of a new mail message. Specify **license request** in the **Subject** field, and attach the license request file you saved in the previous step. Send the email.

5. Once Coyote Point processes your request, you will receive a signed license file in a return email from Coyote Point. Save the licensing file you receive from Coyote Point to an appropriate location on your local system.

6. Select **install signed licensed file** and use the browse box to select the signed license file you saved in the previous step.

7. Equalizer installs the license and asks you if you want to reboot to apply the license. Select **reboot** to reboot.

After the system comes back up, there should be no unlicensed error in the left frame or on the **Help > About** screen. If you licensed Envoy, the **Help > About** screen should show **Envoy geographic load balancing enabled** when the **Equalizer System Information** box is expanded.

Modifying Global Parameters

Global or System Parameters are divided into two tabs, Probes and Networking. Most clusters will work with default values on these tabs. To view or modify the default global parameter values:

1. Log into the Administrative Interface using a login that has **add/del** access for global parameters to add, remove, and update parameters; **write** access to update parameters with existing values; **read** access or greater to view (see "Logging In" on page 31).

2. Select *Equalizer* > **Probes** to view/modify the global probe parameters, or *Equalizer* > **Networking** to view/ modify the global networking parameters.

3. Change the appropriate fields.

4. Click the **commit** button.

The following two sections explain the global probe and networking parameters.

Global Probe Parameters

Selecting *Equalizer* > **Probes** displays the global probe parameters:

The global probe parameters are described below:

probe interval	The target interval between TCP probes of a cluster that has been marked *failing* in the load balancing daemon's internal tables. If the server does not respond to **strikeout threshold** (see below) additional TCP probes after it is marked *failing*, then the server is marked down. These additional probes are at least **probe interval** seconds apart. This value is solely a target; the monitoring process adjusts itself based on a number of factors, including system load. The default value is 20 seconds.
probe timeout	The time in seconds that the probe daemon waits for a response from a server to a TCP or ACV probe. The default is 10 seconds.

strikeout threshold	The number of additional TCP probes sent to a server that is marked *failing* (see **probe delay**, below), and after which the server is marked *down* if no response is received. The default value is 2; must be between 1 and 6.
event interval	The number of seconds between evaluation of all Smart Control Events for all clusters. The default is 15 seconds.
probe delay	The minimum time in seconds (default is 10) between successive TCP probes of servers by the probe daemon. If a server fails to respond to a probe, the probe daemon marks it *failing* in its internal server status table. You can override this value for each cluster. Specifying 0 to 5 seconds for probe delay means a 5-second delay (due to the fact that Equalizer's probe daemon goes through a probing cycle about every 5 seconds). Specifying 6 or more seconds increases the delay to at least that number of seconds, plus additional time due to load, latency, and other factors.
agent delay	The minimum time in seconds (default is 10) between successive probes of server agents by the probe daemon. Specifying 0 to 5 seconds for agent delay means a 5-second delay (due to the fact that Equalizer's probe daemon goes through a probing cycle about every 5 seconds). Specifying 6 or more seconds increases the delay to at least that number of seconds, plus additional time due to load, latency, and other factors.
require agent response	Applies when a cluster uses either **server agents** or **VLB agents**. When you check this box, Equalizer will mark a server *down* when it receives no response from the appropriate agent. See Appendix A, "Server Agent Probes" and Appendix F, "Equalizer VLB".
ICMP probe	Enables probing servers using ICMP echo (ping) probes. These probes are 5 seconds apart. If a server does not respond to an ICMP probe, it is marked *down* only if there are no other probes (TCP, ACV, or server agent) active for the cluster.

See "Server Health Check Probes and Timeouts" on page 215 for a complete description of Equalizer's server health checks and the global probe parameters.

Global Networking Parameters

Selecting *Equalizer > Networking* displays the global networking parameters:

The global networking parameters are described below:

send buffer	Applies to L7 clusters and is the amount of memory in kilobytes reserved by each L7 proxy process to store outgoing data before it is placed on the network interface. Default: 32. Minimum: 4. Maximum: 128. If this value is set for a cluster, the cluster value overrides the global value.
receive buffer	Applies to L7 clusters and is the amount of memory in kilobytes reserved by each L7 proxy process to store data that has been received on an interface before it is processed by an L7 proxy process. Default: 16. Minimum: 4. Maximum: 128. If this value is set for a cluster, the cluster value overrides the global value.
connect timeout	The time in seconds that Equalizer waits for a server to respond to a connection request. Layer 7 clusters only, and does not apply to the E250. See "HTTP and HTTPS Connection Timeouts" on page 210.
client timeout	The time in seconds that Equalizer waits before closing an idle client connection. Layer 7 clusters only, and does not apply to the E250. See "HTTP and HTTPS Connection Timeouts" on page 210.
server timeout	The time in seconds that Equalizer waits before closing an idle server connection. Layer 7 clusters only, and does not apply to the E250. See "HTTP and HTTPS Connection Timeouts" on page 210.

idle timeout	The time in seconds before reclaiming idle Layer 4 connection records. See "Layer 4 Connection Timeouts" on page 213.
stale timeout	The length of time that a partially open or closed Layer 4 connection is maintained. If a client fails to complete the TCP connection termination handshake sequence or sends a SYN packet but does not respond to the server's SYN/ACK, Equalizer marks the connection as incomplete. See "Layer 4 Connection Timeouts" on page 213.
sticky netmask	Enables sticky network aggregation for a subnet.
	Sticky network aggregation enables Equalizer to correctly handle sticky connections from ISPs that use multiple proxy servers to direct user connections. When you enable sticky network aggregation, all the connections coming from a particular network are directed to the same server. (Typically, all the servers in a proxy farm are on the same network.)
	The **sticky netmask** value indicates which portion of the address Equalizer should use to identify particular networks. The mask corresponds to the number of bits in the network portion of the address:
	8 bits corresponds to a Class A network
	16 bits corresponds to a Class B network
	24 bits corresponds to a Class C network
	In previous versions of Equalizer, enabling sticky network aggregation was the equivalent of setting the sticky network aggregation mask to 24 bits (that is, Equalizer routed all connections from the same class C network to the same server).
	Sticky network aggregation is applicable only for Layer 4 TCP and UDP clusters. For Layer 4 clusters with the **spoof** flag disabled and for Layer 4 clusters configured for FTP, a sticky record is maintained for each connection whether sticky network aggregation is enabled or not.
	A potential drawback of using sticky network aggregation is that all users connecting through a particular proxy farm might be directed to the same server. In practice, this has not been a problem. Equalizer's load-balancing algorithms direct other visitors to different servers to keep the load balanced.
	Note – If you are using two Equalizers in a failover configuration, you must set the sticky network aggregation mask identically for both Equalizers.

enable outbound NAT	When outbound NAT (Network Address Translation) is enabled, Equalizer modifies all server responses, substituting the Equalizer's external interface IP address for the server IP address in each response. This option is disabled by default. Outbound NAT is usually necessary in **dual network mode**; for example, when you are using reserved IP addresses on your internal server network and public IP addresses on the external network. NAT ensures that external hosts won't see packets originating from non-routable IP addresses. There is a performance cost for using NAT, since Equalizer must modify every outbound packet's source address. In **single network mode**, outbound NAT should be *disabled*. Because the clusters and servers are all on the same subnet, NAT is not needed and may interfere with other features (such as **spoof**). **Note –** In a failover configuration, be sure to use the same outbound NAT setting on both units in case a failover actually occurs.
passive FTP translation	If your servers are on a network the outside world cannot reach, consider enabling Equalizer's passive FTP translation option. This option causes the Equalizer to rewrite outgoing FTP PASV control messages from the servers so they contain the IP address of the virtual cluster rather than that of the server.
ICMP drop redirects	tells Equalizer to drop (i.e., ignore) incoming ICMP redirect messages.
ignore case	applies to L7 clusters and is the global setting to ignore case in match expressions. You can override this value per cluster and per match rule. See Chapter 7, "Using Match Rules".
no outbound RST	applies to L4 clusters only and causes Equalizer to disable forwarding of untranslated TCP RST (reset) packets. You may want to enable this flag if other network devices (e.g., firewalls, routers, etc.) are logging unexpected source IP messages for the real IPs of servers behind Equalizer (and not the cluster IP). When Equalizer manages a cluster connection, it keeps a record of the connection so it can translate the source IP in a server response before forwarding it. If a client connected to a server IP directly, or if the server sends a RST after Equalizer has already removed the connection record, the RST packet will not be translated by Equalizer. Enabling this option tells Equalizer to drop any RST packets from servers that do not currently have a Layer 4 connection record that matches the RST packet; with this option disabled (the default) Equalizer will forward all RST packets.
allow extended chars	By default, support for extended characters (8-bit ASCII and multibyte UTF characters) in URIs is disabled. Equalizer returns a **400 Bad Request** error when a request URI contains 8-bit or multibyte characters. To enable support for 8-bit and multibyte characters in URIs, turn on the **allow extended chars** flag. **Caution –** There are potential risks to enabling this option, because it allows Equalizer to pass requests that violate RFC2396; load-balanced servers may be running software that is incapable of handling such requests. Therefore, ensure that your server software is capable of handling URIs containing extended characters and will not serve as a potential weak point in your network *before* you enable extended characters.

abort server	By default, when a client closes a connection, Equalizer waits for a response from the server before closing the server connection. If this flag is enabled, Equalizer will not wait for a response before closing the connection to the server; instead it sends a TCP RST (reset) to the server when the client closes the connection.
RST on server failure	Applies to Layer 4 clusters only and enables the sending of TCP RST (reset) packets to clients on established connections when the server on the other end of the connection goes down. The RST packet is sent when Equalizer removes the connection or when another packet using the same connection is received from the client, whichever happens first. By default, this option is disabled and Equalizer does not send RST packets to clients. Enabling this option is useful when load balancing an application that requires a TCP RST to close a connection; for example, Network File System (NFS).

Setting Up a Failover Configuration

You can set up two Equalizers in a hot backup, or failover, configuration. In such a configuration, one of the systems handles incoming requests (the primary system), while the other (the backup system) waits for a failure to occur and automatically takes over if the Equalizer that is currently handling requests fails. The two Equalizers are called *failover peers* or *siblings* in such a configuration.

To use a second Equalizer as a hot backup or failover peer, you need to install both Equalizers so their network interfaces have corresponding configurations (see Figure 10 on page 14). The two peers need to be able to communicate over the configured failover aliases.

> **Note –** Be sure that you do *not* create a loop between the external and internal interfaces of the two units when connecting them to the network.

You must designate one of the Equalizers as the *preferred primary*; the second is the *preferred backup*. When you boot both Equalizers at the same time, the preferred primary Equalizer is activated. If the primary Equalizer fails, the backup takes over. When you bring the failed unit back online, it assumes the backup role until another failure occurs or you reboot its peer.

A failover configuration requires one or two additional IP addresses, called the *failover aliases*. In a dual network configuration, failover aliases must be supplied for both the internal and external interfaces; in a single network configuration, only an internal alias is needed. These IP addresses are initially assumed by the preferred primary system and are used as the network-visible interfaces of the Equalizer, instead of the addresses assigned to the individual Equalizers via the **eqadmin** interface. When a failover occurs, the failover aliases are then assumed by the backup system.

When Equalizer is brought online, it checks to make sure that the configured network interfaces are link active. In the case of the internal interface, Equalizer attempts to ping a configured server or failover peer. If the interfaces are not active, Equalizer sits in a loop waiting for them to become active (and sends comments to the console). Once the network interfaces are active, the failover peers begin a negotiation in which one system becomes the primary unit and the other becomes the backup unit. This is accomplished by the backup system performing a reboot.

When a backup Equalizer loses contact with its failover peer, it tries to determine the cause. If it cannot identify the cause, it will try to assume the primary role. It checks that no other system has configured the gateway IP address or virtual cluster addresses. If these tests are successful, the Equalizer assumes those IP addresses and starts handling traffic.

A *partition* occurs when both systems are unable to communicate with each other and both Equalizers enter primary mode. When the partition is healed and both units regain communication, the two systems resolve this dispute by choosing one system to reboot itself. Generally, this means that the system that is configured as the default backup will reboot; upon coming back up, it will enter backup mode.

> **Note –** Any switch, such as one from Cisco or Dell, that comes with Spanning Tree enabled by default can cause a communication problem in a failover configuration when one or both of the Equalizers are dual-port models. This problem occurs at bootup because the switch disables its ports for roughly 30 seconds to listen to BPDU (bridge protocol data unit) traffic. The 30-second pause causes both Equalizers to attempt to become the primary unit; the default backup continually reboots.
>
> To repair this condition, either disable Spanning Tree or enable PortFast for the ports connected to the Equalizers. This enables the ports to act as normal hubs and accept all traffic immediately.

Equalizer Models, Software, and Failover

Since different Equalizer models and software revisions have varying configuration parameters, it is recommended that both of the failover peers are the same model Equalizer hardware running the same software version. So, for

eaxmple, if you already have one E350GX running Version 8.5 and want to add another, you should add an E350GX running Version 8.5.

Some Equalizer software versions are supported on multiple hardware platforms. We do not, however, support mixing hardware platforms in a failover pair, with one exception: GX hardware models can be used in a failover pair with legacy 'si' hardware models. For example, you could run an E350GX and an E350si in a failover pair, both running software Version 8.5. If you do so, you must be careful to disable configuration file transfer between the two units as discussed in the section "Using Failover with Different Hardware or Software" on page 58.

Setting Up Failover

You'll need to create the two failover peer definitions and define the failover aliases on both systems. The following procedure leads you through the failover setup process on both Equalizers in the failover pair.

1. Log into the Equalizer Administration Interface on the failover peer that will assume the *preferred primary* role. Use a login that has **add/del** access on global parameters to initially define the configuration, or **write** access on global parameters to update an existing failover configuration. (Configuring and rebooting the preferred primary Equalizer first ensures that it assumes the primary role.)

2. Select **Mode: Standalone** (or the *Failover Peer Name*) at the top of the left frame Configuration Tree. The **Peers** tab looks like this when failover has never been enabled:
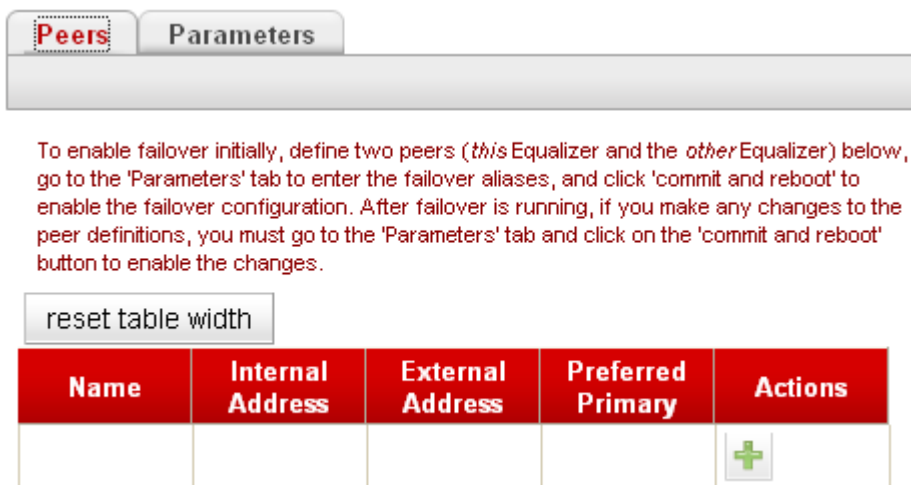
Figure 24   Peers tab

3. To add a new peer, click on the **Add** icon; the following dialog is displayed:

Figure 25   Add New Peer dialog

4. Select **This Equalizer** and click on the **Next** icon [>]

5. Enter the following information:

Peer Name:	A unique name for this failover peer. We suggest the system host name, but any name can be used.
Peer Internal IP Address:	The IP address of this peer's internal interface.
Peer External IP Address:	The IP address of this peer's external interface. **Note**: this only appears when Equalizer is in dual network mode (i.e., the external interface is configured).
Preferred Primary:	Indicates that this system should assume the primary role when both peers come up together. Check this box for this system; we'll leave it unchecked for the next.

6. Click the **Next** icon; a confirmation box is displayed. Make sure the information is correct and then press **commit** to add the definition to the **Peers** table.

7. Click on the **Add** icon. Since you've already defined this Equalizer in the table, the other peer information dialog is displayed. Enter the following information:

Peer Name:	A unique name for the other failover peer. We suggest the system host name, but any name can be used.
Peer Internal IP Address:	The IP address of the other peer's internal interface.
Peer External IP Address:	The IP address of the other peer's external interface. **Note**: this only appears when Equalizer is in dual network mode (i.e., the external interface is configured).

Preferred Primary:	Indicates that this system should assume the primary role when both peers come up together. Do not Check this box for this system (the backup system).

8. Click the **Next** icon; a confirmation box is displayed. Make sure the information is correct and then press **commit** to add the definition to the **Peers** table.

9. Select the **Parameters** tab; the following is displayed:

Figure 26   Failover Parameters

Enter the following information:

internal address	One address on the internal network shared between peers (also called the **internal failover alias**). Useful as a default route for other machines on the internal network. In single network mode, used to log into the Administrative Interface of the current primary peer.
internal netmask	Network mask for the internal IP; defaults to the netmask defined for the internal interface.
external address	One address on the external network shared between peers. Also called the **external failover alias**. Typically used to log into the Administrative Interface of the current primary peer. If you are in dual network mode and running Envoy, the external failover alias is used for DNS queries to Envoy as well. (**Note**: only displayed and used when Equalizer is in dual network mode.)
external netmask	Network mask for the external IP; defaults to the netmask defined for the external interface. (**Note:** only displayed and used when Equalizer is in dual network mode.)
receive timeout	Time in seconds (default: 0.6) to wait for a heartbeat response from peer before timing out.
connection timeout	Time in seconds (default: 0.5) to wait for a connection attempt to the other peer to succeed before timing out.

probe interval	Time in seconds (default: 5.0) between successive heartbeat checks of the peer.
dont transfer	By default, changes committed to the configuration file on the primary system are transmitted to the backup system when the next heartbeat occurs. Enabling this flag tells Equalizer not to transfer configuration changes to the peer Equalizer. Used when the peer Equalizers are different hardware models or are running different software versions. This normally occurs only during the process of upgrading a failover pair to a new software version, and you want to upgrade the peers at different times to maintain service. After both peers are upgraded to the new release, you can disable this flag on both peers.

The internal and external failover alias addresses are unique IP addresses assigned to the failover pair, and are passed between them whenever a failover occurs. The Equalizer that is running in primary mode assumes these aliases. The servers should use the internal address (when in dual network mode) or the single address (when in single network mode) as their default gateway.

When either the **receive timeout** or the **connection timeout** occurs on the backup system, that counts as one "strikeout", and the system attempts to check the heartbeat on the primary peer again. If three strikeouts occur in succession, the backup takes the primary role).

You should accept the default failover timing parameters, and only change them if there is a problem with heartbeat detection between the peers. For example, if you notice the log files contain too many false positives (messages that Equalizer has regained contact with its peer) you may want to increase the values.

10. Click the **commit & reboot** button.

   Errors are reported when a failover configurations is not successfully committed. If successful, you will be prompted to reboot immediately. (Click the **cancel** button if you want to wait to reboot the Equalizer.)

> **Note –** Both Equalizers must reboot in order for the failover configuration to work. Also note that selecting the **commit & reboot** button on one of the peers does not cause the second Equalizer (the peer that is not the system being configured) to reboot.

   As the Equalizer reboots:
   - Watch the console for messages indicating that the Equalizer has successfully assumed the primary (or backup) role.
   - Check the event logs (**View > Event Log** in the Administrative Interface) for each Equalizer to see that there are no related errors.
   - Make sure that "Successfully assumed PRIMARY role" appears in the log for the preferred primary system; the default backup system's log should contain "Successfully assumed BACKUP role".

11. Repeat this procedure on the default *backup* Equalizer peer starting at Step 1.

12. If you have not already rebooted the two Equalizers as part of the above procedure, reboot the *preferred primary* system first, then the *backup* system.

Modifying a Failover Configuration

To modify a currently running failover configuration, do the following:

1. Log into the Equalizer Administration Interface using login that has at least **write** access on global parameters to update an existing failover configuration.

2. Select the *Failover Peer Name* at the top of the left frame Configuration Tree. The **Peers** is displayed.

3. To modify a peer definition, select the **Modify** icon in the same row as the name of the peer you want to modify. Otherwise, go to the next step.

   Change the **Peer Name**, **Peer Internal IP Address**, **Peer External IP Address** (if in dual network mode), and **Preferred Primary** parameters as required. See the descriptions of these parameters in the previous section.

   Click **commit** to save your changes and return to the **Peers** table. Modifying a peer definition disables the current failover configuration and displays a warning at the top of the table. Disabled means that if the system reboots while the current failover configuration is disabled, it will start up in **standalone** (i.e., non-failover) mode.

   > **Note –** If both systems in a failover pair start in standalone mode, each will assume the cluster aliases and neither will assume a failover alias, resulting in no working clusters. To resolve this type of problem, configure and commit failover on both Equalizers, and then reboot both.

4. To modify failover parameters, or if you made any changes in the last step, click on the **Parameters** tab.

   Change the **internal address**, **internal netmask**, **external address**, ex**ternal netmask**, **receive timeout**, **connection timeout**, **probe interval**, and **dont transfer** parameters as required. See the descriptions of these parameters in the previous section. If no changes are needed, go to the next step.

5. If any changes were made above, you must select the **commit & reboot** button on the Parameters tab to save your changes and reboot the system with the new failover configuration.

Using Failover with Different Hardware or Software

We recommend that you use the same model Equalizer hardware (e.g., E350, E450, etc.) for both systems in a failover pair and that both Equalizers are running the same version of the software (e.g., 8.0.0). This is recommended because the default behavior of Equalizer is to maintain the same configuration files on both systems in a failover pair (so that you don't need to manually update both Equalizers with the same configuration changes). Changes committed to one system are copied to the configuration files on the other system.

For this reason, it is *not* generally recommended to deploy two different Equalizer models in a failover pair. However, some sites prefer to upgrade failover pairs to new hardware one at a time rather than deploying new models for both failover systems at the same time. If you are pairing an older model with a newer model (such as a newer switch-integrated E350 or E450 system with an older E350 or E450 non-switch system in single network mode), the differences in hardware configuration on these models *require* that the systems do not share changes to their configuration files by setting a special flag (**dont transfer**) on *both* Equalizers.

Similarly, some sites prefer to upgrade one Equalizer in a failover pair to a major new software revision and leave the other running the previous release for a limited period of time, in case there are any unforeseen configuration problems.

> **Note –** Whenever the **dont transfer** flag is enabled, you must manually perform any changes to your Equalizer and cluster configuration (such as adding/removing clusters or servers, changing system parameters, etc) on *both* Equalizers in the failover pair.

To prevent Equalizers in a failover pair from sharing changes to configuration files, perform the following procedure on both systems. We assume here that both systems already have a failover configuration defined:

1. Select the *Failover Peer Name* at the top of the left frame Configuration Tree, then select the **Parameters** tab in the right frame.

2. At the bottom of the screen, check the box labeled **dont transfer**.

3. Click the **commit & reboot** button to save the flag change.

4. Perform Steps 1 to 3 on the other Equalizer in the failover pair.

Upgrading Failover Configurations Prior to 7.2

The upgrade script contains facilities to migrate a version 7.1 format failover configuration (stored in */etc/eq.static*) to the new format used in 7.2 and later systems.

When the upgrade script runs, it will detect the presence of a valid configuration in the *eq.static* file. If it finds this file, the script prompts you whether to migrate the failover configuration.

If you respond 'y' to the upgrade script's prompt, the configuration file will be migrated to the upgrade partition, and the following message displayed:

```
IMPORTANT NOTE: configuration file transfers will be disabled when the system
reboots. You may re-enable configuration sharing by clearing the dont transfer
checkbox in the equalizer global parameters page.

If you are configuring failover between two different types of Equalizers, where
one contains a built-in switch and the other does not, configuration file
transfers must remain disabled between the two systems. (See release notes)
```

This indicates that when the system reboots, the **dont transfer** flag is set and any changes that are made to the configuration of this system will not be shared with the failover peer. You may clear the **dont transfer** flag once the system reboots, provided the failover pair is not both operating in single network mode and a combination of a switch-integrated system with a non-switch system. See "Using Failover with Different Hardware or Software" on page 58 for more information.

Changing the Network Mode between Single and Dual

It is important to delete the failover configuration before changing the network mode between single and dual network on an Equalizer that is already configured for failover. If the network mode is changed before the failover configuration is deleted, the web browser interface will become unusable because the configuration parser generates error messages stating that the failover configuration does not match the network mode.

Changing the Network Mode without Deleting the Failover Configuration

The procedure below shows you how to manually delete failover parameters from the Equalizer configuration file. It should only be used if the network mode was changed without first deleting the failover configuration. If you need help using the Equalizer command line interface, please contact Coyote Point Support (**support@coyotepoint.com**) and follow this procedure with the assistance of a member of the technical support team .

1. Log into the Equalizer via SSH using the *eqsupport* account (if enabled), or via the serial port using the *root* account.

2. Mount the root file system in read-write mode (if using the *eqsupport* account, you must use **su** first):

   ```
   # su
   # mount -w /
   ```

3. Edit the file */var/eq/eq.conf*:

   ```
   # ee /var/eq/eq.conf   (vi may be used as well)
   ```

4. Remove the `interface` stanza from the file (that is, the `interface` keyword, plus all the text between the curly braces that follow); an example `interface` stanza is shown below:

   ```
   interface {
     if_flags            = !disable;
     virt_intaddr        = "10.0.0.200";
     sibling this_eq {
       intaddr             = "10.0.0.199";
       extaddr             = <>;
       flags               = preferred_primary;
     }
   ```

```
      sibling the_other_eq {
        intaddr                 = "10.0.0.198";
        extaddr                 = <>;
        flags                   = !preferred_primary;
      }
    }
```

5. Save your changes to the file.

6. Enter:

    ```
    # shadow /var/eq/eq.conf
    ```

7. Reboot Equalizer:

    ```
    # shutdown -r now
    ```

After Equalizer comes back up, you can re-create your failover configuration.

Managing System Time and NTP

Through Equalizer's Administrative Interface, you can:

- set the time zone

- set the system date and time

- set up to three Network Time Protocol (NTP) servers, and enable or disable synchronization with these servers

NTP is a protocol designed to synchronize the clocks of computers over a network. NTP on Equalizer is compatible with servers running versions 1, 2, 3, or 4 of the NTP protocol. An RFC for NTPv4 has not been written; NTPv3 is described in RFC 1305.

On Equalizer, NTP is used primarily to time various operations, to ensure accurate timestamps on log entries (with respect to server and client log timing), and to allow for examination of the timing of log entries on two Equalizers in a failover configuration.

NTP on Equalizer works by polling an NTP server defined through the Administrative Interface. The time between polls of the NTP server is controlled by the **minpoll** and **maxpoll** NTP parameters, which default to 64 seconds (1 min 4 sec) and 1024 seconds (~17 mins), respectively. The behavior of NTP is to poll with a frequency starting at **minpoll** and then decrease polling frequency over time to **maxpoll**, as the accuracy of the local clock approaches the accuracy of the remote server clock. The time it takes for the polling delay to increase from **minpoll** to **maxpoll** will vary based on a number of factors, including the accuracy of the clocks on the client and server, network latency, and other timing factors.

NTP calculates when the local and remote system clocks are sufficiently in sync to begin increasing the polling delay towards **maxpoll**. When the accuracy of the two clocks is significantly different, or there is significant latency, for example, the two clocks may never be in sufficient agreement to increase the delay towards **maxpoll**. In this case, Equalizer will continue to sync approximately every 64 seconds. This behavior indicates that a different NTP server should be chosen.

We do NOT recommend changing the default **minpoll** and **maxpoll** delays in the NTP configuration file, in order to ensure an accurate system clock. NTP packets are very small and should not cause any problems with Equalizer or network operation, except as described in the following section.

NTP and Plotting

When you initially configure NTP, this may effectively disable plotting until NTP completes the initial synchronization of Equalizer's system clock with the NTP server -- which may take from several hours to several days. This is because plotting depends on accurate timestamps in the plot log. Since initially NTP is adjusting the time at frequent intervals, the timestamps in the plot log may become out of sync with the system clock, and so no plot data may be returned. Once NTP is no longer making adjustments to the system clock, plotting will function normally.

To manage system time on Equalizer, follow this procedure:

1. Log into the Administrative Interface using a login that has **add/del** access for global parameters (see "Logging In" on page 31).

2. Select *Equalizer* **> Maintenance > System Time**:

Figure 27   The System Time tab

3. To set the time zone, make a selection from the drop down box in the **timezone setting** section, and select the **commit** button in that section. To configure the system time or NTP, go to the next step.

4. You can set the system time manually or using a Network Time Protocol (NTP) server. Do one f the following:

   a. Use the drop-down boxes at the top of the **date and time** field to manually set the date and time. Make sure the **enable NTP synchronization** check box is disabled.

   b. Turn on the **enable NTP synchronization** check box and type in the name of an NTP server into the **primary server** text box. You can also specify two additional servers to be used in sequence if the first is unavailable. See the section "Selecting an NTP Server" on page 62 for help choosing an appropriate NTP server. The above is an example appropriate for locations in the United States.

5. Select the **commit** button in the **date and time** section to save your changes.

Selecting an NTP Server

We recommend that you specify NTP pool servers appropriate for your geographic location. Selecting a pool server means that you are specifying an alias that is assigned by **ntp.isc.org** to a list of time servers for a region. Thus, NTP

pool servers are specified by geography. The following table shows the naming convention for servers specified by continent:

Table 28:

Worldwide	pool.ntp.org
Asia	asia.pool.ntp.org
Europe	europe.pool.ntp.org
North America	north-america.pool.ntp.org
Oceania	oceania.pool.ntp.org
South America	south-america.pool.ntp.org

To use the continent-based NTP pool servers for Europe, for example, you could specify the following pool servers in Equalizer's **time configuration** screen:

```
0.europe.pool.ntp.org
1.europe.pool.ntp.org
2.europe.pool.ntp.org
```

You can also specify servers by country. So, for example, to specify a UK based time server pool, you would use:

```
0.uk.pool.ntp.org
1.uk.pool.ntp.org
2.uk.pool.ntp.org
```

Or, for the US, you would use:

```
0.us.pool.ntp.org
1.us.pool.ntp.org
2.us.pool.ntp.org
```

Be careful when using country based NTP pool servers, since some countries contain a very limited number of time servers. In these cases, it is best to use a mix of country and continent based pool servers. If a country has only one time server, then it is recommended you use a time server pool based in another nearby country that supports more servers, or use the continent based server pools.

For example, Japan has 6 (six) time servers as of the date this document was published. The organization that maintains time server pools recommends using the following to specify time server pools for Japanese locations:

```
2.jp.pool.ntp.org
0.asia.pool.ntp.org
2.asia.pool.ntp.org
```

For more information on choosing NTP pool servers, please see the NTP pool server web pages at:

```
http://ntp.isc.org/bin/view/Servers/NTPPoolServers
```

General System Maintenance

The *Equalizer* **> Maintenance > General** tab contains buttons for the system maintenance tasks described in the following sections:

Saving or Restoring Your Configuration

You can save your Equalizer configuration to an archive file or to load a saved configuration to restore a previous Equalizer configuration. When you save your configuration, Equalizer wraps up the following information in a binary file:

- `/var/eq/eq.conf`, which contains the cluster/server configurations that appear in the left frame of the administrative interface, the failover configuration, interface IP addresses, and GUI logins.

- `/var/eq/envoy.conf`, which is the Envoy configuration (if Envoy is installed); it contains geographic cluster and site information from the left pane of the administrative interface.

- `/var/eq/licenses`, which contains licensing information.

- Configuration files from `/etc` (including `hosts`, `master.passwd`, `ntp.conf`, `passwd`, `rc.conf-eq`, `resolv.conf`, `syslog.conf`) and `/etc/ssh` (including `ssh_config`, `sshd_config`, and host keys).

Backing Up Your Configuration

To back up your current configuration to a file, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for global parameters (see "Logging In" on page 31).

2. Select *Equalizer* **> Maintenance > General**. Then select the **Backup** button.

3. When prompted, specify the location and filename to use for the backup archive. The default backup archive name is of the form *hostname-mm.dd.yyyy-HH.MM*.bkp, where *hostname* is the Equalizer system name, *mm* is the month, *dd* is the day, *yyyy* is the year, *HH* is hours and *MM* is minutes. Click **OK** to save the backup archive.

Restoring a Saved Configuration

To restore a saved configuration, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for global parameters (see "Logging In" on page 31).

2. Select *Equalizer* **> Maintenance > General**. Then select the **restore** button.

3. Click **Browse...** to locate and select the previously created backup archive that you want to use to restore the Equalizer configuration.

4. Click **restore** to upload the configuration file. Equalizer automatically reboots to update the configuration.

> **Caution –** Be very careful when restoring configurations. The saved IP information could cause conflicts on the network if the restored file comes from another Equalizer (for example, the backup Equalizer in a failover configuration). If IP conflicts occur, use the console-based Equalizer Configuration Utility (**eqadmin**) to reconfigure the restored system's IP addresses. See Chapter 3, "Configuring Equalizer Hardware".

Shutting Down Equalizer

Before turning off Equalizer or disconnecting the power, you should perform a clean shutdown. Once Equalizer shuts down, it must be power cycled to boot.

To shut down Equalizer cleanly, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for global parameters (see "Logging In" on page 31).

2. Select *Equalizer* **> Maintenance > General**. Then select the **shutdown** button.

3. In the confirmation dialog box, click **shutdown** to confirm that you really want to shut down Equalizer (or click **Cancel** to abort the shutdown request). If you click **shutdown**, Equalizer immediately initiates the shutdown cycle. After waiting 30 seconds, you can safely power down the Equalizer.

Rebooting Equalizer

Rebooting Equalizer shuts it down cleanly and then restarts the system. To reboot the Equalizer:

1. Log into the Administrative Interface using a login that has **add/del** access for global parameters (see "Logging In" on page 31).

2. Select *Equalizer* **> Maintenance > General**. Then select the **reboot** button.

3. In the confirmation dialog box, click **reboot** to confirm that you really want to reboot Equalizer. A progress dialog is displayed while the system shuts down and reboots. Once the progress screen closes, refresh the browser display.

Creating a System Information Archive

You can create an archive that contains various configuration files, logs, and other information used by Coyote Point Support to help diagnose problems you are having with Equalizer. (In earlier releases, creating this archive was performed by logging into Equalizer and executing the **eqcollect** command.)

To create the system information archive:

1. Log into the Administrative Interface using a login that has **read** access for global parameters (see "Logging In" on page 31).

2. Select the *Equalizer* **> Maintenance > General** tab**.**

3. Select the **save state** button to create the archive. Once Equalizer collects the information for the archive, a dialog box is displayed by your browser to open or save the archive. Save the archive to a file on your local hard disk and note its location.

    The default archive name is *eqcollect.tgz*; we recommend you use a unique file name that includes the name of the system from which the archive was taken and the date, as in: *eqcollect_system-name_dd-mm-yy.tgz*. This ensures that you don't overwrite an existing archive, and helps identify the archive to Coyote Point Support.

4. Open your email client, and send the file you saved to **support@coyotepoint.com** as an attachment. Explain the nature of your problem in the email, or just include the support ticket number you were given previously by Coyote Point Support.

Configuring Static Routes

Static routes are commonly used to specify routes to IP addresses via gateways other than the default.

A default gateway is specified when you configure Equalizer via the **eqadmin** character based interface. If you need to access systems on a subnet that cannot be reached via this gateway, then you need to specify a **static route** to those systems through the gateway for that subnet.

Static routes on Equalizer are specified using the browser-based Administration Interface. Static routes can also be defined from the command line via the serial interface, but we recommend you use the browser interface exclusively to manage static routes on Equalizer. The interface manages changes to the */var/etc/rc.conf-eq* file for you, and updates Equalizer's routing tables (displayed using the **netstat -nr** shell command) as you add and delete them.

Adding a Static Route

1. Log into the Administrative Interface using a login that has **add/del** access for global parameters (see "Logging In" on page 31).

2. Select *Equalizer* **> Maintenance > Static Routes**:

Figure 29   The static routes screen

The table contains the following information for each configured static route on the system:

Name	An identifier for the route.
Type	Either **host** to specify a route to a host address, or **net** to specify an address for a subnet.
Network	The IP address for the host or subnet. Can be specified as a Classless Internet Domain Routing (CIDR) address to specify a netmask; for example: 192.168.1.0/24.
Gateway	The IP address of the gateway used to reach the host or subnet.

3. Click on the **Add** icon ![add icon]. The **Add New Route** screen appears:
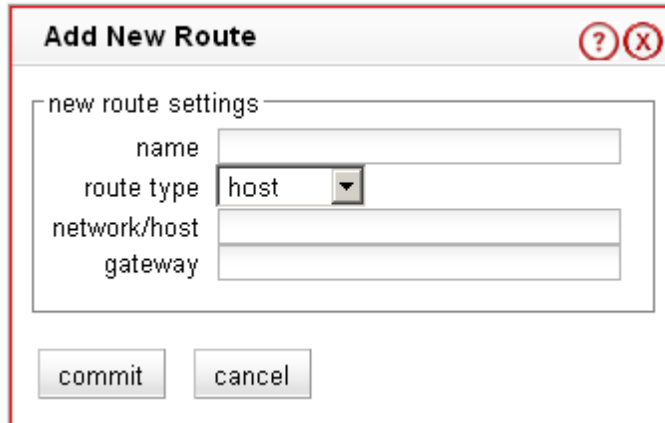
Figure 30   The add new route screen

4. Enter the parameters for the route, and select **commit**. You are returned to the **Static Routes** table, which now displays the route you added.

Modifying a Static Route

1. Log into the Administrative Interface using a login that has **write** access for global parameters (see "Logging In" on page 31).

2. Select *Equalizer* **> Maintenance > Static Routes**.

3. Highlight the route you want to change in the table and select the **Modify** icon ![modify icon]. The **Modify Route** screen is displayed:

4. Edit the values shown as needed and select **commit** to submit your changes. You are returned to the **Static Routes** screen, which now displays the updated route.

Deleting a Static Route

1. Log into the Administrative Interface using a login that has **write** access for global parameters (see "Logging In" on page 31).

2. Select *Equalizer* **> Maintenance > Static Routes**.

3. Highlight the route you want to delete in the table and select the **Delete** icon ![delete icon]. A confirmation screen appears.

4. Select **commit** to delete the route. You are returned to the **Static Routes** screen, from which the route has been removed.

Managing Gigabit Switch Ports

All Equalizer GX models have two Ethernet adapters on the motherboard. The switches on the front panel are connected to both adapters. This allows any switch port to be configured to work with either adapter.

In the default configuration, switch ports #1 & #2 are connected by VLAN to one of the motherboard interfaces (emulating the 'external interface' port provided on older 'si' hardware models). The remainder of the ports are configured on a separate VLAN connected to the other motherboard port (emulating the 'internal interface' ports provided on the 'si' hardware models).

Version 8.5 provides GUI-based user configuration for these "port based" VLANs, so that the user can select any set of ports as internal or external. Note that the #1 and #2 ports are identified with red labels to remind the user that there is something special about them (they are the external ports by default), as shown in the figure below.
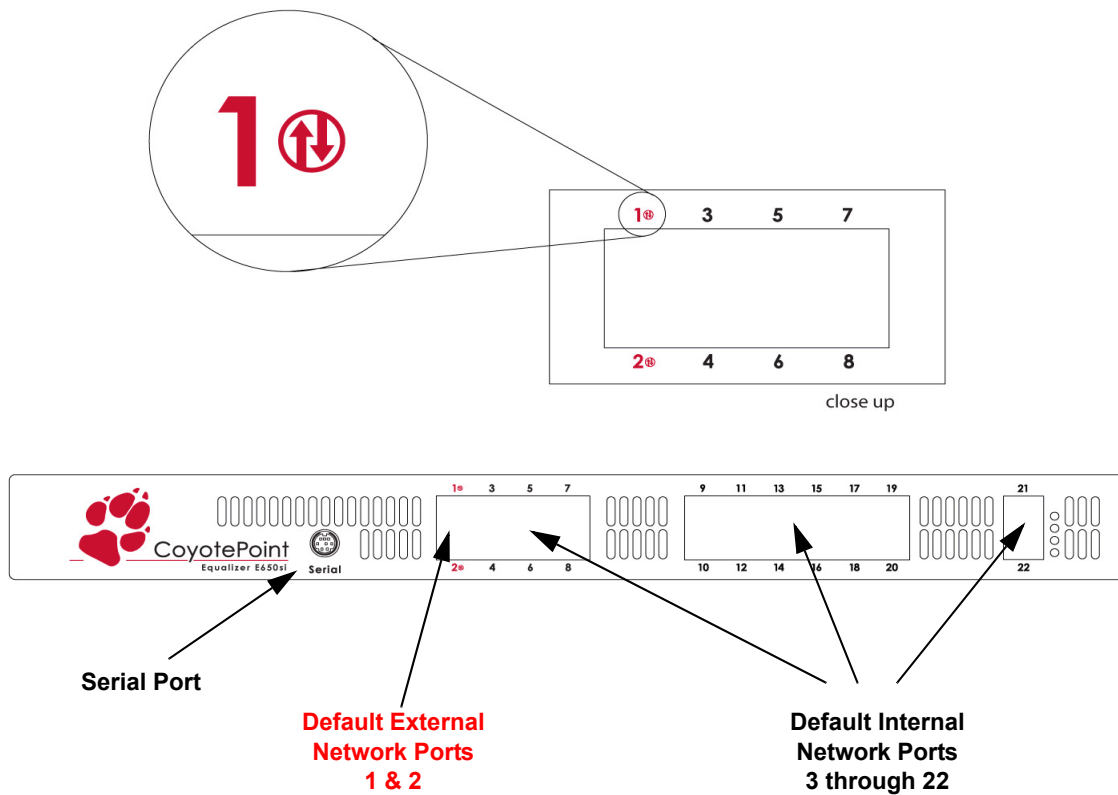
Figure 31 Equalizer 650GX Front Panel

Two VLANs are supported. In a dual network configuration, the main IP, cluster IPs and Envoy IP are on the external VLAN; the main IP and failover gateway alias are on the internal VLAN. In a sinfgle network configuration, the external VLAN is unused; ports currently assigned to the external VLAN can be reassigned to the internal VLAN, if desired.

Switch Administration Interface

The **Switch Status & Configuration** interface allows you to easily view and modify the configuration of each port on Equalizer's front panel. Click on *Equalizer* in the left frame, and then click **Status > Switch Status** in the right frame to display the **Switch Status** tab:
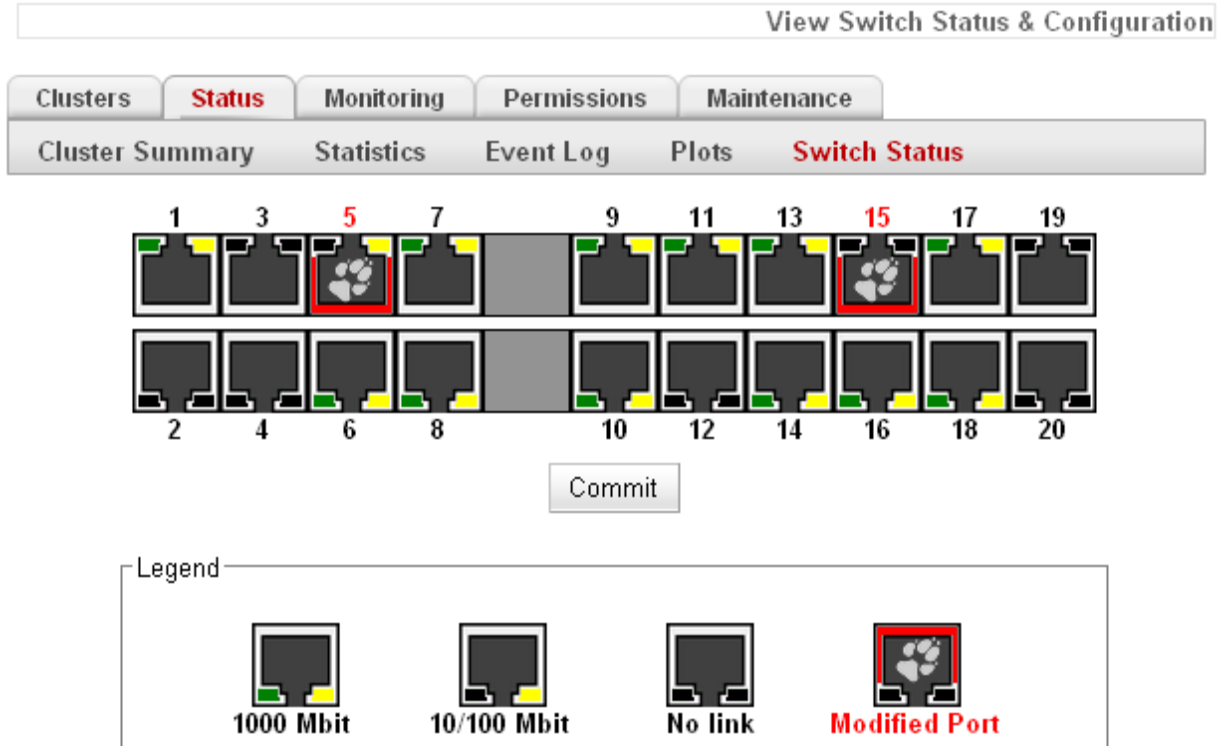
Viewing Link Status

The current link status of each port is displayed graphically, as shown in the Legend. Ports whose configuration have been modified but not committed (saved) are displayed with a red border and a coyote paw print.

Viewing Current Port Settings

To display the configuration settings for any port, mouse over the port to display a popup screen with the current parameters. Mousing over a modified port displays the modified (i.e., not yet committed) parameters.
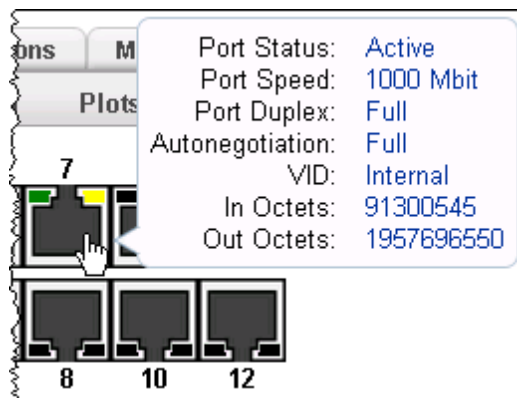
The port setting displayed in the mouseover popup are explained in the table below.

Port Status	Displays **Active** if the port has an active link, **No Link** if not.
Port Speed	If the port is **Active**, this is the current port speed. If there is **No Link**, this is the highest speed that can be negotiated, or the forced speed setting.
Port Duplex	If the port is **Active**, this is the current port duplex setting. If there is **No Link**, this is the highest duplex that can be negotiated, or the forced duplex setting.
Autonegotiation	Can be one of the following: **Full** (full autonegotiation at all support speeds and duplex settings), **Select** (autonegotiation at only selected speeds and duplex setting), and **Force Speed/Duplex** (set the port to a particular speed and duplex).
VID	The **VID** (VLAN ID) value indicates whether the port is configured on the External or Internal VLAN.
In Octets **Out Octets**	**In Octets** and **Out Octets** are the current count of bytes received and sent (respectively) through this switch port -- this includes *all* traffic, including broadcast traffic as well as traffic not destined for the Equalizer. Note that these counters do *not* indicate the number of bytes sent and received through the port since the last reboot. Each counter has a limit of 2^{32} - 1, and when this limit is reached, the counter is restarted.

Editing Port Settings

To edit the settings for any port, click on the port. A popup screen is displayed with the current parameters. Make changes to the port parameters and click **Apply**. Changes to port settings take effect immediately, but will not persist across system reboots until saved using the **commit** button on the **Switch Status** tab.

The parameters that appear on the **Configure switch port** popup depend on the current **Autonegotiation** setting of the port. When **Autonegotiation** is set to **Full** (the default setting), the popup looks like this:

With **Autonegotiation** set to **Full**, Equalizer will advertise its full range of speed and duplex settings on this port. The actual negotiated speed and duplex depend on the outcome of the autonegotiation process between the port and the device on the other end of the connection. This is the recommended setting for the majority of modern networking devices, and is the default autonegotiation setting for all switch ports.

The **VID** selection box allows you to specify the VLAN ID for this port. Choices are **Internal** (for the VLAN defined for the internal network), and **External** (for the external network VLAN).

When **Autonegotiation** is set to **Select**, the popup looks like this:

Setting **Autonegotiation** to **Select** allows you to set a specific speed and duplex to be negotiated with the device on the other end of the connection to this port. Only this speed and duplex combination is advertised for autonegotiation.

When **Autonegotiation** is set to **Force Speed/Duplex**, the popup looks like this:

The **Force Speed/Duplex** setting is similar to the **Select** setting in that only one speed and duplex combination is chosen, but with the Force setting there is no negotiation with the device on the other end of the connection.

Whether you choose **Full, Select,** or **Force Speed/Duplex** really depends on the operating characrteristics of the device on the other end of the connection. Some older network devices do not correctly autonegotiate, and Equalizer may (for example) be able to determine the correct speed setting, but cannot determine the duplex setting. For such devices, the Force setting is probably appropriate. Similarly, some devices set to autonegotiate may not work with a forced speed/duplex setting, even if that setting is correct for the device.

Commiting and Applying Switch Port Configuration Changes

As shown in the previous section, the **Configure switch port** popup has a **commit** button. When you click **commit** on the popup, the changes you have made are written to the switch configuration file. They will not, however, take effect on the port until you click **apply** on the **Switch Status** screen (or reboot the system).

Caution – Be sure that any changes you make to Equalizer's VLAN configuration are compatible with your current network configuration. It is possible to lock Equalizer out of the network if you mis-configure the switch interface VLANs. If you do misconfigure the switch and can no longer access Equalizer via the web interface, see the section "Resetting the Gigabit Switch VLAN IDs" on page 72.

Switch Interface Usage Scenarios

Some suggestions for using the new switch interface:

- In a single network configuration, the two external VLAN ports are unused. Ports #1 & 2 can be re-configured as part of the internal VLAN, adding two more server ports.

- Switch ports can be configured to match specific port settings required by the connected server. For example, you could use the switch interface to configure a particular switch port to be 100Mb/s and half-duplex to accommodate older hardware.

- The graphic switch port display provides dynamic switch status feedback. The lights for each port, for example, indicate the current status of the port without a browser refresh. This is a handy way for checking port status when you are accessing the system remotely.

Resetting the Gigabit Switch VLAN IDs

A new **eqadmin** console utility option allows you to reset the VLAN IDs associated with each port on the front-panel Gigabit Switch to their default values. This is provided in the unlikely event that misconfiguration of the switch interface disables access to the web-based Administration Interface and the graphical switch management interface.

In the default configuration of the Gigabit Switch, ports 1 and 2 are associated with one network interface card on Equalizer (the 'external interface' provided in earlier releases), while the remainder of the ports (starting at port 3) are associated with the other network interface card (the 'internal interface' provided in earlier releases).

Resetting the Gigabit Switch via **eqadmin** returns the Gigabit Switch ports to the default configuration described above. To reset the Gigabit Switch:

1. Log into Equalizer via the serial port or **ssh** (if enabled). As *root*, enter `eqadmin` and press **Enter** to display the **Equalizer Configuration Menu**:

2. Press '**0**' or use the arrow keys to scroll down to the last option on the screen, **Reset Switch**, and press **Enter**.

3. A confirmation screen is displayed. Select **Yes** and press **Enter** to reset the switch. The change takes effect immediately.

Switch Interface Notes for Older Equalizer Users

The switch management interface is primarily intended for the managed gigabit switches included in 'GX' model Equalizers (as described above), and two earlier models: the E550si and E650si. If Version 8.5 or later is installed on an older model E550si or E650si Equalizer, the switch management interface is also available on these models with reduced functionality:

1. The **Ext** port on the front panel of the 'si' unit is *not* managed via the Switch management interface.

2. The **speed**, **duplex**, and **autonegotiation** settings for all internal ports (ports numbered starting at "1") can be set via the **Switch Status** tab, but VLANs are not supported (the VID cannot be set).

Note – Tthe **Switch Status** tab is unavailable (grayed-out) on E250, E350si, E450si, and Extreme model Equalizers.

Chapter 5:

Administering Virtual Clusters

Working with Virtual Clusters

A virtual cluster is a collection of servers with a single network visible IP address. All client requests come into Equalizer through a cluster IP address, and are routed by Equalizer to the appropriate server in the cluster, according to the load balancing options set on the cluster. The figure below shows a conceptual diagram of an Equalizer with three clusters.
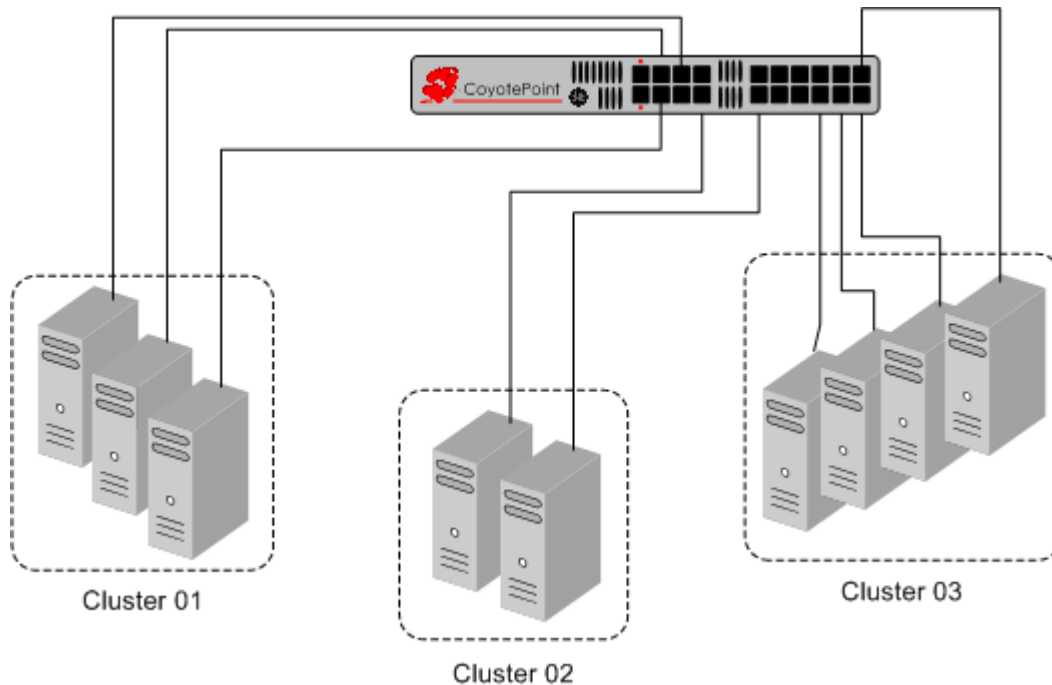
Figure 32   An Equalizer with three defined clusters

The parameters you specify when setting up a virtual cluster determine how the Equalizer manages connections between the Equalizer and the servers in a cluster, and how incoming requests are routed through the Equalizer to the cluster. Before beginning to define a cluster, we recommend you read this chapter in its entirety so that you can:

1. Determine the **IP addresses** to use for each cluster, and for every server in each cluster.

2. Determine the **protocol** (Layer 4 TCP, Layer 4 UDP, Layer 7 HTTP, Layer 7 HTTPS) that will be used to communicate between the Equalizer and the servers in each cluster:

    *   *In Layer 4 TCP and UDP clusters,* Equalizer routes requests based on configured load balancing criteria, the IP address, and the TCP or UDP port number. Load balancing decisions do not take into account the content of the request.

        Any TCP-based protocol (HTTP, HTTPS, FTP, etc.) can be load balanced by an L4 TCP cluster.

        L4 UDP clusters are appropriate for connectionless (stateless) applications, such as DNS, TFTP, Voice over IP (VoIP), and streaming applications -- any application that exchanges short packets with many clients, and where dropped packets are preferred to delayed packets (i.e., the highest possible network performance is required).

    *   *In Layer 7 HTTP and HTTPS clusters,* Equalizer routes requests to particular servers based on configured load balancing criteria, the IP address, the port, *and the content of the request*. Because Equalizer examines the request content, load balancing decisions can be made based on application specific criteria through the use of Match Rules.

        Also note that in HTTPS clusters, Equalizer accepts HTTPS connections from clients, performs all the SSL operations necessary to examine the request, and sends the request on to a server in the cluster using HTTP.

This offloads resource-intensive SSL operations from the server to Equalizer, improving overall server and cluster performance.

3. Determine the **load balancing policy** (**round robin**, **static weight**, **adaptive**, **fastest response**, **least connections**, or **server agent**) that the Equalizer will use to decide how to route incoming requests to the servers in the cluster.

4. Determine the additional settings and flags to be used on the cluster and its servers. For most options, start with the defaults and make incremental changes as you examine traffic passing through your clusters.

Adding a Layer 7 Virtual Cluster

To add a new virtual cluster, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for global parameters (see "Logging In" on page 31).

2. Right click on **Equalizer** (or the configure *Failover Peer Name* for this Equalizer) at the top of the left frame, and select **Add Cluster** from the menu that appears. The **Add New Cluster** dialog appears.

3. Select **Layer 7 HTTP** or **Layer 7 HTTPS** and then click the **Next** icon > .

4. Enter the following information:

Cluster Name	The logical name for the cluster, or accept Equalizer's default. Each cluster must have a unique name that begins with an alphabetical character (for example, *CPImages1*). The cluster name is limited to 63 characters.
Cluster IP Address	Enter the **ip address**, which is the dotted decimal IP address of the cluster. The IP address of the cluster is the external address (for example, `199.146.85.0`) with which clients connect to the cluster.
Cluster Port	For HTTP and HTTPS protocol clusters, enter the **port:** the numeric port number on the Equalizer to be used for traffic between the clients and the cluster. For HTTP clusters, the cluster port defaults to 80. For HTTPS clusters, the cluster port defaults to 443.

Click the **Next** icon > .

5. A confirmation screen appears; click commit to create the cluster with the parameters shown.

6. The **Configuration** tab for the new cluster is opened. See the following section for an explanation of the Layer 7 cluster configuration tabs and parameters.

Modifying a Layer 7 Virtual Cluster

The configuration tabs for a cluster are displayed automatically when a cluster is added to the system, or by selecting the cluster name from the left frame Configuration Tree. HTTP and HTTPS clusters parameters are divided among the following tabs:

- **Layer 7 Required Tab**
- **Layer 7 Probes Tab**
- **Layer 7 Persistence Tab**
- **Layer 7 Networking Tab**
- **Layer 7 Security > Certificates Tab (HTTPS only)**
- **Layer 7 Security > SSL Tab (HTTPS only)**

These are described in the following sections. To update the settings on any tab, make changes and select the **commit** button to save them.

Layer 7 Required Tab

ip	Enter the **ip address**, which is the dotted decimal IP address of the cluster. The IP address of the cluster is the external address (for example, `199.146.85.0`) with which clients connect to the cluster.
port	For HTTP and HTTPS protocol clusters, enter the **port:** the numeric port number on the Equalizer to be used for traffic between the clients and the cluster. For HTTP clusters, the port defaults to 80. For HTTPS clusters, the port defaults to 443. This port also becomes the default port for servers added to the cluster (though servers can use a different port number than the one used by the cluster).
netmask	Used to define an IP subnet that is different from the subnet defined for the cluster IP interface. If this is defined, it is assumed that the customer has the proper routing in place for clients to access multiple IP subnets defined on the Equalizer. The default is the netmask of the network interface for the cluster IP.
disable	Disable this cluster. The cluster IP address will not accept requests when this flag is enabled.
ignore case	**ignore case** causes all of the cluster's match rules to use case insensitive comparisons when this box is checked. You can override this setting by changing **ignore case** for a specific match rule.
spoof	**spoof** causes Equalizer to spoof the client IP address when Equalizer routes a request to a server in a virtual cluster; that is, the IP address of the *client* is sent to the server, not the IP address of the Equalizer. This option is on by default. If you disable this option, the server receiving the request will see the Equalizer's address as the client address because the TCP connection to the client is terminated when the request is routed. When spoof is enabled, the servers in the cluster must use the Equalizer as the default gateway for routing.
compress	If Express Hardware GZIP Compression is installed, the **compress** flag appears in the HTTP and HTTPS cluster configuration screens. When the **compress** cluster flag is enabled, Equalizer automatically detects requests to the cluster from compression-capable browser clients and performs GZIP compression on all cluster responses sent to that client. This effectively enables compression for all clients using recent browser versions. Also see "Layer 7 Networking Tab" on page 79.

Layer 7 Probes Tab

probe port	The port on the Equalizer to be used to for all TCP and ACV server health check probes for this cluster. The port specified here becomes the default probe port used when a new server is added to the cluster. By default, the **probe port** field is set to zero and the Equalizer uses the Layer 7 **port** field value for the probe port when a new server is created. To change the default behavior, set **probe port** to a specific port number. A **probe port** value can also be set on individual servers as well; see Adding a Server to a Cluster. (Note that the **server agent port** remains a separate cluster port that is used only for server agent communication; see below.)
ACV probe	The active content verification (ACV) probe string. This string is sent to the server as part of a TCP probe; the server must respond with the **ACV response** string or it is marked down. The string is limited to 99 characters, and must use only the printable ASCII characters (decimal 32 to 126). For more information, refer to "Using Active Content Verification (ACV)" on page 93.
ACV response	The active content verification response string. This string must appear in the first 1024 characters of the server's response to the **ACV probe**, or the server is marked down. For more information, refer to "Using Active Content Verification (ACV)" on page 93.
probe delay	The minimum number of seconds between TCP and ACV probes of the cluster's servers. Also see the global parameters **probe interval**, **probe timeout**, **probe delay**, and **strikeout threshold** under "Modifying Global Parameters" on page 47.
server agent port	The port used to contact server agents. The default port is **1510**. See Appendix A, "Server Agent Probes" on page 205 for more information.
agent probe	An optional string that is sent to an agent when an agent probe occurs. See Appendix A, "Server Agent Probes" on page 205 for more information.
agent type	**server agent --** Equalizer uses a server agent to gather performance statistics from the servers in the cluster. If you enable this option, you must run Server Agent daemons on each server in the cluster and must specify a value in **server agent port**. See Appendix A, "Server Agent Probes" on page 205 for more information about configuring server agents. **VLB** -- Equalizer uses the VMware Infrastructure Management API to retrieve real-time virtual server performance information from a VMware vCenter (Virtual Center) console or from a single ESX Server. Before selecting this option, see Appendix F, "Equalizer VLB" on page 241. **none** -- No server agent is used.

Layer 7 Persistence Tab

Please see "Enabling Persistent Server Connections" on page 88 for a discussion of server persistence on Equalizer.

cookie age	**cookie age** sets the time, in seconds, over which the client browser maintains the cookie (0 means the cookie never expires). After the specified number of seconds have elapsed, the browser can delete the cookie and any subsequent client requests will be handled by Equalizer's load-balancing algorithms.

cookie scheme	Specifies the format of the cookie to be used for the cluster as an integer between 0 and 2 (default is 2) **0** Constructs a cookie which will be named in such a way that so as long as the cluster maintains the same IP address, servers can be added to and removed from the cluster without invalidating all of the existing cookies. **This cookie stores the cluster IP and port, and the server IP and port.** **1** Constructs a cookie which will be valid across all clusters with the same IP address (not port specific). A requirement for this to be useful is that all clusters on that IP address share the same set of servers. **This cookie stores the Cluster IP, and Server IP and port.** **2** Constructs a cookie which will be valid across all clusters with the same IP address (using any port), and the same server within those clusters (with the server using any port). A requirement for this to be useful is that all clusters on that IP address share the same set of servers. **This cookie encodes the Cluster IP and Server IP.**
cookie generation	A value added to cookies when the cookie scheme is 2. In order for cookies to be valid, cookie generation must match the equivalent number embedded in the cookie. Conversely if you need to invalidate old cookies, increment this number.
cookie domain	Limits the presented cookie only to servers whose host name is within the specified domain. For example, if the cookie domain is `coyotepoint.com`, the browser will only present the cookie to servers in the `coyotepoint.com` domain (for example, `www.coyotepoint.com` or `my.coyotepoint.com`).
cookie path	Presents the cookie only when the path component of the request URI has the same prefix as that of the specified path. For example, if the cookie path is `/store/`, the browser presents the cookie only if the request URI includes a path such as `/store/mypage.html`.
persist	Equalizer uses cookies to maintain a persistent session between a client and a particular server. This option is on by default. Equalizer "stuffs" a cookie into the server's response header on its way back to the client. This cookie uniquely identifies the server to which the client was just connected. With **persist** enabled, Equalizer routes only the first request from a client using load balancing criteria; subsequent client requests are routed to the same selected server for the entire session (while the cookie is valid -- see **cookie age**, above).
always	By default, Equalizer inserts a persistence cookie into a server response only if it finds a cookie from the server in the response. If **always** and **persist** are enabled, Equalizer includes a cookie in the response regardless of whether the server sent a cookie.

LB Algorithm Tab

On this tab, choose a load balancing **policy** and **responsiveness** for the cluster:

policy	For all cluster protocols, choose the appropriate load-balancing **policy** to be used by this cluster. Choose from **round robin** (default), **static weight**, **adaptive**, **fastest response**, **least connections**, **server agent**, and **custom**. For more information, refer to "Configuring a Cluster's Load-Balancing Options" on page 86.
responsiveness	**responsiveness** sets the load-balancing response setting for this cluster. For more information, refer to "Configuring a Cluster's Load-Balancing Options" on page 86.

The parameter slider bars are static for all policies other than **custom**, and their positions indicate the relative weight that the selected policy uses for each parameter. If custom is selected, you can adjust the following load balancing policy parameters::

delay weight	The relative influence on the policy of the current response time between Equalizer and the server.
active connections weight	The relative influence on the policy of the number of active connections currently open to a server
agent weight	The relative influence on the policy of the return value of a server agent (if any) running on the servers in the cluster.

Layer 7 Networking Tab

The parameters in the **Networking** tab affect:

- the amount of memory Equalizer allocates for data buffers and HTTP headers
- the connections between clients and Equalizer
- the connections between Equalizer and the servers in virtual clusters

send buffer	The amount of memory in kilobytes reserved by each Layer 7 proxy process to store outgoing data before it is placed on the network interface. **Default: 32. Minimum: 4. Maximum: 128.** This global parameter applies to Layer 7 HTTP and HTTPS clusters only, and can also be set per cluster.
receive buffer	The amount of memory in kilobytes reserved by each Layer 7 proxy process to store data that has been received on a network interface before it is processed. **Default: 16. Minimum: 4. Maximum: 128.** This global parameter applies to Layer 7 HTTP and HTTPS clusters only, and can also be set per cluster.
request max	The maximum amount of memory in kilobytes reserved for HTTP request headers. **Default: 32. Minimum: 4. Maximum: 64.** This global parameter applies to Layer 7 HTTP and HTTPS clusters only.
response max	The maximum amount of memory in kilobytes reserved for HTTP response headers. **Default: 32. Minimum: 4. Maximum: 64.** This global parameter applies to Layer 7 HTTP and HTTPS clusters only.
custom header	A custom HTTP header that Equalizer inserts into all client requests before they are sent to the server. The format of the string is ***text:text***. Also see the **insert client IP** flag, below, and the section "Specifying a Custom Header for HTTPS Clusters" on page 95.
compress minimum	The minimum file size in bytes required for GZIP compression, if enabled (see the **compress** flag under "Layer 7 Required Tab" on page 76). Files smaller than the minimum specified are not compressed. **Default:1024** bytes.

compress mime-types	Specifies the *mime-types* that will be compressed when the **compress** flag is enabled for the cluster (see "Layer 7 Required Tab" on page 76). The value of this parameter is a string (maximum length: 512 characters) with valid mime-type names separated by a colon (:). The default **compress mime-types** string specifies the following mime-types:
	
	`text/*`
	`application/msword`
	`application/postscript`
	`application/rtf`
	`application/x-csh`
	`application/x-javascript`
	`application/x-sh`
	`application/x-shar`
	`application/x-tar`
	`application/x-tcl`
	`application/xslt+xml`
	`audio/midi`
	`audio/32kadpcm`
	`audio/x-wav`
	`image/bmp`
	`image/tiff`
	`image/x-rgb`
	
	Lists of officially supported mime-types can be found at:
	`http://www.iana.org/assignments/media-types/`

connect timeout	The time in seconds that Equalizer waits for a server to respond to a connection request. The default is the global value. See "HTTP and HTTPS Connection Timeouts" on page 210.

client timeout	The time in seconds that Equalizer waits before closing an idle client connection. The default is the global value. See "HTTP and HTTPS Connection Timeouts" on page 210.
server timeout	The time in seconds that Equalizer waits before closing an idle server connection. The default is the global value. See "HTTP and HTTPS Connection Timeouts" on page 210.
abort server	By default, when a client closes a connection, Equalizer waits for a response from the server before closing the server connection. If this flag is enabled, Equalizer will not wait for a response before closing the connection to the server; instead it sends a TCP RST (reset) to the server when the client closes the connection.
once only	Limits Equalizer to parsing headers (and executing match rules) for only the first request of any client making multiple requests across a single TCP connection. This option is off by default: meaning that Equalizer will parse the headers of every client request. See "Enabling the Once Only and Persist Options" on page 90.
insert client IP	When this flag is enabled, Equalizer inserts an **X-forwarded-for:** header with the client's IP address into all client requests before they are sent to the server. This flag is *disabled* by default for HTTP clusters and *enabled* by default for HTTPS clusters.

Layer 7 Security > Certificates Tab (HTTPS only)

Use the **Security > Certificates** tab to:

- upload an SSL certificate that clients will use to validate a connection to an HTTPS cluster (a **cluster** certificate)

- upload an SSL certificate for Equalizer to use to validate clients that request connections to HTTPS clusters (a **client** certificate)

See "Using Certificates in HTTPS Clusters" on page 228 for more information.

Layer 7 Security > SSL Tab (HTTPS only)

The **Security > SSL** tab allows you to configure various options that are specific to HTTPS connections.

cipher suite	Lists the supported cipher suites for incoming HTTPS requests. If a client request comes into Equalizer that does not use a cipher in this list, the connection is refused. Please see "Configuring Cipher Suites" on page 239.
session cache timeout	The number of seconds that Equalizer waits before disposing of an SSL session cache entry.
session cache kbytes	The maximum amount of memory in kilobytes allotted to an SSL session cache.
client verification depth	The depth to which certificate checking is done on the client certificate chain. The default of 2 indicates that the client certificate (level 0) and two levels above it (levels 1 and 2) are checked; any certificates above level 2 in the chain are ignored. You should only need to increase this value if the Certificate Authority that issued your certificate provided you with more than 2 chained certificates in addition to your client certificate. See Appendix E, "Using Certificates in HTTPS Clusters" on page 227.

certify client	Indicates whether the server asks the client for a client certificate when a client request is received. The connection will succeed even if the client does not provide a certificate; but, if one is provided by the client it will be validated. See Appendix E, "Using Certificates in HTTPS Clusters" on page 227.
require certificate	Indicates whether the server requires a client certificate when a client request is received. If the client does not provide a certificate, the connection is refused. See Appendix E, "Using Certificates in HTTPS Clusters" on page 227.
verify once	Indicates that the server will verify certificates only on the first client request, even if SSL is renegotiated. See Appendix E, "Using Certificates in HTTPS Clusters" on page 227.
ssl unclean shutdown	Should be enabled if you cannot access pages while trying to maintain HTTPS persistent connections over HTTP/1.1. This problem especially applies to connections between Internet Explorer and Apache Servers and usually occurs intermittently.
no header rewrite	When enabled, forces Equalizer to pass responses from an HTTPS cluster's servers without rewriting them. In the typical Equalizer setup, you configure servers in an HTTPS cluster to listen and respond using HTTP; Equalizer communicates with the clients using SSL. If a server sends an HTTP redirect using the Location: header, this URL most likely will not include the `https:` protocol. Equalizer rewrites responses from the server so that they are HTTPS. You can direct Equalizer to pass responses from the server without rewriting them by enabling the **no header rewrite** flag.

Adding a Layer 4 Virtual Cluster

To add a new Layer 4 virtual cluster, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for global parameters (see "Logging In" on page 31).

2. Right click on **Equalizer** (or the configure *Failover Peer Name* for this Equalizer) at the top of the left frame, and select **Add Cluster** from the menu that appears. The **Add New Cluster** dialog appears.

3. Select **Layer 4 TCP** or **Layer 4 UDP** and then click the **Next** icon ⟩ .

4. Enter the following information:

Cluster Name	The logical name for the cluster, or accept Equalizer's default. Each cluster must have a unique name that begins with an alphabetical character (for example, *CPImages)*.
Cluster IP Address	Enter the **ip address**, which is the dotted decimal IP address of the cluster. The IP address of the cluster is the external address (for example, `199.146.85.0`) with which clients connect to the cluster.

Start Port **End Port**	For L4 UDP and L4 TCP protocol clusters, a *port range* can be defined using the **start port** and **end port** fields. These are the ports on the Equalizer to be used to send traffic to the servers in the cluster. Port ranges allow Equalizer users to create a single cluster to control the traffic for multiple, contiguous ports. There are two typical uses for port ranges: Specific applications that require a range of ports. The need to open up access to servers behind the Equalizer for all ports. Enter the first port number in the **start port** field (which is required). Enter the end port number in the **end port** field. (If **end port** is not visible, check the **advanced** flag.) When the **end port** field is left with a value of zero (the default), Equalizer disables the port range feature and uses the **start port** as the server port. The **start port** cannot be higher than **end port** when **end port** is nonzero. The **port** defined for a *server* in the cluster for which a port range is defined indicates the port on the server that starts the range of ports to be opened. See Step on page 101, under Adding a Server to a Cluster

Click the **Next** icon ⟩ .

5. A confirmation screen appears; click commit to create the cluster with the parameters shown.

6. The **Configuration** tab for the new cluster is opened. See the following section for an explanation of the Layer 4 cluster configuration tabs and parameters.

Modifying a Layer 4 Virtual Cluster

The configuration tabs for a cluster are displayed automatically when a cluster is added to the system, or by selecting the cluster name from the left frame Configuration Tree. TCP and UDP cluster parameters are divided among the following tabs:

- **Layer 4 Required Tab**
- **Layer 4 Probes Tab**
- **Layer 4 Persistence Tab**

These are described in the following sections. To update the settings on any tab, make changes and select the **commit** button to save them.

Layer 4 Required Tab

ip	Enter the **ip address**, which is the dotted decimal IP address of the cluster. The IP address of the cluster is the external address (for example, `199.146.85.0`) with which clients connect to the cluster.

port	For L4 UDP and L4 TCP protocol clusters, a *port range* can be defined using the **start port** and **end port** fields. These are the ports on the Equalizer to be used to send traffic to the servers in the cluster. Port ranges allow Equalizer users to create a single cluster to control the traffic for multiple, contiguous ports. There are two typical uses for port ranges: 1 Specific applications that require a range of ports. 2 The need to open up access to servers behind the Equalizer for all ports. Enter the first port number in the **start port** field (which is required). Enter the end port number in the **end port** field. When the **end port** field is left with a value of zero (the default), Equalizer disables the port range feature and uses the **start port** as the server port. The **start port** cannot be higher than **end port** when **end port** is nonzero. The **port** defined for a *server* in the cluster for which a port range is defined indicates the port on the server that starts the range of ports to be opened. See "Adding a Server to a Cluster" on page 101.
policy	For all cluster protocols, choose the appropriate load-balancing **policy** to be used by this cluster. Choose from **round robin** (default), **static weight**, **adaptive**, **fastest response**, **least connections**, or **server agent**. For more information, refer to "Configuring a Cluster's Load-Balancing Options" on page 86.
responsiveness	**responsiveness** sets the load-balancing response setting for this cluster. For more information, refer to "Configuring a Cluster's Load-Balancing Options" on page 86.
disable	Disable this cluster. The cluster ip will not accept requests when this flag is enabled.
spoof	**spoof** causes Equalizer to spoof the client IP address when Equalizer routes a request to a server in a virtual cluster; that is, the IP address of the *client* is sent to the server, not the IP address of the Equalizer. This option is on by default. If you disable this option, the server receiving the request will see the Equalizer's address as the client address because the TCP connection to the client is terminated when the request is routed. When spoof is enabled, the servers in the cluster must use the Equalizer as the default gateway for routing.
direct server return	When enabled, Equalizer forwards packets to the server in such a way that the server responds directly to the client, rather than through Equalizer. This option requires special configuration on the servers in the cluster; see "Configuring Direct Server Return" on page 128 before enabling this option. The **spoof** option must also be enabled when this option is enabled.

Layer 4 Probes Tab

probe port	The port on the Equalizer to be used to for all TCP and ACV server health check probes for this cluster. The port specified here becomes the default probe port used when a new server is added to the cluster. By default, the **probe port** field is set to zero and the Equalizer uses the Layer 4 **start port** field value for the probe port when a new server is created. To change the default behavior, set **probe port** to a specific port number. A **probe port** value can also be set on individual servers as well; see Adding a Server to a Cluster. (Note that the **server agent port** remains a separate cluster port that is used only for server agent communication; see below.)
ACV probe	The active content verification probe string. For more information, refer to "Using Active Content Verification (ACV)" on page 93. **Note:** Not supported for UDP clusters.
ACV response	The active content verification response string. For more information, refer to "Using Active Content Verification (ACV)" on page 93. **Note:** Not supported for UDP clusters.
probe delay	The minimum number of seconds between TCP and ACV probes of the cluster's servers. Also see the global parameters **probe interval**, **probe timeout**, **probe delay**, and **strikeout threshold** under "Modifying Global Parameters" on page 47.
server agent port	The port used to contact server agents. The default port is 1510. See Appendix A, "Server Agent Probes" on page 205 for more information.
agent probe	An optional string that is sent to an agent when an agent probe occurs. See Appendix A, "Server Agent Probes" on page 205 for more information.
probe ssl	Equalizer uses SSL when it sends the **ACV probe** string. For more information, refer to "Using Active Content Verification (ACV)" on page 93. **Note:** Not supported for UDP clusters.
agent type	**server agent --** Equalizer uses a server agent to gather performance statistics from the servers in the cluster. If you enable this option, you must run Server Agent daemons on each server in the cluster and must specify a value in **server agent port**. See Appendix A, "Server Agent Probes" on page 205 for more information about configuring server agents. **VLB** -- Equalizer uses the VMware Infrastructure Management API to retrieve real-time virtual server performance information from a VMware vCenter (Virtual Center) console or from a single ESX Server. Before selecting this option, see Appendix F, "Equalizer VLB" on page 241. **none** -- No server agent is used.

Layer 4 Persistence Tab

sticky time	**sticky time** is the number of seconds that Equalizer should "remember" connections from clients. Valid values are from 0 (which disables sticky connections) to 1073741823 seconds (or over 34 years). For more information, refer to "Enabling Sticky Connections" on page 88.

inter-cluster sticky	**inter-cluster sticky** is a Layer 4 option that when enabled ensures that Equalizer attempts to direct requests from a particular client to the same server on another available port if the intended server port is unreachable. The Layer 4 clusters must have the same IP address, different ports, and a non-zero sticky time. For more information, refer to "Enabling Sticky Connections" on page 88.

Deleting a Virtual Cluster

Deleting a cluster with servers assigned to it also deletes the server definitions as well. To delete a cluster, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for global parameters (see "Logging In" on page 31).

2. In the left frame, right-click on the name of the cluster to be deleted and select **Delete Cluster** from the menu.

3. When prompted, click **delete** to confirm that you want to remove the cluster.

Configuring a Cluster's Load-Balancing Options

Configure load balancing policy and response settings for each cluster independently. Multiple clusters do not need to use the same load balancing configuration even if the same physical server machines host them. For example, if one cluster on port 80 handles HTML traffic and one on port 8000 serves images, you can configure different load balancing policies for each cluster.

When you use adaptive load balancing (that is, you have **not** set the cluster's load balancing policy to round robin or static weight), you can adjust Equalizer to optimize cluster performance. For more information, see "Adjusting a Server's Initial Weight" on page 105.

Equalizer's Load Balancing Policies

Equalizer supports the following load balancing policies, each of which is associated with a particular algorithm that Equalizer uses to determine how to distribute requests among the servers in the cluster:

- **round robin** load balancing distributes requests equally among all the servers in the cluster. Equalizer dispatches the first incoming request to the first server, the second to the second server, and so on. When Equalizer reaches the last server, it repeats the cycle. If a server in the cluster is down, Equalizer does not send requests to that server. This is the default method.

  The round robin method does not support Equalizer's adaptive load balancing feature; so, Equalizer ignores the servers' initial weights and does not attempt to dynamically adjust server weights based on server performance.

- **static weight** load balancing distributes requests among the servers depending on their assigned initial weights. A server with a higher initial weight gets a higher percentage of the incoming requests. Think of this method as a *weighted round robin* implementation. Static weight load balancing does not support Equalizer's adaptive load balancing feature; Equalizer does not dynamically adjust server weights based on server performance.

- **adaptive** load balancing distributes the load according to the following performance indicators for each server.

  - **Server response time** is the length of time for the server to begin sending reply packets after Equalizer sends a request.

  - **Active connection count** shows the number of connections currently active on the server.

- • **Server agent value** is the value returned by the server agent daemon (if any) running on the server.

- • **fastest response** load balancing dispatches the highest percentage of requests to the server with the shortest response time. Equalizer does this carefully: if Equalizer sends too many requests to a server, the result can be an overloaded server with slower response time. The fastest response policy optimizes the cluster-wide response time. The fastest response policy also checks the number of active connections and server agent values (if configured); but both of these have less of an influence than they do under the adaptive load balancing policy. For example, if a server's active connection count and server agent values are high, Equalizer might not dispatch new requests to that server even if that server's response time is the fastest in the cluster.

- • **least connections** load balancing dispatches the highest percentage of requests to the server with the least number of active connections. In the same way as Fastest Response, Equalizer tries to avoid overloading the server so it checks the server's response time and server agent value. Least Connections optimizes the balance of connections to servers in the cluster.

- • **server agent** load balancing dispatches the highest percentage of requests to the server with the lowest server agent value. In a similar way to Fastest Response, Equalizer tries to avoid overloading the server by checking the number of connections and response time. This method only works if server agents are running on all servers in the cluster. For more information about server agents, see "Configuring a Cluster to Use Server Agents" on page 88.

Equalizer's Load Balancing Response Settings

The **responsiveness** setting controls how aggressively Equalizer adjusts the servers' dynamic weights. Equalizer provides five response settings: Slowest, Slow, Medium, Fast, and Fastest. The response setting affects the dynamic weight spread, weight spread coefficient, and optimization threshold that Equalizer uses when it performs adaptive load balancing:

- • **Dynamic Weight Spread** indicates how far a server's dynamic weight can vary (or *spread*) from its initial weight.

- • **Weight Spread Coefficient** regulates the speed of change to a server's dynamic weight. The weight spread coefficient causes dynamic weight changes to happen more slowly as the difference between the dynamic weight and the initial weight increases.

- • **Optimization Threshold** controls how frequently Equalizer adjusts dynamic weights. If Equalizer adjusts server weights too aggressively, oscillations in server weights can occur and cluster-wide performance can suffer. On the other hand, if Equalizer does not adjust weights often enough, server overloads might not be compensated for quickly enough and cluster-wide performance can suffer.

Aggressive Load Balancing

After you fine-tune the initial weights of each server in the cluster, you might discover that Equalizer is not adjusting the dynamic weights of the servers at all: the dynamic weights are very stable, even under a heavy load. In this case, you might want to set the cluster's load balancing response parameter to *fast*. Then Equalizer tries to optimize the performance of your servers more aggressively; this should improve the overall cluster performance. For more information about setting server weights, see "Adjusting a Server's Initial Weight" on page 105.

Dynamic Weight Oscillations

If you notice a particular server's dynamic weight oscillates (for example, the dynamic weight varies from far below 100 to far above 100 and back again), you might benefit by choosing *slow* response for the cluster. You should also investigate the reason for this behavior; it is possible that the server application is behaving erratically.

Configuring a Cluster to Use Server Agents

A *server agent* collects performance statistics from a server. If you configure a cluster to use server agents, Equalizer periodically contacts the server agent daemon running on each server and downloads the server performance statistics. You can also customize server agents to report on server resource availability; then Equalizer can stop sending requests to a server if a database or other vital resource is unavailable.

> **Note –** When you configure a cluster to use server agents, each server in the cluster ***must*** run a server agent daemon, so that the agent can provide status information to the Equalizer. If no agent is running on a server in a cluster configured to use the server agent load balancing policy, then the Equalizer will load balance without using the agent return value for that server (unless **require agent response** is set for the cluster, in which case Equalizer regards that server as down).

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster (see "Logging In" on page 31).

2. In the left frame, click the name of the cluster to be configured. The cluster's parameters appear in the right frame.

3. Select the **Probes** tab in the right frame.

4. Check the **server agent** checkbox.

5. In the **server agent port** field, specify the port used to contact the server agent; the default port is 1510.

6. If your agent needs to have a string sent to it before it will respond, provide the string to be sent to the agent in the **agent probe** field.

7. Click the **commit** button.

For information about writing your own server agents and using agents to monitor server resource availability, see "Server Agent Probes" on page 205.

Enabling Persistent Server Connections

Equalizer provides several methods by which connections between clients and servers can be made *persistent*; that is, it is possible to route a series of requests from a particular client to the same server, rather than have the Equalizer load balance each request in the series -- potentially sending each request to a different server.

For Layer 4 clusters, persistent server connections are enabled using the **sticky time** cluster parameter and (optionally) the **inter-cluster sticky** cluster flag. See "Enabling Sticky Connections" on page 88.

For Layer 7 clusters, persistent server connections are enabled using the **persist** and **always** cluster flags. See "Enabling Cookies for Persistent Connections" on page 89.

Enabling Sticky Connections

For Layer 4 TCP and UDP clusters, you can use IP-address based sticky connections to maintain persistent sessions.

The **sticky time** period is the length of time over which Equalizer ensures that it directs new connections from a particular client to the same server. The timer for the sticky time period begins to expire as soon as there are no active connections between the client and the cluster. If Equalizer establishes a new connection to the cluster, Equalizer resets the timer for the sticky time period.

When you enable sticky connections, the memory and CPU overhead for a connection increase. This overhead increases as the sticky period increases. You should use the shortest reasonable period for your application and avoid enabling sticky connections for applications unless they need it. For most clusters, a reasonable value for the sticky time period is 600 seconds (that is,10 minutes). If your site is extremely busy, consider using a shorter sticky time period.

With the **inter-cluster sticky** option, you can configure Equalizer to direct requests from a particular client to the same server on all available ports. Let's say that a server has the same service available on ports 80 and 8080. If a client attempts to connect on one port and the connection fails, tou want Equalizer to attempt to connect to the other port.

You can do this by configuring two Layer 4 clusters with the same IP address, one on port 80 and the other on port 8080, a non-zero **sticky time**, and the **inter-cluster sticky** flag enabled. Define a server in each cluster using the same IP and the appropriate port.

Note that inter-cluster stickiness only works between Layer 4 clusters. Although Layer 7 clusters automatically provide inter-cluster stickiness, inter-cluster stickiness will not work between Layer 4 and Layer 7 clusters.

Inter-cluster stickiness is provided for the case where you have similar services running on the same server IP but on different ports. If one service on one port becomdes unavailable, you'd like the traffic re-directed to the other port instead of returning an error to the client.

Using *port ranges* for a cluster achieves essentially the same effect, without using another cluster IP address (see "Layer 4 Required Tab" on page 83). Using **inter-cluster sticky** is preferable in situations where you'd like the service available on multiple cluster IPs as well as multiple ports.

You must enable **inter-cluster sticky** for all the clusters to be bound together. The clusters with enabled inter-cluster stickiness should contain identical sets of server IP addresses. For example:

```
Cluster www.coyotepoint.com:80 (HTTP)
    Server srv1 192.168.0.5
    Server srv2

Cluster www.coyotepoint.com:443 (HTTPS)
    Server srv1 192.168.0.5
    Server srv2
```

To enable sticky connections for a cluster, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster (see "Logging In" on page 31).

2. In the left frame, click the name of the Layer 4 TCP or UDP cluster to be configured. The cluster's parameters appear in the right frame.

3. Select the **Persistence** tab in the right frame.

4. In the **sticky time** field, specify the sticky time period in seconds greater than zero.

5. To direct all requests from a particular client to the same server even if the connection is to a different virtual cluster, check the **inter-cluster sticky** checkbox. You can turn on inter-cluster stickiness only if you have enabled sticky connections by specifying a **sticky time** greater than zero.

6. Click the **commit** button.

Enabling Cookies for Persistent Connections

This section does not apply to the E250

For Layer 7 HTTP and HTTPS clusters, you can enable the **persist** check box to use cookies to maintain a persistent session between a client and a particular server for the duration of the session.

When you use cookie-based persistence, Equalizer inserts a cookie into the server's response header on its way back to the client. This cookie uniquely identifies the server to which the client was connected and is included automatically in subsequent requests from the client to the same cluster. Equalizer can use the information in the cookie to route the requests to the same server. If the server is unavailable, Equalizer automatically selects a different server.

This option is enabled by default. Also see the descriptions of the **always**, **cookie age**, **cookie domain**, and **cookie path** cluster parameters under "Modifying a Layer 7 Virtual Cluster" on page 75.

Enabling the Once Only and Persist Options

Since HTTP 1.1, web browsers and servers have been able to negotiate persistent connections over which multiple HTTP transactions could take place, by specifying a *keep-alive* option in the request header. This is useful when several TCP connections are required in order to satisfy a single client request.

<div style="border:1px solid blue; color:blue; text-align:center;">This section does not apply to the E250</div>

For example, before HTTP 1.1, if a browser wished to retrieve the file *index.html* from the server `www.coyotepoint.com`, the browser would take the following actions:

1. Browser opens TCP connection to `www.coyotepoint.com`.

2. Browser sends request to server "**GET /index.html**".

3. Server responds with the content of the page (a bunch of HTML).

4. Server closes connection.

5. Browser determines that there are objects (images) in the HTML document that need to be retrieved, so the browser repeats Steps 1 to 4 for each of the objects.

As you can imagine, there is a lot of overhead associated with opening and closing the TCP connections for each image. The way HTTP 1.1 optimizes this is by allowing multiple objects (pages, images, etc) to be fetched and returned across one TCP socket connection. The client requests that the server keep the connection open by adding the request header **Connection: keep-alive** to the request.

If the server agrees, the server will also include **Connection: keep-alive** in its response headers, and the client is able to send the next request over the persistent HTTP connection without the bother of opening additional connections. This is how Equalizer behaves.

For a Layer 7 cluster, Equalizer evaluates (and possibly changes) both the request and response headers that flow between the client and server (the request and response bodies are not examined). Match rules are applied to each client header, cookies may be inserted, and headers may be rewritten. When a client includes **keep-alive** in its headers, there is a fair amount of work required by the Equalizer to determine when the next set of request headers is ready to be parsed (evaluated), since there may be quite a lot of data going across the connection between sets of headers.

To reduce this workload, the **once only** flag instructs the Equalizer to evaluate (and potentially modify) only the *first* set of headers in a connection. So, in our example above, only the headers in the request for the *index.html* file are evaluated; the subsequent requests to obtain the images are not load balanced, but sent to the same server as the first request.

Enabling **once only** is basically not compatible with persistence and Layer 7 HTTPS cluster (SSL offloading), since we generally want to examine every request in a connection when persistence or SSL offloading are enabled. Whether **once only** is enabled or not has a significant effect on how Equalizer routes requests, as summarized in the following table:

Requests in a single keep-alive connection	once only enabled	once only disabled
First Request		
persist enabled	If request contains a cookie and there is no match rule hit, send request to the server in the cookie. If request contains a cookie and there is a match rule hit, send the request to the server in the cookie *only if it is in the list of servers selected in the match rule definition*. Otherwise, ignore the cookie. If there is no cookie, load balance the request and send to the server chosen.	If request contains a cookie and there is no match rule hit, send request to the server in the cookie. If request contains a cookie and there is a match rule hit, send the request to the server in the cookie *only if it is in the list of servers selected in the match rule definition*. Otherwise, ignore the cookie. If there is no cookie, load balance the request and send to the server chosen.
persist disabled	Load balance the request and send to the server chosen.	Load balance the request and send to the server chosen.
match rule hit	Send to the server chosen by the match rule.	Send to the server chosen by the match rule.
Subsequent Requests		
persist enabled	Send to same server as *first* request (any cookie in request is ignored).	If request contains a cookie, send request to the server in the cookie. If there is no cookie, load balance request and send to server chosen by policy.
persist disabled	Send to same server as *first* request.	Load balance the request and send to the server chosen.
match rule hit	Send to same server as *first* request.	Send to the server chosen by the match rule.

For example, let's look at how Equalizer processes HTTPS requests. For an HTTPS cluster, Equalizer offloads SSL processing from the servers in the cluster; that is, Equalizer does all the SSL related processing itself, and then forwards the request in HTTP to the server. When it does this, it inserts special headers into the request to indicate that the request was received by Equalizer in HTTPS and processed into HTTP (see "HTTPS Header Insertion" on page 95). If **once only** is set, these special headers are only inserted into the *first* request in a connection; the remainder of the requests in the connection are still processed, but no headers are inserted. Most servers that support SSL offloading require that every request contain the special headers -- therefore, in most cases like this you need to disable the **once only** flag for the cluster if you want to be able to parse for these headers in every request on the server end.

The **once only** flag is enabled by default when adding an L7 cluster. In general, it is more efficient to enable **once only**; but, in situations where load balancing decisions need to be made for every request or where any of the above effects are undesirable, **once only** should be disabled.

> **Note –** Although it is permitted by the software, it is *not* recommended to define a Layer 7 cluster with **persist** and **once only** both turned off, and with no match rules. By defining a Layer 7 cluster in such a way, you are essentially disabling Layer 7 processing, while still incurring extra overhead for the Layer 7 cluster. If your application requires a cluster with no persistence, header processing, or match rules, then we recommend that you define a Layer 4 UDP or TCP cluster for the best performance.

Enabling Both the Once Only and Always Options

The **always** flag influences when Equalizer inserts cookies into server responses; it in turn is affected by the setting of the **once only** flag, as shown in the following table:

	once only enabled	**once only** disabled
always enabled	Equalizer always inserts a cookie into the *first* set of response headers on a connection *only*. The cookie is inserted regardless of whether the server included one in the response. Subsequent responses on the same connection are forwarded to the client *unchanged* by Equalizer.	Equalizer inserts its own cookie into *all* server responses on a connection. The cookie is inserted regardless of whether the server included one in the response.
always disabled	If the *first* server response on a connection already has a server cookie in it, Equalizer inserts its own cookie into the *first* set of response headers on the connection. If the response has no cookie in it, Equalizer does *not* insert one of its own. Subsequent responses on the same connection are forwarded to the client *unchanged* by Equalizer.	If the *first* server response on a connection already has a server cookie in it, Equalizer inserts its own cookie into the *first* set of response headers on the connection. Equalizer will insert a cookie into subsequent responses on the same connection if: • they do not contain a valid cookie • the **cookie generation** has changed • the server in the cookie has the **quiesce** flag enabled

This section does not apply to the E250

Note that the cluster parameters **cookie path**, **cookie age**, **cookie generation**, and **cookie domain** specify cookie content for the cluster (see "Layer 7 Persistence Tab" on page 77). If any of these parameters are updated, this changes the information used in the cookies that Equalizer inserts into server responses.

Enabling Once Only and No Header Rewrite for HTTPS

This section does not apply to the E250

In a Layer 7 HTTPS cluster, clients connect to the cluster IP using HTTPS connections. Equalizer terminates the HTTPS connection and communicates with the servers in the cluster using the HTTP protocol. By default, Equalizer examines server responses for `http://` URLs and rewrites them as `https://` URLs, so that these URLs work

properly on the client. If, for example, a server sends an HTTP redirect using the `Location:` header, this URL most likely will include the `http://` protocol. Equalizer rewrites this response so that the URL uses `https://`.

For server connections that contain multiple server responses, the setting of the **once only** flag determines whether Location: headers in all server responses are rewritten. This is shown in the table below.

Note that the Administrative Interface does not permit you to *enable* **once only** and *disable* **no header rewrite** -- this option combination would rewrite the `Location:` header in only the first response in the connection, and not rewrite the headers in subsequent responses in the same connection. Doing so would produce errors on the client.

Of course, you can also direct Equalizer to pass responses from the server *without* rewriting URLs by enabling the **no header rewrite** flag on the cluster.

	once only enabled	**once only** disabled
no header rewrite disabled	*Not supported.*	The Location: headers of *every* response in a connection are rewritten.
no header rewrite enabled	No headers are rewritten.	No headers are rewritten.

Enabling Once Only and Compression

This section does not apply to the E250

Enabling both the **once only** and **compress** options is not allowed by the Administrative Interface. These two options are not compatible, since setting them both would mean that only the first response in a connection would be compressed and not the remainder of the responses, which would likely cause client errors.

Using Active Content Verification (ACV)

Active Content Verification (ACV) is a mechanism for checking the validity of a server. When you enable ACV for a cluster, Equalizer requests data from each server in the cluster and verifies that the returned data contains a character string that indicates that the data is valid. You can use ACV with most network services that support a text-based request/response protocol, such as HTTP. Note, however, that you cannot use ACV with Layer 4 UDP clusters.

Controlling Server Verification Information

Specifying an **ACV probe** string and an **ACV response** string is one way to control the information that Equalizer uses to verify the servers. Equalizer uses the probe string to request data from each server. To verify the server's content, Equalizer searches the returned data for the response string. Equalizer sends the **ACV probe** string as part of the TCP server health check probe, and so expects to receive the **ACV response** string within the number of seconds specified by the **probe timeout** global parameter (default 10) when performing ACV.

If there is no response or the response string does not appear in the first 1024 characters of the response, the verification fails; once the number of failures equals the **strikeout threshold**, Equalizer marks the server down and stops routing requests to that server.

If requests come in that contain cookies for a persistent connection to the down server, Equalizer will attempt to route the packets to the server in the cookie, and when this fails Equalizer sends the request to the next available server in the cluster (depending on the load balancing algorithm for the cluster). For the client, this means that any connection-related data stored on the downed server (such as a shopping cart) will be lost, and the client will need to restart any operations begun that depend on that data.

How ACV works is best explained using an example. The HTTP protocol enables you to establish a connection to a server, request a file, and read the result. Figure 33 illustrates the connection process when a user requests a telnet connection to an HTTP server and requests an HTML page.

```
> telnet www.myserver.com 80 ————————————▶ User requests connection to server.
Connected to www.myserver.com ——————————▶ Telnet indicates connection is established.
> GET /index.html ———————————————————————▶ User sends request for HTML page.
<HTML> ——————————————————————————————————▶ Server responds with requested page.
<TITLE>Welcome to our Home Page</TITLE>
</HTML>
Connection closed by foreign host.——————▶ Telnet indicates server connection closed.
```
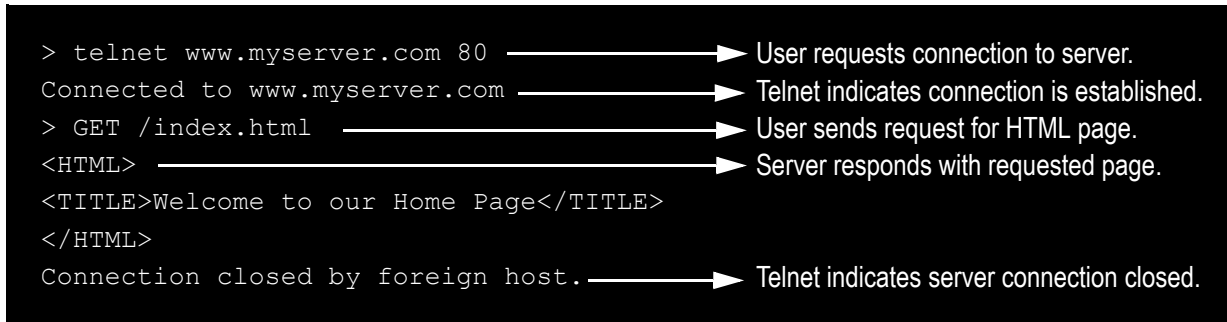
Figure 33   Retrieving content from a server via telnet.

Equalizer can perform the same exchange automatically and verify the server's response by checking the returned data against an expected result.

Specify an *ACV probe string* and an *ACV response string to* control the information that Equalizer uses to perform the verification. Equalizer uses the probe string to request data from each server. To verify the server's content, Equalizer searches the returned data for the response string.

For example, you can use "GET /index.html" as the *ACV probe string* and you can set the response string to some text, such as "Welcome" in the example in Figure 33, which appears on the home page.

Similarly, if you have a Web server with a PHP application that accesses a database, you can use ACV to ensure that all the components of the application are working. You could set up a PHP page called **test.php** that accesses the database and returns a page containing "ALL OK" if there are no problems.

Then you would enter the following values on the **add cluster** or **modify cluster** screens:

ACV probe	GET /test.php
ACV response	ALL OK

If the page that is returned contains the correct response string (in the first 1000 characters, including headers) the server is marked "up"; if "ALL OK" were not present, the server is marked down.

The response string should be text that appears only in a valid response. This string is case-sensitive. An example of a poorly chosen string would be "HTML", since most web servers automatically generate error pages that contain valid HTML.

Enabling ACV

To enable ACV in an HTTP, HTTPS, or TCP cluster, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster (see "Logging In" on page 31).

2. In the left frame, click the name of the cluster to be configured. The cluster's parameters appear in the right frame.

3. Select the **Probes** tab in the right frame.

4. In the **ACV probe** field, specify the string Equalizer will send to the server's probe port; this string should cause the application on the server's probe port to respond with a string that contains the ACV response.

Most protocols require a string to be sent to the server before a response is received. Some protocols, such as SSH, do not require a probe string; for such protocols, the ACV probe can be left blank.

5. Equalizer sends this string to each server in the cluster to request verifiable data.

> **Note –** When you set up a L7 cluster and add a probe string, `\r\n` (that is, a "carriage return" followed by a "line feed") is automatically added to the end of the string. On the other hand, when you set up a L4 cluster and add a probe string, `\r\n` is *not* automatically added to the end of the string. The reason for this different behavior is that L7 "knows" the protocol is HTTP/HTTPS but L4 does not know the protocol to be used for the probe. If required for an L4 cluster, these characters need to be added manually.

6. In the **ACV response** field, specify a case-sensitive string. An **ACV response** string must be supplied or ACV probing will not be enabled. Equalizer uses this string to verify the data with which the server responds to the ACV probe. For content verification to succeed, the specified string must appear in the first 1024 characters of the server's response (including any headers).

7. Click the **commit** button.

HTTPS Header Insertion

When a connection is established by a client for an HTTPS cluster, Equalizer performs the SSL processing on the request (this is called SSL offloading), and adds some additional headers to the client's request before forwarding the request on to a server:

```
X-LoadBalancer: CoyotePoint Equalizer

X-Forwarded-For: (client's IP address)
```

If the client provides an SSL certificate, the following are also added:

```
X-SSL-Subject: (certificate's X509 subject)

X-SSL-Issuer: (certificate's X509 issuer)

X-SSL-notBefore: (certificate not valid before info)

X-SSL-notAfter: (certificate not valid after info)

X-SSL-serial: (certs serial number)

X-SSL-cipher: (cipher spec)
```

If these headers are present in a request received by a server, then the server knows that the request was originally an HTTPS request and was processed by Equalizer before being forwarded to the server.

These headers are inserted into every request if the **once only** flag is disabled; if **once only** is enabled, then only the first request in a connection will have these headers inserted.

Some application may require a special header in the request, and the following section describes how Equalizer can be configured to provide a custom HTTPS header for such applications.

Specifying a Custom Header for HTTPS Clusters

Some applications, such as Microsoft Outlook Web Access (OWA), may require that all incoming client requests use the Secure Sockets Layer (SSL) protocol, meaning that all client requests must have the `https://` protocol in the URI.

If OWA is running on a server in an Equalizer Layer 7 HTTPS cluster, then OWA will receive all requests with `http://` in the URI, since Equalizer performs SSL processing before passing the requests on to the server.

OWA allows for SSL offloading through the use of a special header, as explained in the following Microsoft technical article:

> **http://technet.microsoft.com/en-us/library/578a8973-dc2f-4fff-83c6-39b1d771514c.aspx.)**

Two things are necessary when running OWA behind Equalizer:

- configure OWA to watch HTTP traffic for requests containing a custom header that indicates that the request was originally an SSL request that was processed by SSL offloading hardware (i.e., Equalizer) before reaching OWA (see the above article for instructions)

- configure the Equalizer cluster to add the custom header to all requests before sending them on to the OWA server (this is explained below)

Equalizer provides the ability to specify a custom header for HTTPS clusters. The following procedure shows you how to add a custom header to a new or existing HTTPS cluster definition, using the header required for an OWA server as an example.

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster (see "Logging In" on page 31).

2. In the left frame, click the name of the cluster to be configured. The cluster's parameters appear in the right frame.

3. Select the **Networking** tab in the right frame.

4. Type the following in the **custom header** field:

   `Front-End-Https: on`

5. Set other parameters and flags for the cluster as desired; see "Adding a Layer 7 Virtual Cluster" on page 75 for more details.

6. Select **commit** to create or modify the cluster.

Performance Considerations for HTTPS Clusters

Layer 7 HTTPS clusters have several features which are not present in HTTP clusters. The two most important of these features are:

> This section does not apply to the E250

- the injection of custom headers to relay to the server the fact that Equalizer terminated the HTTPS connection and performed SSL processing on the incoming request (see the previous section, above)

- the "munging", or translation, of HTTP redirects to HTTPS redirects (see the description of the **no header rewrite** flag under Modifying a Layer 7 Virtual Cluster)

One flag which frequently affects the behavior of these options is the **once only** flag. This flag is present to speed up processing of HTTP requests by only looking at the first request, but since HTTPS has a lot of overhead associated with it anyway, turning this flag off does not reduce HTTPS performance. Furthermore, having this flag on for HTTPS clusters causes some applications to not function as needed.

In general, it is recommended to turn the **once only** flag off for HTTPS clusters. This is particularly true if you're using Microsoft Internet information Service (IIS) on the servers in your cluster.

For most applications, Xcel will process several hundred HTTPS transactions per second with no noticeable degradation in performance either of the cluster or Equalizer.

In terms of bulk data throughput, the theoretical maximum throughput for Xcel/HTTPS is roughly 50% of that for the Equalizer in HTTP mode: Equalizer models with gigabit Ethernet can move HTTP traffic at wire speed (1Gbit/s) for large transfers, while Xcel can encrypt only approximately 400Mbit/s with 3DES/SHA1 or 600Mbit/s with RC4/MD5. This reflects the fact that Xcel is primarily a transaction accelerator, not a bulk data encryption device. It is noteworthy, however, that even when moving bulk data at 600Mbit/s, Xcel removes the entire load of HTTPS/SSL processing from the servers in the cluster.

One final issue to be aware of is that Xcel supports only 3DES and RC4 encryption; it does not support AES. It also does not support SSL or TLS cipher suites that use ephemeral or anonymous Diffie-Hellman exchange (cipher suites whose names contain "EDH", "DHE", or "ADH").

The default configuration for HTTPS clusters created with Xcel enabled will not use the modes described above. If, however, one either modifies the cluster's **cipher suite** string to use them, it is possible that they may be negotiated with clients. This will not lead to incorrect operation of the system, but encryption for these cipher suites will occur in software instead of taking advantage of the improved performance provided by the Xcel hardware.

Providing FTP Services on a Virtual Cluster

The FTP protocol dates from the 1970s, and was designed to be used in an environment where:

- the network topology is simple

- the FTP server and client communicate directly with one another

- the addresses used by the client and server for active FTP data connections can be negotiated over the FTP control connection

- the FTP server is able to make connections back to the FTP client

These operational characteristics of FTP require special configuration for load balancers (as well as firewalls and NAT devices) that pass traffic between FTP servers and FTP clients:

- NAT devices and routers (including load balancers like Equalizer) on the client and server sides must be configured to monitor FTP transactions and provide appropriate address translation and packet rewriting.

- Firewalls on the client and server sides must be configured to let traffic on the ports used for FTP through the firewall.

Consult the documentation for the firewalls and NAT devices used at your site to determine how to set up those devices appropriately for FTP transfers. See the next section for how to configure an Equalizer cluster for responding to FTP requests from clients.

FTP Cluster Configuration

When configuring an FTP cluster on Equalizer, the following guidelines must be followed:

1. The **protocol** for the cluster must be **Layer 4 TCP.**

2. The **start port** parameter for the cluster must be set to port **21**. (Note that port 20 is also used, but you do not specify it when adding the cluster.)

3. The **spoof** flag must be enabled for the cluster.

4. If your servers are on a network the outside world cannot reach, consider enabling Equalizer's **passive FTP translation** global flag. This option causes the Equalizer to rewrite outgoing FTP PASV control messages from the servers so they contain the IP address of the virtual cluster rather than that of the server. Note that if you select this option, clients will only be able to connect to the cluster in passive (PASV) mode.

Also observe the following notes and limitations:

- Port redirection cannot be used with an FTP cluster; that is, the port range defined for the cluster and the port ranges defined for the servers in the cluster must be identical.

- Defining a port range that includes but does not start at port 21 does *not* define an FTP cluster. The port range *must* begin at port 21. In other words, specifying a **start_port** of 19 and an **end_port** of 50 does *not* define an FTP cluster; Equalizer will assume that services other than FTP will be running on these ports.

- FTP data connections are automatically configured (internally) with a **sticky time** of one second. This is necessary to support the passive mode FTP data connection that most web browsers use. This means that

there will be one sticky record kept for each FTP data connection. For an explanation of sticky records, see "Enabling Sticky Connections" on page 88.

- FTP clusters occupy two internal virtual cluster slots, even though only one appears in the interface. This permits Equalizer's NAT subsystem to rewrite server-originated FTP data connections as they are forwarded to the external network.

- You cannot enable the **direct server return** option on an FTP cluster.

Managing Servers

The following sections discuss viewing, adding, and deleting servers, as well as server configuration options:

> The Server Table
> Server Software Configuration
> Adding a Server to a Cluster
> Modifying a Server
> Configuring Outbound NAT
> Adjusting a Server's Initial Weight
> Setting Maximum Connections per Server
> Shutting Down a Server Gracefully
> Deleting a Server

The Server Table

Every cluster has a **Servers** tab that lists all of the currently defined servers in the cluster, and provides basic configuration and status information for each server. To display the server table for a cluster, click on the cluster name in the left frame and then click on the **Servers** tab in the right frame:

Figure 34   The server table

Name	
Name	The server's name.
IP Address	The server's IP address.
Port	The server port.

	Status indicators for each server in the cluster:	
Status		The server is responding to probes and is ready to receive traffic.
		The server is not responding to probes and no traffic is being routed to it.
		The server is responding to probes, and is either disabled (the server's **initial weight** is set to 0) or the **quiesce** option is enabled.
		The server is not responding to probes, and is either disabled (the server's **initial weight** is set to 0) or the **quiesce** option is enabled.
		The server's **dont probe** option is enabled, so the probing status is unknown.
		The server's **hot spare** option is enabled. This icon appears after one of the above icons.
Actions		Delete the server from the cluster.
		Display and modify the server's configuration and option settings.
		Add a new server to the cluster.

Server Software Configuration

Please observe the following guidelines and restrictions when configuring the software on your servers:

- If the **spoof** flag is turned on for a cluster (the default), you should configure your network topology so that Equalizer is the gateway for *all* traffic for its virtual clusters. In most cases, this means that each server in a cluster should be configured to use Equalizer as its default gateway, so that all packets that come through Equalizer from clients will pass back through Equalizer and then to the clients.

  You do *not* need to configure Equalizer as the gateway for the servers if you have *disabled* the IP **spoof** flag for the cluster.

- Server responses (and client requests) must contain 64 or fewer headers; any packet that contains more than 64 headers is dropped by Equalizer (along with the connection), and a message like the following is printed to Equalizer's event log:

  ```
  Warning: Dropping connection from ip-address -- too many headers
  ```

  Make sure that your server software is configured to return 64 headers or less in any response it sends back through Equalizer.

  If your application must use 64 headers or more in server responses, then you can turn the **spoof** flag off so that server responses go back to the client *without* going through Equalizer. Be aware, however, that this has no effect on the client side; any packets from the client with more than 64 headers will still be dropped by Equalizer (and a warning appended to the event log). In most cases, client requests do not include that many headers.

Adding a Server to a Cluster

To add a server to a virtual cluster, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster (see "Logging In" on page 31).

2. In the left frame, right-click the name of the cluster to be configured and select the **Add Server** command from the menu.

3. Enter the following information:

Server Name	The logical name for the server, or accept Equalizer's default. Each server must have a unique name in the cluster that begins with an alphabetical character (for example, *CPImages)*.
Server IP Address	Enter the dotted decimal IP address of the server. This is the address Equalizer uses to communicate with the server.
Server Port	Enter the numeric port number on the Equalizer to be used for traffic between Equalizer and the server. For Layer 7 clusters, the default server port is 80. [Note that in *Layer 7 HTTPS* clusters, Equalizer performs all the SSL encryption and decryption and forwards traffic to the servers using the HTTP protocol. This is why when you add servers to an HTTPS cluster, the default server port is 80 (and should always be a port other than port 443).] For L4 UDP and L4 TCP protocol clusters, a cluster *port range* can be defined. These are the ports on the Equalizer to be used to send traffic to the servers in the cluster. Port ranges allow Equalizer users to create a single cluster to control the traffic for multiple, contiguous ports. The **Server Port** defined for a *server* in the cluster for which a port range is defined indicates the port on the server that starts the range of ports to be opened (see below).

Unless you want to set up port redirection, you can accept the default value; to redirect to a port other than the cluster port, enter the appropriate value for **Server Port**.

If a *port range* has been defined for the Layer 4 cluster to which the server is being added, the **Server Port** field refers to the first port on which to start servicing the cluster's port range. For example:

Cluster Port Range	Server Port	Port Mapping (cluster to server)
start port = 80 end port = 90	80	80 to 80 81 to 81 … 90 to 90
start port = 80 end port = 90	100	80 to 100 81 to 101 … 90 to 110

If there is no service running on one or more ports in the port range, Equalizer will still attempt to forward traffic to that port and return an error code to the client, just as if the client was connecting to the server directly.

Click the **Next** icon ⟩ .

4. The following screen allows you to associate this server definition with a VMware virtual machine, if Equalizer VLB is licensed and enabled. See "Associating a Server with a Virtual Machine" on page 246. If you are not using Equalizer VLB an VMware virtual machines as servers, click the **Next** icon ⟩ .

5. A confirmation screen appears; click **commit** to create the server with the parameters shown.

6. The **Configuration** tab for the new server is opened. See the following section for an explanation of the server configuration parameters.

Modifying a Server

The configuration tabs for a server are displayed automatically when a server is added to the system, or by selecting the server name from the left frame Configuration Tree.

1. Log into the Administrative Interface using a login that has at least **write** access for the cluster that contains the server (see "Logging In" on page 31).

2. In the left frame, select the name of the server to modify. The server **Configuration** tab opens in the right frame:

See the table below for an explanation of the server **Configuration** tab parameters:

ip	The dotted decimal IP address of the server. This is the address Equalizer uses to communicate with the server.
port	Enter the numeric port number on the Equalizer to be used for traffic between Equalizer and the server. The default is port 80. (Note that in *Layer 7 HTTPS* clusters, the server port should be set to something other than 443 since Equalizer communicates with servers in an HTTPS cluster via HTTP.) For L4 UDP and L4 TCP protocol clusters, a cluster *port range* can be defined. These are the ports on the Equalizer to be used to send traffic to the servers in the cluster. Port ranges allow Equalizer users to create a single cluster to control the traffic for multiple, contiguous ports. The **port** defined for a *server* in the cluster for which a port range is defined indicates the port on the server that starts the range of ports to be opened. See "Adding a Server to a Cluster" on page 101 for more information on defining a port range and port redirection.

probe port	By default, the server **probe port** field is set to zero and the Equalizer uses the server's **port** field value for all TCP and ACV probes. If **probe port** is not zero, Equalizer uses the value specified as the port for all TCP and ACV probes. **Note**: *For servers in Layer 7 HTTPS clusters*, set **probe port** to something other than 443, since Equalizer communicates with the servers via HTTP. In many configurations, it is set to the server **port**. (Note that the **server agent port**, set on the cluster, remains a separate port that is used only for server agent communication.)
max connections	Sets the maximum number of permitted open connections for the server. Once this limit is reached, no more traffic is routed to the server until the number of open connections falls below this limit. This limit is set by default to 0, which means that there is no maximum connections imit on the server. See "Setting Maximum Connections per Server" on page 106 for more information.
initial weight	Determines a starting point for the percentage of requests to route to each server. For information about selecting an appropriate initial weight, refer to "Adjusting a Server's Initial Weight" on page 105.
hot spare	Enable the **hot spare** check box if you plan to use this server as a backup server, in case the other servers in the cluster fail. Checking **hot spare** forces Equalizer to direct incoming connections to this server only if *all* the other servers in the cluster are down. You should only configure *one* server in a cluster as a hot spare. For example, you might configure a server as a hot spare if you are using licensed software on your servers and the license allows you to run the software only on one node at a time. In this situation, you could configure the software on two servers in the cluster and then configure one of those servers as a hot spare. Equalizer will use the second server only if the first goes down, enabling you to make your application available without violating the licensing terms or having to buy two software licenses.
quiesce	When enabled, Equalizer avoids sending new requests to the server. This is usually used in preparation for shutting down an HTTP or HTTPS server. Please see "Shutting Down a Server Gracefully" on page 108. **Note**: if a cluster receives a client request when all non-quiesced servers (including hot spares) are unavailable, then the request will be load balanced across all quiesced servers in the cluster.
dont probe	Disables High Level Probes (TCP and ACV) for the server. This is usually used to disable probe checks for a particular server without changing the probe settings for the entire cluster.
dont persist	Disables persistence for the server when the **persist** flag (Layer 7 cluster) or a non-zero **sticky time** (Layer 4 cluster) is set on the cluster. For a Layer 7 cluster, this means that no cookie will be inserted into the response header on the way back to the client. For a Layer 4 cluster, no sticky record is set. This flag is usually used to disable persistence for a hot spare. For an example, see "Using a Hot Spare in a Cluster with a Maximum Connections Limit" on page 107.

3. If you made any changes to the default configuration values, click the **commit** button to save your changes.

Configuring Outbound NAT

The **enable outbound NAT** global parameter allows servers on a non-routable network to communicate with hosts on the Internet by mapping the server's IP address to another IP address that is routable on the Internet. It is disabled by default. Enabling outbound NAT has a performance impact since Equalizer needs to modify every server response. This parameter should only be enabled when the system is configured in dual network mode, and is incompatible with single network mode operation.

In the default outbound NAT configuration, the Network Address Translation (NAT) daemon maps internal server IP addresses to Equalizer's administrative IP address on the external interface. You can also configure outbound NAT for individual servers, so server responses appear as if they came from the cluster IP address, instead of Equalizer's external interface IP address.

The address used for outbound NAT can be adjusted on a server by server basis, by clicking on the server in the left frame and opening the **Outbound NAT** tab. Outbound NAT must also be enabled in the global parameters, as shown below.

[**Note:** In previous releases, this feature was known as "advanced outbound NAT" and was available only from the command line interface.]

Enabling Outbound NAT

To enable outbound NAT:

1. Log into the Administrative Interface using a login that has **add/del** access for global parameters (see "Logging In" on page 31).

2. Click on **Equalizer** (or the configured *Failover Peer Name* for this Equalizer) in the left frame, and open the **Networking** tab in the right frame.

3. Enable the check box next to **enable outbound NAT**.

4. Select the **commit** button.

Configuring Outbound NAT for a Server

Each server defined on Equalizer can have a specifically assigned outbound NAT address that overrides the default (Equalizer's external IP address). Note that outbound NAT must be enabled globally as described in the previous section for server specific outbound NAT settings to take effect.

To configure an outbound NAT address for a server:

1. Log into the Administrative Interface using a login that has **add/del** access on the cluster to which the server belongs (see "Logging In" on page 31).

2. Click on the name of the server in the left frame, and then open the **Outbound NAT** tab in the right frame.

3. Choose one of the following options:

⊙ **Default address**
This option will set the outbound NAT address to the Default setting. This means that if this server's outbound NAT option is not set explicitly in another cluster, the Equalizer's address will be used. If it is set explicitly in another cluster, that address (either Failover or Cluster) will be used.

○ **Equalizer address**
This option will set the outbound NAT address for this server (in this and all other clusters) to use the Equalizer's administrative IP address as the source address for outbound traffic.

○ **Failover address**
This option will set the outbound NAT address for this server (in this and all other clusters) to use the Failover IP address as the source address for outbound traffic.

○ **Cluster L7-HTTP's address**
This option will set the outbound NAT address for this server (in this and all other clusters) to use the IP address of this cluster. This means that if this server (a server with the same IP address) exists in another cluster, that server's Outbound NAT option will be reset to use this address.

[commit] [cancel]

4. Click **commit** to save your selection, which takes affect immediately.

Using Outbound NAT on a Server IP in Multiple Clusters

Servers are identified in the NAT daemon configuration file by their IP addresses. If a server IP address is listed more than once in the file, it is the *last* NAT setting listed in the file that takes effect for that server IP.

This means that:

• If the same server IP address is used in more than one cluster, then changing the outbound NAT setting for one of the server instances to anything other than **Default address** changes the NAT IP address for all the instances of that server in all clusters.

• If the outbound NAT setting for one of the instances of a server is subsequently changed to something other than the default, then the new setting takes effect for all instances of the server in all clusters.

Adjusting a Server's Initial Weight

Equalizer uses a server's initial weight as the starting point for determining the percentage of requests to route to that server. As Equalizer gathers information about the actual performance of a server against client requests, it adjusts the server's current weight so that servers that are performing well receive a higher percentage of the cluster load than servers that are performing at a slower rate.

When you install servers, set each server's initial weight value in proportion to its "horsepower." All the initial weights in a cluster do not need to add up to any particular number; *it's the ratio of the assigned server weight for a server to the total of all the server weights that determines the amount of traffic sent to a server*.

For example, you might assign a server with 4 dual-core 64-bit processors operating at 3.40GHz a value of 100 and a server with 2 dual-core 64-bit processors operating at 1.86GHz a value of 50. The first server will initially receive approximately 66% (100 divided by 150) of the traffic. The second server will initially get about 33% (50 divided by 150) of the traffic. It's important to note that setting the initial weights of these servers to 100 and 50 is equivalent to setting the initial weights to 180 and 90.

Values for server weights can be in the range 0-200, with 0 meaning that no new requests will be routed to the server, essentially disabling the server for subsequent requests. In general, you should use higher initial weights. When you have enabled Equalizer's ALB feature (that is, the load balancing policy is *not* set to round robin or static weight), using higher initial weights will produce finer-grained load balancing. Higher weights enable Equalizer to adjust server weights more gradually; increasing the weight by 1 produces a smaller change if the starting weight is 100 than it does if the starting weight is 50.

However you set the initial weights, Equalizer will adjust the weight of servers dynamically as traffic goes through the cluster. Dynamic server weights might vary from 50-150% of the statically assigned values. To optimize cluster performance, you might need to adjust the initial weights of the servers in the cluster based on their performance.

> **Note –** Equalizer stops dynamically adjusting server weights if the load on the cluster drops below a certain threshold. For example, if web traffic slows significantly at 4:00 AM PST, Equalizer will not modify server weights until traffic increases again. Because a server's performance characteristics can be very different under low and high loads, Equalizer optimizes only for the high-load case. Keep this in mind when you configure new Equalizer installations; to test Equalizer's ALB performance, you'll need to simulate expected loads.

Setting initial Weights for Homogenous Clusters

If all the servers in a cluster have the same hardware and software configurations, you should set their initial weights to the same value initially. We recommend that you use a initial weight of 100 and set the load-balancing response parameter to *medium*.

As with any new configuration, you will need to monitor the performance of the servers under load for two to three hours. If you observe that the servers differ in the load they can handle, adjust their initial weights accordingly and again monitor their performance. You should adjust server weights by small increments; for example, you might set the initial weight of one server to 110 and the other to 90. Fine-tuning server weights to match each server's actual capability can easily improve your cluster's response time by 5 to 10%.

> **Note –** A change to a server's initial weight is reflected in cluster performance only after Equalizer has load balanced a significant number of new client requests for up to 30 minutes against the cluster in which the servers reside. When testing initial weights, it is most useful to use a load-generating tool to run typical client requests against the cluster to determine appropriate server initial weights.

Setting initial Weights for Mixed Clusters

Equalizer enables you to build heterogeneous clusters using servers of widely varying capabilities. Adjust for the differences by assigning initial weights that correspond to the relative capabilities of the available servers. This enables you to get the most out of existing hardware, so you can use an older server side-by-side with a new one.

After you assign relative initial weights, monitor cluster performance for two to three hours under load. You will probably fine-tune the weights and optimize performance of your cluster two or three times.

Continue monitoring the performance of your cluster and servers and watch for any trends. For example, if you notice that Equalizer *always* adjusts the dynamic weights so that the weight of one server is far below 100 and the weight of another is far above 100, the server whose dynamic weight is consistently being reduced might have a problem.

Setting Maximum Connections per Server

You can set a hard upper limit on the number of active connections per server in HTTP, HTTPS, and L4 TCP clusters. When a server reaches the maximum connections limit, requests will not be routed to that server until the number of active connections falls below the limit. The default value of 0 means that there is no maximum connections limit on a server.

Typical reasons to set a maximum number of connections include:

- implementing a connection limit that is required due to software limitations, such as an application that can service a limited number of concurrent requests

- implementing license restrictions that are not enforced by software; such as limiting the number of active connections to an application that is licensed for a limited number of concurrent connections

- setting a threshold that will limit resource utilization on the server

The **max connections** limit is set on individual servers in a cluster, and behaves as described below:

- The **max connections** limit may be ignored on Layer 7 clusters with **persist** enabled. The **persist** option tells Equalizer to insert a session cookie into all responses back to the client. When Equalizer gets another request containing the cookie, and the **max connections** limit has already been reached, it accepts the request anyway. However, if a hot spare is defined for the cluster, it sends the request to the hot spare instead. If **persist** is not enabled, **max connections** is always enforced.

- The **max connections** limit may be ignored on L4 TCP clusters with a non-zero **sticky time**. The **sticky time** option tells Equalizer to keep a "sticky record" so an L4 connection can be persistent. When Equalizer gets another request on a connection that already has a sticky record, and the **max connections** limit has already been reached, it accepts the request anyway. However, if a hot spare is defined for the cluster, it sends the request to the hot spare instead. If no **sticky time** is set, **max connections** is always enforced.

- A new flag, **dont persist**, has been introduced. It is intended to be used to override persistent connections for a hot spare in an L4 or L7 cluster whose other servers have a maximum connection limit. See the section "Using a Hot Spare in a Cluster with a Maximum Connections Limit" on page 107.

Setting Maximum Connections on a Server

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster that contains the server (see "Logging In" on page 31).

2. Do *one* of the following:

   a. Create a new server: right-click a cluster name in the left frame and select **Add Server**. After you enter and **commit** the basic information, you'll be taken to the server **Configuration** tab, where you can set max connections as shown in Step 2.

   b. Modify an existing server: click on the server name in the left frame to display the server's **Configuration** tab in the right frame.

3. Set **max connections** to a positive integer between 0 and 65535. A zero (the default) means that no connection limit is set for this server. (Set other parameters and flags for the server as desired; see Chapter 6, "*Administering Virtual Clusters*", in the *Installation and Administration Guide*, for more details.)

4. Select **commit** to save your changes to the server configuration.

Using a Hot Spare in a Cluster with a Maximum Connections Limit

When a maximum connections limit is set on all the servers in a cluster, it is often desirable to define a **hot spare** server for the cluster, so that any attempted connections to the cluster that occur after the **max connections** limit has been reached are directed to the hot spare instead of being refused or sent to the server anyway because of a persistent connection.

In this case, the hot spare could be configured to return a page to the client that contains text explaining the reason the connection has been refused. For example, the hot spare could return a page that says "All servers are currently busy -- please try again later.".

The hot spare server should be configured as follows:

- Set **max connections** to zero (0), so that all connection requests sent to the hot spare are accepted.

- Enable the **hot spare** flag. This specifies that any requests refused by all the other servers in the cluster because they reached their **max connections** limit (or are down) will be forwarded to the hot spare server.

- Enable the **dont persist** flag. We do not want connections made to the hot spare to persist. Each connection to the cluster must first be load balanced amongst the other servers in the cluster and only go to the hot spare if all the other servers have reached their **max connections** limit.

Shutting Down a Server Gracefully

To avoid interrupting user sessions, make sure that a server to be shut down or deleted from a cluster no longer has any active connections. When a server's initial weight is zero, Equalizer will not send new requests to that server. Connections that are already established continue to exist until the client and server application end them or they time out because they are idle.

To shut down servers in a generic TCP or UDP (L4) cluster, you can set the server's weight to zero and wait for the existing connections to terminate. However, you need to quiesce servers in HTTP and HTTPS (L7) clusters to enable servers to finish processing requests for clients that have a persistent session with the server.

When you quiesce a server, Equalizer does not route new connections from new clients to the server, but will still send requests from clients with a persistent session to the quiescing server. Once all the persistent sessions on the server have expired, you can set the server's initial weight to zero; then Equalizer will not send additional requests to the server.

Note that while a server is quiescing, it will still receive new requests *if all of the other servers in the cluster are unavailable*. This behavior prevents any new requests from being refused, but may lengthen the time needed to terminate all active persistent connections.

Removing a Layer 7 Server from Service

To remove a Layer 7 server from service, follow these steps:

1. In the left frame, click the name of the server to be quiesced. The server's parameters appear in the right frame.

2. Check the **quiesce** checkbox; then click **commit** to save your changes.

3. Click on **Equalizer > Status > Cluster Summary** and click the cluster name in the table. Watch the quiescing server's number of **active connections**. Once there are no active connections shown, click the server name in the left frame and set the server's weight to zero; click **commit** to save the change.

4. Click on the server name in the left frame and open the **Reporting** tab. Check the number of **total connections** (click the server name to refresh). If this number does not go to zero after a reasonable period of time, then there are clients that still have open persistent connections to the server. To make sure that these connections are not dropped, but are renegotiated after you take the server down, you can increment the cluster's **cookie generation** parameter. Click on the cluster name in the left frame and open the **Persistence** tab. Increment the **cookie generation** parameter by 1; then click **commit**.

> This section does not apply to the E250

To ensure that no cookie ever persists beyond a given time period, you can change the **cookie age** cluster parameter from the default of 0 to some number of seconds that is reasonable for your application. Then, you only need to wait that number of seconds after quiescing the server and changing its weight to 0 before it's safe to take the server down. Note that this only applies to cookies created after the change is committed.

Removing a Layer 4 Server from Service

To remove a Layer 4 server from service, follow these steps:

1. In the left frame, click the name of the server to be removed. The server's parameters appear in the right frame.

2. Set the server's weight to zero; click **commit** to save the change. This action prevents Equalizer from routing new connections to the server.

3. Click on **Equalizer > Status > Cluster Summary** and click on the cluster name in the table. Watch the server's number of **active** and **sticky** connections. Once both of these numbers are **0**, click on the server name in the left frame and check the number of total connections (click the server name to refresh). Once that number is **0** and the server's **idle time** is greater than your application's session lifetime, you can take the server offline.

Deleting a Server

To delete a server from a virtual cluster, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster that contains the server (see "Logging In" on page 31).

2. If necessary, shut the server down gracefully before taking it out of service, as shown in the section "Shutting Down a Server Gracefully" on page 108. This is particularly important if the server is in a Layer 4 cluster and may have active connections; see Step 4.

3. In the left frame, right-click the name of the server to be removed and select the **Delete Server** command from the menu.

4. When prompted, click **delete** to confirm that you want to remove the server from the cluster. Clicking **delete** removes the server from the configuration immediately. To cancel the deletion, click **cancel**. If you attempt to delete a server that has active connections:

   • If the server is being deleted from a Layer 4 cluster, clicking delete removes the server from the configuration and immediately terminates all active connections for that cluster IP and server.

   • If the server is being deleted from a Layer 7 cluster, clicking delete removes the server from the configuration, but does not terminate any active connections. Active connections for that cluster IP and server will remain open until they are completed or reach the appropriate timeout.

Automatic Cluster Responders

A Responder is a server-like object that can be associated with a Match Rule. If an incoming request matches a Match Rule expression and all of the servers specified in the Match Rule are down, a Responder definition in the Match Rule (if present) tells Equalizer to send one of two automatic responses to the client:

   • **A customized HTML "sorry page"** that can, for example, ask the client to retry later or go to another URL.

   • **A standard HTTP Redirect response** that specifies a return code and redirect URL. When the client receives this page, it is automatically redirected to the redirect URL. Redirect pages can be configured to use parts of the request URL in the HTTP Redirect response (using a regular expression).

Note – In previous releases, HTTP Redirects and "sorry pages" for clusters were configured by defining a separate server as a hot spare -- this server would return a customized page to the client when no other servers are available. Responders allow you to do this without configuring a separate server, and avoid the problem of the hot spare itself becoming unavailable. Please see "Responders and Hot Spares" on page 117 for a discussion of when each is appropriate to use in your configuration.

Note – Responder definitions are not automatically transferred between Failover pairs or amongst Envoy sites. This means that Responders need to be created and maintained separately on each Equalizer in a Failover or Envoy configuration.

Managing Responders

To display a list of all currently defined Responders, click **Responders** in the left frame to open the **Responders** tab. The table lists all the existing Responders and their configuration:

Name	Type	Status	URL	Actions
resp00	redirect	307 Temporary Redirect	http://www.example.com	🗑 🔧
resp01	sorry			🗑 🔧
				➕

Figure 35   The Responders tab

To add a Responder, you can either click the **Add** icon at the bottom of the **Actions** column, or click the Responder name in the left frame (click the plus sign next to **Responders** to display a list of Responder names).

To edit a Responder's configuration, you can either click the **Edit** icon in one of the rows in the table above, or click the Responder name in the left frame (click the plus sign next to **Responders** to display a list of Responder names).

To delete a Responder, you can either click the **Delete** icon in one of the rows in the table above, or right-click the Responder name in the left frame and select **Delete Responder** from the menu (click the plus sign next to **Responders** to display a list of Responder names). A Responder cannot be deleted if it is currently used in a match rule definition.

Adding a Responder

Responders are a "global" resource: once created, they can be individually assigned to one or more match rules in one or more clusters.

1. To create a Responder, you can either:

    •   Right-click on **Responders** in the left frame and then select **Add New Responder** from the menu.

    •   Click on **Responders** in the left frame and then select the **Add** icon in the table in the right frame.

    The **Add New Responder** dialog appears. By default, the form for creating a **Redirect Responder** is displayed:

2. Type a **Name** for the Responder or leave the default name provided.

3. Do one of the following:

- Create a standard **Redirect** page by supplying the following information in the popup screen:

Status	The HTTP status code to return to the client. The default return code is **307 (Temporary Redirect)**. Use the drop-down box to choose a different return code:
	
	    301 (Moved Permanently)
	    302 (Found)
	    303 (See Other)

URL	The HTTP Redirect URL: the full URL of the page to which the client will be redirected, as in the following example:
	
	    `http://www.coyotepoint.com/redirect/redirect.html`
	
	If a **Regex** is used to split the client URL into string variables, any variables appearing in the URL are replaced with strings from the request URL. The following is an example of a Redirct URL with named variables:
	
	    `http://$1.$2.net$3$4`

Regex	An optional POSIX-style regular expression that splits the incoming request URL into variables that can be used for string replacement in the HTTP Redirect **URL** (see above). See the section "Using Regular Expressions in Redirect Responders" on page 112.
Test	A URL used to test the return value of a **Regex** (see above). Clicking the **test** button evaluates the specified **Regex** against the specified **Test** URL; any variables in the HTTP Redirect **URL** are replaced with strings from the **Test** URL, and the result is displayed in a popup window. See the section "Using Regular Expressions in Redirect Responders" on page 112.

- Create a custom HTML page by selecting **Sorry Server**. The dialog changes to a text entry box, into which you can type the HTML that Equalizer will return to clients. The text size limit is 4096 bytes.

4. Once you are done configuring your Responder, select one of the following:

commit	Save the Responder configuration.
test	Test the Responder's **Regex** (if specified) against the **Test** URL.
cancel	Cancel the operation and do not save the Responder configuration.

Modifying a Responder

1. To modify the configuration of an existing Responder, you can either:

   - Click on the name of the Responder (under **Responders**) in the left frame.

   - Click **Responders** in the left frame and then click on the **Edit** icon in the **Action** column of the table, on the same row as the name of the Responder you want to modify.

      Edit icon

   The Responder's **Configuration** tab appears.

2. Update the Responder configuration as desired; see the previous section, "Adding a Responder" on page 110, for a description of all Responder parameters.

3. Click **commit** to save your changes.

Plotting Responder Statistics

See the section "Plotting Responder Performance History" on page 148.

Using Regular Expressions in Redirect Responders

In some cases, it may be desirable to examine the URL of an incoming request and re-use parts of it in the URL returned to the client by a Redirect Responder. This is the purpose of the **Regex** field: specify a custom regular expression that is used to:

- parse the URL of an incoming request

- break it down into separate strings

- assign each string to a named variable

These named variables can then be used in the URL field of the Redirect Responder. When the Responder replies to a client, it performs string substitution on the URL.

Because the purpose of using regular expressions to perform string substitution in Redirect URLs is to parse request URLs into strings, constructing an appropriate regular expression requires an exact knowledge of the format of the request URLs that will typically be coming in to the cluster IP.

Equalizer supports POSIX-style extended regular expressions. The following escape character sequences are supported:

****	matches a single backslash (\ )
\b	matches the beginning of a word (e.g.: `\bex` matches 'example' but not 'text')
\n, \r, \t, \v	match whitespace characters
\', \"	match single and double quotes

Example 1 -- HTTPS Redirect

The simplest form of HTTPS redirect involves simply referring the user to the top level of the https:// site, regardless of the path information that may have been included in the original request URL. For example, we could direct all requests for:

```
http://www.example.com/<path>
```

to:

```
https://www.example.com
```

But, this forces the client to re-specify the <path> after the redirect. It would be better to redirect to a URL that includes the path information:

```
https://www.example.com/<path>
```

The following regular expression:

```
^(([^ :/?#]+):)?//(.*)
```

breaks a request URL into the following named variables:

```
$0      http://www.example.com/<path>
$1      http
$2      http:
$3      www.example.com/<path>
```

We can then use these variables in the URL field as shown in the following Responder configuration screen:

Clicking the **test** button displays a popup that shows the effect of applying the **Regex** to the **Test** URL:

This Responder can be used in any cluster where a Redirect to an HTTPS cluster is desired.

Example 2 -- Multi-Hostname Redirect

Let's assume that we have a set of '.com' hostnames, all of which resolve to the same cluster IP, and we need a Responder that redirects requests to the same hostname prefixes with a '.net' suffix. We also want to include the rest of the URL exactly as specified by the client. For example, we want requests to URLs in these formats:

```
http://www.example.com/<path>
http://www.example2.com/<path>
http://www.example3.com/<path>
```

to be redirected to the following URLs:

```
http://www.example.net/<path>
http://www.example2.net/<path>
http://www.example3.net/<path>
```

The following regular expression:

```
^(([^ :/?#]+):)?//([^ \r/?#.]+)?\.([^ \r/?#.]+)?\.([^ \r/?#]+)?(/[^ \r]+)?
```

breaks the request URL into the following named variables:

```
$0      http://www.example.com/<path>
$1      http
$2      http:
$3      www
$4      example
$5      com
$6      /<path>
```

We can then use these variables in the URL field as shown in the following Responder configuration screen:

```
Add New Responder                                        (?) (X)

Name: resp00
Type:  ● Redirect  ○ Sorry Server
Status: 307 (Temporary Redirect)        ▼
URL:    http://$3.$4.net$6
Regex  ^(([^ :/?#]+):)?//([^ \r/?#.]+)?\.([^ \r/?#.]+)?\.([^ \r/?#]+)?(/[^ \r]+)?
Test:   http://www.example.com/index.html

  commit      test      cancel
```

Clicking the **test** button displays a popup that shows the effect of applying the **Regex** to the **Test** URL:

```
The page at http://192.168.0.200 says:              [X]

  ⚠   Regex matches:
       $0: "http://www.example.com/index.html"
       $1: "http:"
       $2: "http"
       $3: "www"
       $4: "example"
       $5: "com"
       $6: "/index.html"

       Redirect URL: http://www.example.net/index.html

                    [    OK    ]
```
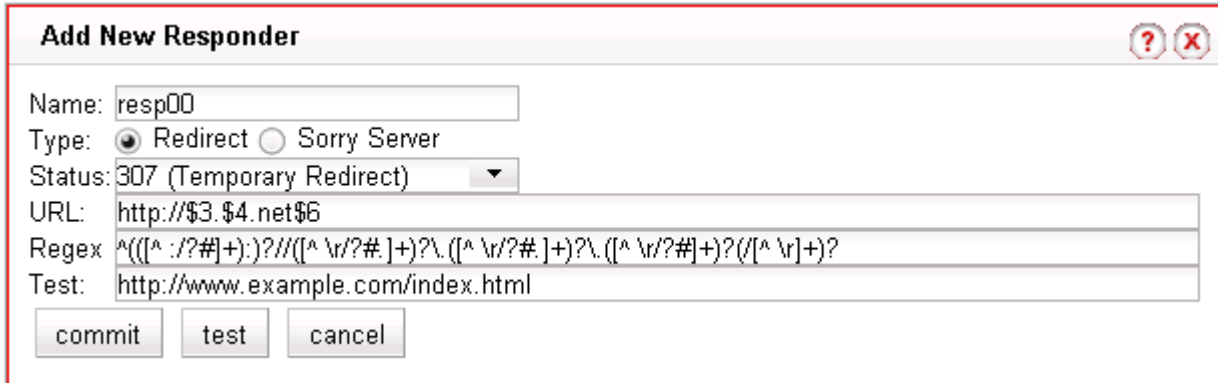
It should be noted that this example will not work for requests with destination URLs specified with an IP address for a hostname (e.g., '12.34.56.78' instead of 'www.example.com'). Providing support for IP addresses in URLs as well as DNS hostnames would involve either: a more complex regular expression that matches both; or, an additional Responder with a regular expression that matches IP addresses, as well as two match rules to match the two types of hostnames (so that the appropriate Responder replies to the client).

Example 3 -- Directory Redirect

The next example involves redirecting requests that include a particular directory to a different domain, omitting the directory from the redirect URL's path. Let's say we want all requests for:

```
http://www.example.com/images/<path>
```

to be redirected to:

```
http://images.example.com/<path>
```

The following regular expression:

```
(([^ :/?#]+):)?//([^ \r/?#.]+)?.([^ \r/?#.]+)?.([^ \r/?#]+)?(/[^ \r]+)?(/[^ \r]+)
```

breaks the request URL into the following named variables:

```
$0     http://www.example.com/images/<path>
$1     http
$2     http:
$3     www
$4     example
$5     com
$6     /images
$7     /<path>
```

We can then use these variables in the URL field as shown in the following Responder configuration screen:

Clicking the **test** button displays a popup that shows the effect of applying the **Regex** to the **Test** URL:
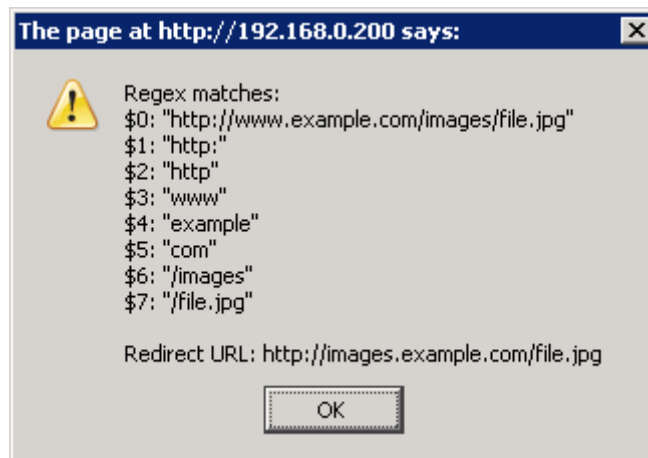
This Responder can be used in a Match Rule in any cluster where a similar directory name based redirect is required.

Using Responders in Match Rules

Once a Responder is created, it can be associated with a cluster using a Match Rule (see "Using Match Rules" on page 159). When adding a Responder to a match Rule, the way the Match Rule is configured has a direct effect on the conditions under which the mResponder isa used:

* **expression:** The default match rule **expression** [`any()`] matches all incoming requests. If you want the Responder to be used only for specific requests, then create an appropriate Match Rule expression to match those requests; see "Using Match Rules" on page 159.

* **server selection:** By default, no servers are selected in a match rule. This means that any incoming request URL that matches the match rule expression will be handled by the Responder specified in the match rule.

If you want the Responder to be used only if no servers (or particular servers) are available, select all (or some) of the the **servers** listed in the match rule configuration screen.

Once a Responder is created, it can then be selected in a match rule's **response** list. The following sections show some common Match Rule and Responder configurations.

Creating a Match Rule for a "Sorry Page"

The most common use of a Responder is to change the default match rule behavior when no servers are available in a cluster. By default, every HTTP and HTTPS cluster is created with a **Default** match rule that does not specify a Responder -- thus, if all the servers in the **Default** match rule are down, Equzlizer rdrops the client connection to the clueter.

In order to change the default behavior and supply a "sorry page" or redirect for a cluster, you need to add a new match rule that:

- matches any incoming request
- selects all servers in the cluster
- has a **Sorry Server** Responder selected

For example, let's say you have two Responders defined as in Figure 35, and there is an existing cluster that you would like to redirect to `http://www.example.com` when no servers in the cluster are available. To accomplish this, we need to create a new Responder and then add a match rule to the cluster:

1. Right-click on **Responders** in the left frame and select **Add New Responder** from the popup menu.

2. Type **SorryExample** into the **Name** field and select **Sorry Server**.

3. Type the HTML content for the page to display into the text box that appears, as shown in the following example:

```
Name: SorryExample
Type:  ○ Redirect  ◉ Sorry Server
Enter HTML for Sorry page here:

We are sorry for the inconvenience, but all of our servers at the selected
location are currently busy. Please go to the following website to select
a different location: <a
href="http://www.example.com">www.example.com</a>.




commit    test    cancel
```

4. Click **commit** to save the new Responder.

5. Right-click on the name of the cluster for which you want to display the sorry page in the left frame and select **Add Match Rule** from the menu.

6. If more than one match rule already exists in the cluster, select the appropriate position for the rule from the immediately before drop-down box.

7. Specify a **Name** for the match rule or accept the default.

8. Leave the match rule **expression** set to the default [any()] -- the rule will match all incoming requests.

9.   Select *all* the servers in the **servers** field; *the Responder will only be used if none of these servers is available*.

10. Select **SorryExample** in the **response** drop-down box.

11. Click **commit** to save the match rule.

Creating a Match Rule to Redirect All Traffic for a Specific URL

Another common cluster configuration requirement is to be able to automatically redirect all traffic that uses a specific URL. To do this, you need to add a new match rule that:

- matches any incoming request
- selects *none* of the servers in the cluster
- has a **Redirect** Responder selected

For example, let's say that we want all traffic to a cluster that uses the URL `http://`*cluster*`/special/` to be redirected to `https://www.example.com/special/`. The following procedure shows you how to add the appropriate Responder and Match Rule:

1.   Right-click on **Responders** in the left frame and select **Add New Responder** from the popup menu.

2.   Type **RedirectExample** into the **Name** field and select **Redirect**.

3.   Type **https://www.example.com/special/** into the **URL** field.

4.   Click **commit** to save the new Responder.

5.   Right-click on the name of the cluster for which you want to display the sorry page in the left frame and select **Add Match Rule** from the menu.

6.   Specify a **Name** for the match rule or accept the default.

7.   Leave the match rule **expression** set to the default [`any()`] -- the rule will match all incoming requests.

8.   Do *not* select any servers in the **servers** box. *This means that if this match rule's expression selects a request, the Responder we select will respond to the selected request regardless of the status of the servers in the cluster.*

9.   Click **commit** to create the match rule.

10. If more than one match rule already exists in the cluster, select the appropriate position for the rule from the **immediately before** drop-down box. Our example redirect rule should be immediately before the *first* existing match rule in the list.

11. Select **RedirectExample** in the **response** drop-down box.

12. Click **commit** to save the match rule.

After completing the above procedure, all client requests to `http://`*cluster*`/special/` will be redirected to `https://www.example.com/special/`, even when all the servers in the cluster are available.

More Responder Examples

More examples of using Responders and Match Rules can be found on the **Coyote Point Support Portal**, in the **Device Manuals** section.

```
http://support.coyotepoint.com
```

Responders and Hot Spares

Responders provide functionality that automates the very basic functions of a hot spare server, and offloads them onto Equalizer. If more functionality is desired, than a separate real server should be used as a hot spare for the cluster.

It should also be noted that resources Equalizer uses to service client requests via the Responder feature are resources potentially taken away from processing other client requests. In most cases, Responders might possibly have an effect on performance if all the servers in one or more clusters are down during periods of peak usage.

Configuring Smart Events

Equalizer's **Smart Control** feature allows administrators to define **Smart Events** that automate common administrative functions based on pre-set threshold values for system parameters and statistics. For example, you could specify that when the number of active servers in a particular cluster falls below a certain number, then a currently quiesced server is made active.

Equalizer provides a basic set of functions that can be used to define a Smart Event for any cluster. Additional functions are available for clusters containing VMware virtual servers running under Equalizer VLB. To enable Equalizer to communicate with a VMware vCenter (Virtual Center) or single ESX server, see Appendix F, "Equalizer VLB" on page 241. This Appendix also shows you how to configure a cluster for VLB Agents.

In releases previous to Version 8.5, Equalizer VLB (Virtualization Load Balancing) "Basic" adjusts the dynamic weight of servers in a VLB enabled cluster based on the performance data retrieved from VMware. The Equalizer VLB "Advanced" functionality enhances this feature-set, and also adds the capability to control the behavior of virtual machines, rather than just the behavior of the traffic routed to those virtual machines.

Smart Events Components

All clusters include a **Smart Events** tab that lists the currently defined Smart Events for the cluster. A newly created cluster has no Smart Events defined.

A Smart Event consists of:

- a **Trigger** expression
- an **Action** expression

The Trigger expressions of all Smart Events for a cluster are evaluated periodically at an interval set by the **smart timer** parameter on a cluster's **Probe** tab (the default is 15 seconds). If a Trigger expression evaluates to 'true', then the associated Action expression is evaluated, and the action specified by the expression result is performed.

Trigger and Action expressions are logical constructs that use Smart Event functions and operators to specify the conditions under which specific actions are to be performed. A simple example of a Trigger expression is:

```
active_servers < 2
```

This expression evaluates to 'true' when less than 2 servers in the cluster are active (i.e., are not down, quiesced, or designated a hot spare), and causes the associated Action expression to be evaluated. The following example of a simple Action expression prints a message to the system log:

```
issue_message("Cluster cl00 has fewer than 2 servers active.")
```

A Smart Event containing these Trigger and Action expressions would print a log message any time the number of active servers for the cluster is less than 2.

Smart Event Trigger and Action Expressions

The basic Smart Control model is that a specific trigger expression is set on a cluster -- if this expression evaluates to 'true', then the action expression associated with that trigger is evaluated and an action performed. Trigger and

action expressions are built using a set of Smart Event functions and variables. The Expression Editor displays the appropriate set of functions and variables for trigger and action expressions, as shown in the following table:

Trigger Functions & Variables	Description	All or VLB only
pick_server(*heuristic*)	Sets the value of the **any** server according to the specified *heuristic* (an integer). The currently supported values for *heuristic*: **1** = lowest dynamic weight **2** = highest dynamic weight **3** = unquiesced server with lowest current weight **4** = unquiesced server with highest current weight	All
quiesced(*server*)	Query to see if a server is currently quiesced. Returns 'true' if the specified server's **quiesce** flag is enabled; returns 'false' if **quiesce** is disabled.	All
true()	Returns a logical 'true' value.	All
false()	Returns a logical 'false' value.	All
server_waiting(*server*)	Query to see if a **server_wait** timer is currently in effect for the specified *server*. Returns 'true' if the specified *server* is blocked; 'false' if not.	All
event_waiting(*event*)	Query to see if a **event_wait** timer is currently in effect for the specified *event*. Returns 'true' if the specified *event* is blocked; 'false' if not.	All
numeric	Adds a numeric value to the expression. Click the drop arrow to enter the value.	All
active_servers	A variable whose value is the current number of active servers for a cluster.	All
powered(*server*)	Queries VMware and returns 'true' if the specified *server* name is associated with a virtual server that is currently powered on; otherwise, returns 'false'.	VLB
running(*server*)	Queries VMware and returns 'true' if the specified *server* name is associated with a virtual server whose operating system is currently running; otherwise, returns 'false'. **Note –** This function requires that the **VMware Tools** software is installed on the virtual machine; see the VMware documentation for instructions.	VLB
cpu_load_*server*	A variable whose value is the CPU load percentage for the specified *server* name, which must be associated with a VMware virtual server. Equalizer queries VMware for the current load on the virtual server. VMware returns the percentage of CPU resources currently in use.	VLB

Figure 36   Smart Event Trigger Functions and Variables

Equalizer Installation and Administration Guide

Action Functions & Variables	Description	All or VLB only
quiesce(*server*)	Enable the **quiesce** option for the indicated *server*. Returns 'true' if the **quiesce** option is set by this function, 'false' if not (for example, if the option is already set).	All
unquiesce(*server*)	Disable the **quiesce** option for the indicated *server*. Returns 'true' if the option is successfully disabled, 'false' if not.	All
log("*message*")	Print the specified message to the system log. Always returns 'true'.	All
shutdown(*server*)	Equalizer VLB asks VMware to shut down a virtual server. Should usually be preceded by **quiesce_server()** to make sure that the shutdown occurs when there are no more live connections on the server. Returns 'true' if the server was shutdown; 'false' otherwise.	VLB only
power_on(*server*)	Power on a VMware virtual server. Returns 'true' if the server was powered on; 'false' otherwise.	VLB only
power_off(*server*)	Power off a VMware virtual server without first shutting down the server. Should usually be preceded by **quiesce_server()** and an **event_wait()** to make sure that the shutdown occurs when there are no more live connections on the server. Returns 'true' if the server was powered off; 'false' otherwise.	VLB only
server_wait(*server*, *seconds*)	Sets the **server_wait** timer for the specified *server*, which blocks any function from operating on the *server* for the given number of *seconds*. This function always returns 'true'. Events called on blocked servers return 'false'.	All
event_wait(*event*, *seconds*)	Sets the **event_wait** timer for the specified *event*, which blocks the *event* from being evaluated for the given number of *seconds*. This function always returns 'true'.	All
pick_server(*heuristic*)	Sets the value of the **any** server according to the specified *heuristic* (an integer). The currently supported values for *heuristic*: **1** = lowest dynamic weight **2** = highest dynamic weight **3** = unquiesced server with lowest current weight **4** = unquiesced server with highest current weight	All
true()	Returns a logical 'true' value.	All
false()	Returns a logical 'false' value.	All
numeric	Adds a numeric value to the expression. Click the drop arrow to enter the value.	All

Figure 37   Smart Event Action Functions and Variables

The functions in the tables above can be combined using the operators shown below:

Operators	Description
== , > , <	numeric equals, greater than, less than
|| , && , !	logical OR, AND, NOT
( )	group two or more functions and operators
X	remove the selected function, variable, or operator from the expression

Figure 38   Smart Event Action and Trigger Operators

Complex Smart Event Expressions

The Smart Control language is flexible and allows you to combine functions, variables, and operators to create complex expressions. For example, the order of processing in a Smart Event expression honors 'short-circuiting' rules; this includes 'chained' events which use the logical OR and AND operators to decide whether expressions on the right side of the operator are evaluated.

Essentially, this means that the right hand side of an expression is not evaluated if the evaluation of the left side of the expression determines the outcome. For example: if the left side of a two-operand expression that uses the **||** operator evaluates to 'true', the right side of the expression is not evaluated. Similarly, if the left side of a two-operand expression that uses the **&&** operator evaluates to 'false', the right side of the expression is not evaluated.

Consider an event with the following **trigger** expression:

```
active_servers < 5 && server_wait(1000)
```

This trigger expression essentially overrides the default **smart timer** value (the time interval between event evaluation) if there are less than 5 servers active in the cluster: if there are less than 5 active servers, a timer is set to 1000 seconds, and after this timer expires the **action** expression for the Smart Event is processed. Therefore, this event will not be evaluated again for at least 1000 seconds. If there *are* 5 or more active servers, the event will be evaluated again after **smart timer** seconds.

Note that 'short-circuiting' in logical expressions is a standard feature of many programming languages. More information on constructing logical expressions can be found in programming texts and on the Internet.

Managing Smart Events

Smart Events are a per-cluster resource, and so are listed under the cluster name in the left frame object tree, after the servers in the cluster. Click on an existing Smart Event name to edit the event as discussed below. Click on any cluster name in the left frame and then open the **Smart Events** tab in the right frame to manage all the Smart Events for that cluster. The **Smart Events** tab lists all the currently defined Smart Events for the cluster in a table; initially, it is empty as shown below:

Name	Status	Actions
low_servers	ready	🗑 🔧
manage_spare	blocked	🗑 🔧
		➕

The **Name** column lists the Smart Event name (supplied when the event is created). The **Status** column can be one of the following: **ready** (the event is ready to be executed) or **blocked** (the event is currently blocked by a `wait_event()` function call). The buttons in the Action column allow you to **Add**, **Edit**, and **Delete** Smart Events.

Adding a Smart Event

1. To add a Smart Event to a cluster, do *one* of the following:

   • Right-click the cluster name in the left frame and select **Add Event** from the menu.

   • Click the cluster name in the left frame, open the **Smart Events** tab, and then click on the **Add** icon in the **Action** column of the table.

   Add icon

2. The **Add New Event** dialog is displayed. Enter a unique event name and click Next (**>**).

3. Enter the Trigger expression using the expression editor. See "Using the Smart Event Expression Editor" on page 124. When you are done, click Next (**>**).

4. Enter the Action expression using the expression editor. See "Using the Smart Event Expression Editor" on page 124. When you are done, click Next (**>**).

5. The trigger and action you have entered are displayed for confirmation; click **commit** to save the new event.

Editing a Smart Event

1. To view and edit a Smart Event, do *one* of the following:

   • Click on the Smart Event name in the left frame. (Use the expand control (plus sign) next to a cluster name to see all the Smart Events defined for the cluster).

   • Click the cluster name in the left frame, open the **Smart Events** tab, and then click on the **Edit** icon in the **Action** column of the table.

   Edit icon

2. The **Configuration > Trigger** tab for the new event is opened. See "Using the Smart Event Expression Editor" on page 124. If you edit the expression, click **commit** to save your changes.

3. Click on the **Action** tab to edit the action expression for the Smart Event. See "Using the Smart Event Expression Editor" on page 124. If you edit the expression, click **commit** to save your changes.
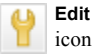
Deleting a Smart Event

1. To delete a Smart Event, do *one* of the following:

   • Right-click the Smart Event name in the left frame and select **Delete Event** from the menu. (Use the expand control (plus sign) next to a cluster name to see all the Smart Events defined for the cluster.)

   • Click the cluster name in the left frame, and open the **Smart Events** tab. Click on the name of the Smart Event you want to delete in the table and then click the **Delete** icon in the **Action** column of the table.

   Delete icon

2. A confirmation dialog is displayed. Click **delete** to delete the Smart Event.
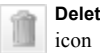
Displaying Smart Event Statistics

1. To display statistics for a Smart Event, do *one* of the following:

   • Click on the Smart Event name in the left frame. (Use the expand control (plus sign) next to a cluster name to see all the Smart Events defined for the cluster).

   • Click the cluster name in the left frame, open the **Smart Events** tab, and then click on the **Edit** icon in the **Action** column of the table.

   Edit icon

2. Open the **Reporting** tab. The following statistics are displayed:

number of times this event was processed	A Smart Event is processed, by default, every 15 seconds (see the **event interval** parameter under "Global Probe Parameters" on page 47).
number of times this event was fired	The number of times a Smart Event's trigger expression evaluated to *true*.
number of times this event was blocked	The number of times a Smart Event was blocked by an **event_wait( )** function call.

Using the Smart Event Expression Editor

Smart Control Events can be entered using a graphical editor in the Equalizer GUI. When a new event is created, you must enter a unique name for the event on the first wizard screen (a name is defaulted for you if you would not like to enter your own). The following two screens allow you to enter a **trigger** and **action** for this event.

The Expression Editor lets you choose the elements of a trigger or action expression from a menu displayed at the top of the editor screen, and manipulate them easily with the mouse. To use the editor:

- Click the button with the name of a function, variable, or operator. See Figure 36 and Figure 38 on page 122 for a list of the expression elements supported.

- The name of the function or variable appears in the edit box below; if you chose the **variable** button (used to enter data values), a blank element appears.

- If the function you chose requires a parameter, click the down arrow next to the function name in the edit box to supply the argument. Click **accept** to save the value, which now appears in the edit box.

- Click on an expression element in the edit box and then click an element in the menu to add the menu element to the expression after the element selected in the edit box.

- Click on an element and hold the mouse button to drag the selected element to a new position in the expression; release the mouse button to drop the element into its new position.

- Click on an element and select the **X** menu item to remove the selected element from the expression.

When your expression is complete, click the **commit** button to save the expression.

Smart Event Examples

Several examples of Smart Events are presented in this section, using functions that are available to all clusters. (For examples of Smart Events the use Equalizer VLB-only functions, see the section "Smart Control Event Examples Using VLB" on page 246.)

Logging a Message When Server Count is Low

Let's say we want to create a Smart Event for a cluster that prints a message to the Equalizer log any time there are fewer than 2 servers active in the cluster. To create this event:

1. Right-click on a cluster name in the left frame and select **Add Event** from the popup menu.

2. Type a **Name** for the event (or accept the default) and click the Next icon (**>**). This opens the **Event Trigger** expression editor:

Event Trigger

functions

| pick_server | quiesced | true | false | server_waiting | event_waiting | powered | running |

operators

| 0 | || | && | > | < | == | numeric | active_servers | cpu_load_[server name] | X |

expression workbench

3. In the **operators** field, click on the following controls:

   a. **active_servers**
   b. the 'less than' operator (**<**).
   c. **numeric**

   The **expression workbench** field should now look like this:

   expression workbench

   | "active_servers" | < | ▼ |

4. In the **expression workbench** field, click on the drop-down arrow shown in the blank parameter. Type '**2**' into the **numeric value** text box and then click **accept**.

5. At the top of the **Add New Event** popup window, click the Next icon (**>**). This displays the **Event Action** editor.

6. In the **functions** field, click on **log**.

7. In the **expression workbench** field, click on the drop-down arrow next to **log("")**. Type the following into the message text box that appears:

   ```
   There are fewer than two servers active.
   ```

   When you finish typing the message, click **accept**. The **expression workbench** field should now look like this:

   expression workbench

   log("There are fewer than two servers active.") ▼

8. At the top of the **Add New Event** popup window, click the Next icon (**>**).

9. A confirmation screen appears that summarizes the new event. Click **commit** to save the new event. The new event is now active, and will be evaluated at the next **smart timer** interval. Equalizer opens the event's configuration tabs so you can inspect the event definition and make further changes, if necessary.

Unquiescing a Server When Server Count is Low

Let's say we have a cluster that has three servers, **sv00**, **sv01**, and **sv02**:

- We want **sv00** and **sv01** to actively serve traffic, and **sv02** to have the **quiesce** option enabled while the other two are running.

- If one (or both) of the active servers becomes unavailable, we want to disable the **quiesce** option on **sv02** so it starts serving traffic.

- Once the original two active servers are available again, we want to quiesce the third server.

The above can be accomplished by constructing two Smart Events:

Event 1: If there are less than two servers active, unquiesce **sv02** and print a log message.

Event 2: If there are three servers active, quiesce **sv02** and print a log message.

To create **Event 1**, do the following:

1. Right-click on the cluster name in the left frame and select **Add Event** from the popup menu.

2. Type a **Name** for the event, such as **unquiesce-sv02**, or accept the default. Click the Next icon (**>**) to open the **Event Trigger** expression editor.

3. In the **operators** field, click on the following controls:

   a. **active_servers**
   b. the 'less than' operator (**<**).
   c. **numeric**

   The **expression workbench** field should now look like this:

4. In the **expression workbench** field, click on the drop-down arrow shown in the blank parameter. Type '**2**' into the **numeric value** text box and then click **accept**.

5. At the top of the **Add New Event** popup window, click the Next icon (**>**). This displays the **Event Action** editor:

6. In the **functions** field, click on **unquiesce**.

7. In the **expression workbench** field, click on the drop-down arrow next to **unquiesce**. Select **sv02** in the popup dialog and click **accept**.

8. In the **operators** field, click on **&&**.

9. In the **functions** field, click on **log**.

10. In the **expression workbench** field, click on the drop-down arrow next to **log("")**. Type the following into the message text box that appears:

    ```
    There are fewer than two servers active. sv02 has been unquiesced.
    ```

When you finish typing the message, click **accept**. The **expression workbench** field should now look like this:

```
┌ expression workbench ────────────────────────────────────────────
│   unquiesce("sv02")  ▼      &&
│
│   log("There are fewer than two servers active. sv02 has been unquiesced.")  ▼
│
```

11. At the top of the **Add New Event** popup window, click the Next icon (**>**).

12. A confirmation screen appears that summarizes the new event. Click **commit** to save the new event.

To create **Event 2**, do the following:

1. Right-click on the cluster name in the left frame and select **Add Event** from the popup menu.

2. Type a **Name** for the event, such as **quiesce-sv02**, or accept the default. Click the Next icon (**>**) to open the **Event Trigger** expression editor.

3. In the **operators** field, click on the following controls:

    a. **active_servers**
    b. the 'equals' operator (**==**).
    c. **numeric**

    In the **expression workbench** field, click on the drop-down arrow shown in the blank parameter. Type '**3**' into the **numeric value** text box and then click **accept**. The **expression workbench** field should now look like this:

```
┌ expression workbench ────────────────────────────────────────────
│   "active_servers"      ==      3  ▼
│
```

4. At the top of the **Add New Event** popup window, click the Next icon (**>**) to open the **Event Action** editor.

5. In the **functions** field, click on **quiesce**.

6. In the **expression workbench** field, click on the drop-down arrow next to **quiesce**. Select **sv02** in the popup dialog and click **accept**.

7. In the **operators** field, click on **&&**.

8. In the **functions** field, click on **log**.

9. In the **expression workbench** field, click on the drop-down arrow next to **log("")**. Type the following into the message text box that appears:

        There are three servers active. sv02 has been quiesced.

    When you finish typing the message, click **accept**. The **expression workbench** field should now look like this:

```
┌ expression workbench ────────────────────────────────────────────
│   quiesce("sv02")  ▼      &&      log("There are three servers active. sv02 has been quiesced.")  ▼
│
```

10. At the top of the **Add New Event** popup window, click the Next icon (**>**).

11. A confirmation screen appears that summarizes the new event. Click **commit** to save the new event.

Configuring Direct Server Return

In a typical load balancing scenario, server responses to client requests are routed through Equalizer on their way back to the client. Equalizer examines the headers of each response and may insert a cookie, before sending the server response on to the client.

In a Direct Server Return (DSR) configuration, the server receiving a client request responds directly to the client IP, bypassing Equalizer. Because Equalizer only processes incoming requests, cluster performance is dramatically improved when using DSR in high bandwidth applications, especially those that deliver a significant amount of streaming content. In such applications, it is not necessary for Equalizer to receive and examine the server's responses: the client makes a request and the server simply streams a large amount of data to the client.

DSR is supported on Layer 4 TCP and UDP clusters only, and is not supported for FTP clusters (Layer 4 TCP clusters with a start port of 21).

DSR configurations are usually configured in single network mode, where the cluster IP and the server IPs are all on the internal interface. An example single network mode DSR configuration is shown below:
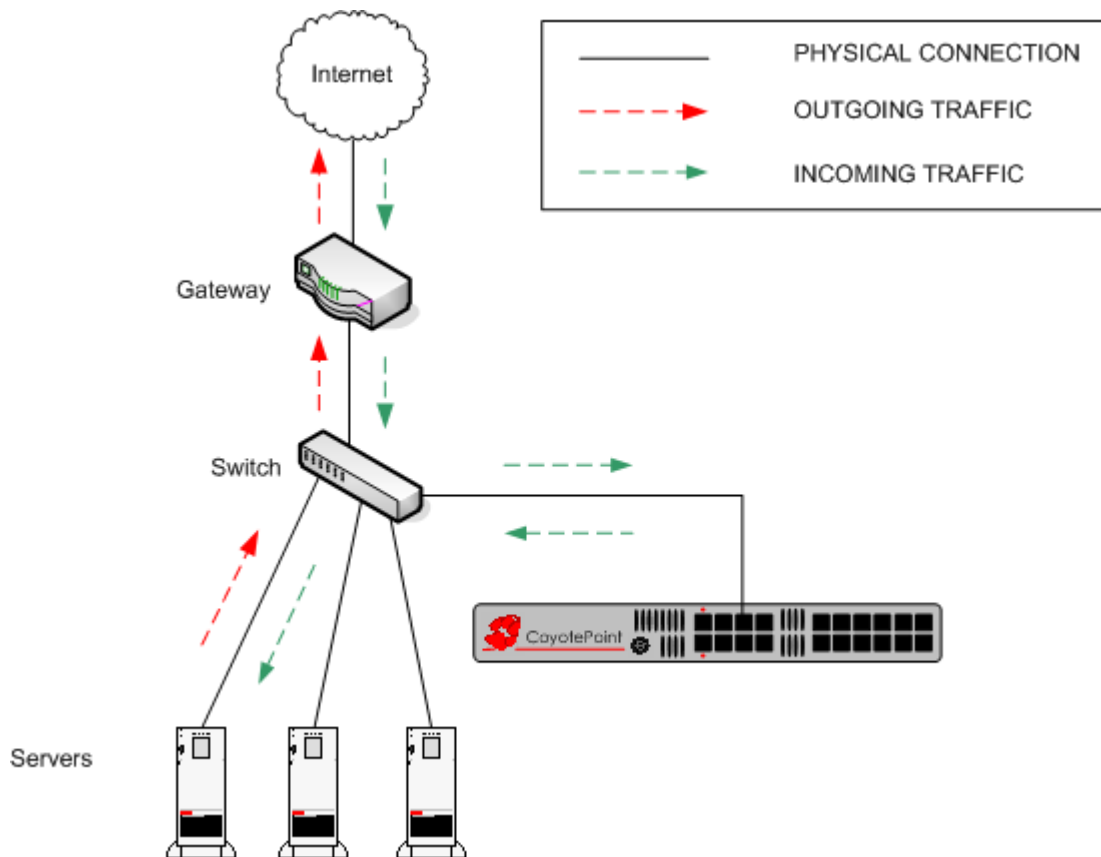
Figure 39 Example of a DSR Single Network Mode Configuration

DSR can also be used in dual network mode, although this is a less common configuration than single network mode. Cluster IPs are on the external interface, and server IPs are on the internal interface. An example of a dual network mode DSR configuration is shown below.
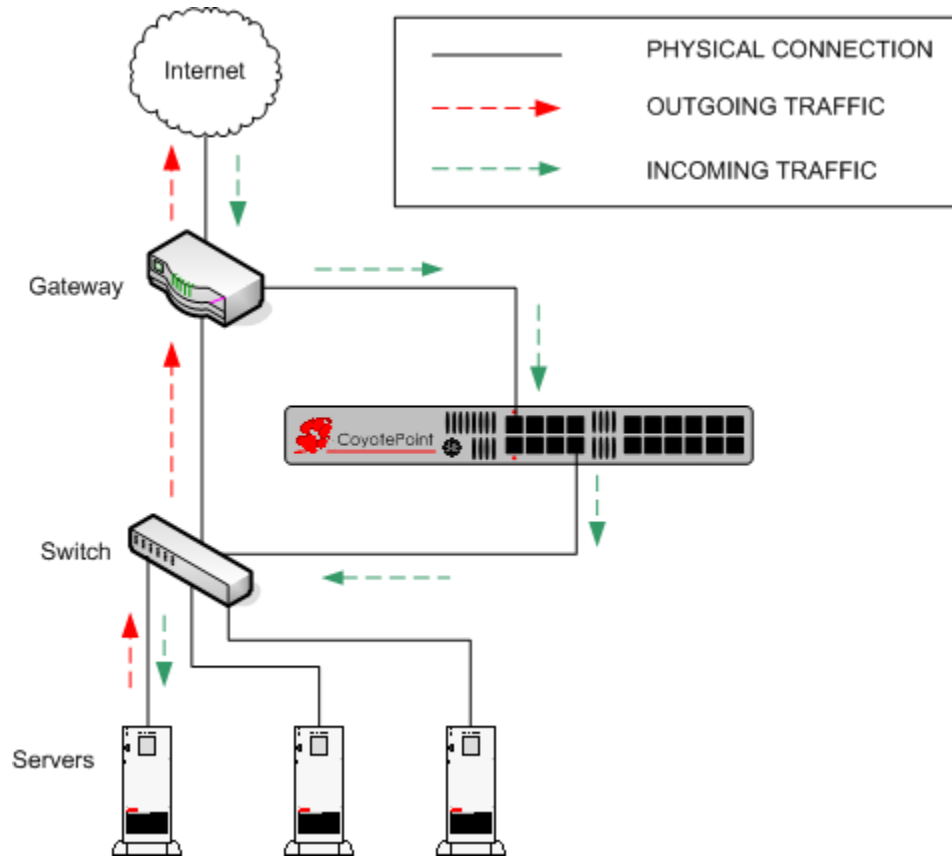
Figure 40 Example of a DSR Dual Network Mode Configuration

Configuring a Cluster for Direct Server Return

The cluster parameters **direct server return**, **spoof**, and **idle timeout** are directly related to direct server return connections:

direct server return	Enables Direct Server Return. All requests to this cluster IP will be forwarded to the server with the client IP as the source IP, and the cluster IP as the destination IP. The loopback interface of the server must be configured with the cluster IP to receive the requests. See "Configuring Servers for Direct Server Return" on page 130.
spoof	**spoof** causes Equalizer to spoof the client IP address when Equalizer routes a request to a server in a virtual cluster; that is, the IP address of the *client* is sent to the server, not the IP address of the Equalizer. This flag must be enabled for DSR.

idle timeout	The time in seconds before reclaiming idle Layer 4 connection records. (See "Layer 4 Connection Timeouts" on page 213 for a full description.) For DSR, **idle timeout** must be set to a non-zero value, or Equalizer will never reclaim connection records for connections terminated by the server. The cluster's **idle timeout** should be set to the longest period within your application that you would like Equalizer to wait for consecutive messages from the client (since the Equalizer does not see server packets on DSR connections). For example, if the longest expected server response time and the longest expected delay between client responses on active connections are both 60 seconds, then set the **idle timeout** to 120 seconds.

To create a new cluster or modify an existing one for DSR, do the following:

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster (see "Logging In" on page 31).

2. Do *one* of the following:

   a. Create a new Layer 4 TCP or UDP cluster: right-click *Equalizer* in the left frame and select **Add Cluster**. After you enter and **commit** the basic information, you'll be taken to the server **Configuration** tab.

   b. Modify an existing Layer 4 TCP or UDP cluster: click on the cluster name in the left frame to display the cluster's **Configuration** tab in the right frame.

3. Enable the **direct server return** and **spoof** check boxes.

4. If the **idle timeout** parameter is set to **0**, increase it as described in the table above.

5. Select **commit** to save your changes to the cluster configuration.

6. Add servers to the cluster by clicking the server name in the left frame and selecting **Add Server**.

7. Perform the procedure in the following section on each server that you add to the cluster.

Configuring Servers for Direct Server Return

Server configuration for DSR involves these basic steps:

1. Add a *loopback* network interface on the server.

2. Configure the loopback interface with the IP address and port of the DSR cluster.

3. Edit the configuration of the application on the server to listen for connections on the cluster IP and port. (An HTTP server, for example, returns a `Bad Hostname` error to the client if there is an IP mismatch.)

4. Check the routing on your network to ensure that traffic is being routed as expected. For example, Equalizer is usually *not* going to be used as the default gateway on your servers, since we want the servers to respond to clients directly. In most DSR configurations, the default gateway used on servers is the gateway most appropriate for reaching the client network. If routes are also needed through Equalizer, they should be configured through static routes on the servers.

The following sections show examples of configuring the loopback adapter and an HTTP server on Windows and Linux platforms for DSR:

Configuring Windows Server 2003 and IIS for DSR

The basic procedure below also applies to Windows XP and other versions of Windows.

1. Open **Start > Control Panel** and double-click **Network Connections**.

2. Select **View > Tiles**. If a **Microsoft Loopback Adapter** is already listed, proceed to the next step. Otherwise, to install the loopback interface as follows:

    a.   Open **Start > Control Panel** > **Add Hardware**, and then click **Next**.

    b.   Click **Yes, I have already connected the hardware**, and then click Next.

    c.   At the bottom of the list, click **Add a new hardware device**, and then click **Next**.

    d.   Click **Install the hardware that I manually select from a list**, and then click **Next**.

    e.   Click **Network adapters**, and then click **Next**.

    f.   In the **Manufacturer** box, click **Microsoft**.

    g.   In the **Network Adapter** box, click **Microsoft Loopback Adapter**, and then click **Next**.

    h.   Click **Finish**.

3.   To configure the loopback interface for DSR:

    a.   In **Network Connections**, right click on the **Microsoft Loopback Adapter** and select **Properties**.

    b.   In the **General** tab, double-click on **Internet Protocol (TCP/IP)** in the scroll box.

    c.   Select **Use the following IP address**, and enter the **IP address** and **Subnet mask** for the Layer 4 cluster, as configured on Equalizer. Click **OK**.

    d.   Click **OK** to return to **Network Connections**.

4.   To configure the IIS HTTP server for DSR:

    a.   Open **Start > Administrative Tools** > **Internet Information Service (IIS) Manager**.

    b.   In the left frame, expand the **local computer** and then **Web Sites** to display a list of the web sites running on the server.

    c.   Right-click on the web site you want to configure for DSR and select **Properties**.

    d.   On the **Web Site** tab, next to **IP address**, select the **Advanced** button.

    e.   Select the **Add...** button under the top list box.

    f.   Enter the **IP address** and the **TCP port** for the Layer 4 cluster, as configured on Equalizer. Click **OK**.

    g.   Click **OK** twice to return to the **Internet Information Service (IIS) Manager**.

You should now be able to send client requests to the cluster IP and port, and get responses directly from the IIS HTTP server running on Windows 2003. Remember that static routes on your servers may be necessary, depending on your network configuration.

Configuring a Linux System running Apache for DSR

This is an example of how to configure a typical Linux system running Apache 2.0 for DSR:

1.   Log into the Linux server as *root*, and enter the following command to configure a loopback interface:

```
# ifconfig lo:dsr inet cluster-ip netmask 255.255.255.255
```

Substitute the IP address of the DSR-enabled cluster on Equalizer for `cluster-ip` in the command above. Note that in most Linux distributions, you are configuring an alias for the loopback interface and should specify a netmask of 255.255.255.255 *instead of* the netmask used to configure the cluster on Equalizer.

2.   Enter the following command to verify that the loopback alias was created:

```
# ifconfig lo:dsr
```

The output should look like this:

```
lo:dsr Link encap:Local Loopback
inet addr:cluster-ip Mask:255.255.255.255
UP LOOPBACK RUNNING MTU:16436 Metric:1
```

3.   To configure an Apache 2.0 server for DSR, edit the server configuration file to add a `Listen` directive for the cluster IP (on many systems, the configuration file is found at */usr/local/etc/apache/httpd.conf*). Look for the first line beginning with the Listen directive, and add another line that looks like this:

```
Listen cluster-ip
```

Where `cluster-ip` is the DSR-enabled cluster IP. Save your changes to the file.

4. Reboot the Apache server:

```
# apachectl restart
```

You should now be able to send client requests to the cluster IP and port, and get responses directly from the Apache server running on Linux. Remember that static routes on your servers may be necessary, depending on your network configuration.

Configuring a Loopback Interface on Other Systems for DSR

The commands and interfaces used to configure a loopback interface vary slightly between operating systems, and sometimes between versions of the same operating system. Check the documentation for your server operating system for instructions on how to configure a loopback interface. For example, on some BSD systems, the command used in Step 1 in the previous section would be slightly different, as shown below:

```
# ifconfig lo0 cluster-ip netmask cluster-netmask alias
```

Notice that in this case, the netmask used matches the netmask used to configure the cluster on Equalizer, instead of 255.255.255.255 as in the Linux system example.

Weak and Strong Host Models and DSR

Network interfaces on non-routing systems use either the "weak host" or "strong host" models for packet transmission and reception (these models are defined in RFC1122). In the "strong host" model, a system that is not acting as a router cannot send or receive any packets on a given interface unless the destination/source IP in the packet is assigned to the interface. In the "weak host" model, this restriction does not apply.

In order for DSR to work, the "weak host" model must be enabled on the server's loopback interface, as well as the interface on which requests are received from Equalizer.

Most Linux and Unix systems default to the "weak host" model on all network interfaces, so no additional configuration is usually necessary. For example, on FreeBSD and NetBSD, this behavior is controlled by the setting of **sysctl net.inet.ip.check_interface**, which by default is set to 0 ("weak host").

Windows XP and Windows 2003 use the "weak host" model on all IPv4 interfaces and the "strong host" model on all IPv6 interfaces, and this is not configurable.

Windows Vista and Windows 2008 support "strong host" by default on all interfaces, but this is configurable for individual interfaces. Use the following command to list interface status:

```
netsh interface [ ipv4 | ipv6 ] show interface
```

The following three command are an example of changing the mode to "weak host" for the LAN and loopback interfaces:

```
netsh interface ipv4 set interface "Local Area Connection" weakhostreceive=enabled
netsh interface ipv4 set interface "Loopback" weakhostreceive=enabled
netsh interface ipv4 set interface "Loopback" weakhostsend=enabled
```

The interface names used in quotes above must match the interface names that appear in the Windows **Network Connections** folder.

Testing Virtual Cluster Configuration

1. After you have configured a virtual cluster and added servers, use a web browser (or just use telnet) to connect to each of the virtual clusters configured on the Equalizer from a system on your network. When you connect to a virtual cluster from the external test machine, Equalizer should send the request to one of the servers configured in the cluster, and you should see the output for that server.

2. From a client machine on the Internet, connect to each virtual cluster using a Web browser.

3. Try to reach Equalizer's Administrative Interface via the internal, external (if configured), and failover (if configured) IP addresses.

For help in resolving configuration problems, see Appendix G, "Troubleshooting" on page 253. Also visit the **Coyote Point Support Portal** (`http://www.coyotepoint.com/support.php`) for more help.
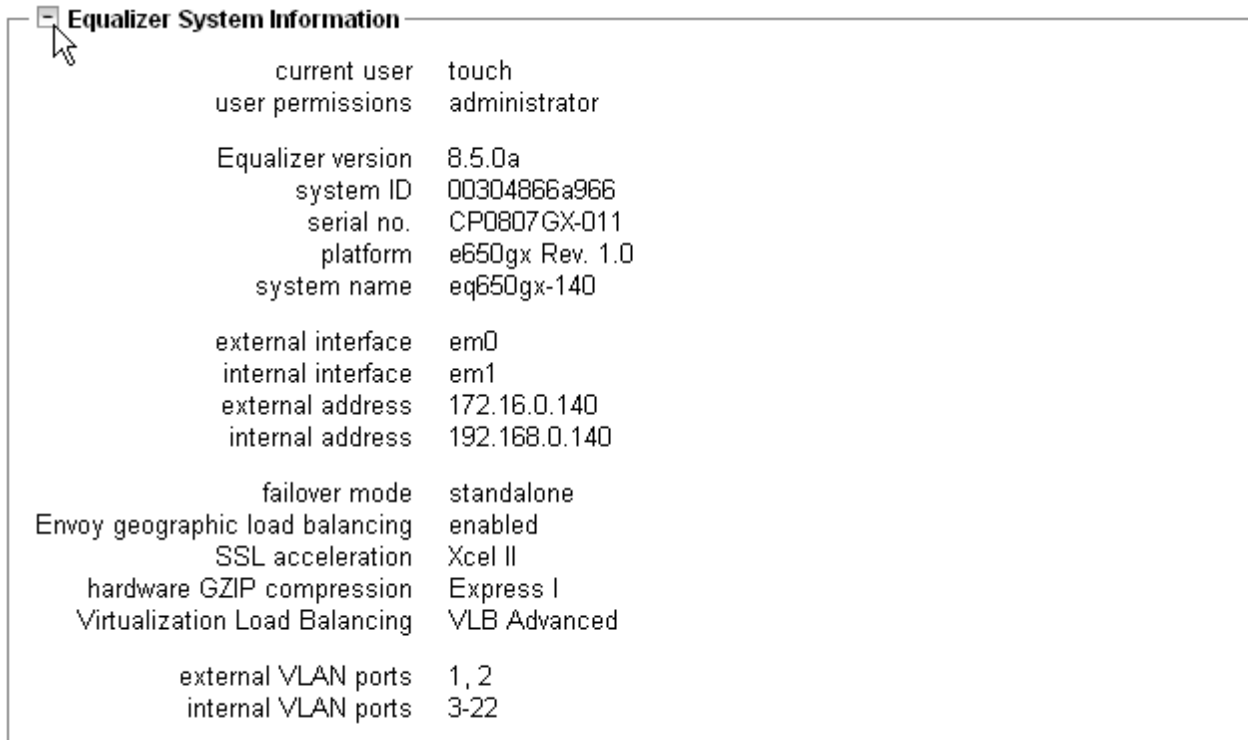
Monitoring Equalizer Operation

System status information and performance statistics can be gathered and displayed from within the Equalizer Administrative Interface. Equalizer models E350GX and above can also be monitored using standard Simple Network Management Protocol (SNMP) utilities:

Displaying Equalizer System Information

The Equalizer Status screen is displayed when you log into the Administrative interface, and anytime by selecting
Help > About:

```
┌─ Equalizer System Information ──────────────────────────┐
│                                                          │
│          current user    touch                           │
│       user permissions   administrator                   │
│                                                          │
│       Equalizer version  8.5.0a                          │
│             system ID    00304866a966                    │
│             serial no.   CP0807GX-011                    │
│              platform    e650gx Rev. 1.0                 │
│           system name    eq650gx-140                     │
│                                                          │
│       external interface em0                             │
│       internal interface em1                             │
│         external address 172.16.0.140                    │
│         internal address 192.168.0.140                   │
│                                                          │
│           failover mode  standalone                      │
│ Envoy geographic load balancing  enabled                 │
│         SSL acceleration Xcel II                         │
│  hardware GZIP compression  Express I                    │
│ Virtualization Load Balancing  VLB Advanced              │
│                                                          │
│       external VLAN ports 1, 2                           │
│       internal VLAN ports 3-22                           │
│                                                          │
└──────────────────────────────────────────────────────────┘
```

Figure 41   Equalizer system information

The Equalizer status screen displays information about Equalizer's operation mode and overall status:

current user	The login name of the currently logged in user. See "Managing Multiple Interface Users" on page 36.
user permissions	The permissions level of the **current user**. See "Objects and Permissions" on page 37.
Equalizer version	The currently running version of the Equalizer software.
system ID	The unique identifier for the Equalizer unit. [Note: in previous releases, this was shown with a colon (:) separating each pair of numbers.]
serial no.	The hardware serial number. This is the same as the serial number on the tag on the back of Equalizer's metal housing.
platform	The model number and hardware revision of Equalizer.
system name	The hostname assigned to Equalizer (default: **equalizer**).
external interface	The name of the external interface (as used, for example, in the **eqadmin** interface).
internal interface	The name of the internal interface.
external address	The IP address assigned to Equalizer's external interface.

internal address	The IP address assigned to Equalizer's internal interface.
failover mode	The current failover mode: **standalone** (no failover); **initializing** (the failover subsystem is coming up); **primary** (the system is the primary failover peer); or, **backup** (the system is the backup failover peer).
Envoy geographic load balancing	Envoy status: **enabled** (licensed) or **disabled** (not licensed).
SSL acceleration	Xcel™ SSL Hardware Acceleration status: **enabled** or **disabled**.
hardware GZIP compression	Express™ GZIP Hardware Compression status: **enabled** or **disabled**.
Virtualization Load Balancing	Equalizer VLB status: **enabled** (licensed) or **disabled** (not licensed).
internal VLAN	List the port numbers of the front panel switch ports currently assigned to the two supported VLANs. See "Managing Gigabit Switch Ports" on page 68.
external VLAN	

Displaying General Cluster Status

To display a quick view of the status of all clusters and servers defined on Equalizer, click the second item from the top of the left frame object tree; this is either **Equalizer** or, if failover is enabled, the failover peer name of the Equalizer. An example of the **General Cluster Status** table is shown below:

Figure 42 The general cluster status table

Name	The cluster name.

Type	The cluster type: one of **tcp_l4** (Layer 4 TCP), **udp_l4** (Layer 4 UDP), **http** (Layer 7 HTTP), **https** (Layer 7 HTTPS).
IP Address	The cluster IP address.
Port	The cluster port.
Servers	Status indicators for all servers in the cluster. Shows the number of servers in the following states: **Up** (responding to health check probes), **Down** (not responding to health check probes), **Quiesced** (not accepting new connections), and **Hot Spare** (only responding to requests when no other server is up).
Actions	Delete or Modify the cluster in the same row as the icon chosen. The Add icon at the bottom of the column opens the **Add New Cluster** dialog.
reset table width	The columns on the table can be resized. If you extend a column too far to the right so that other columns are no longer visible, this button returns the table to its default proportions.

Displaying the System Event Log

The System Event Log displays start-up, operating system, cluster, and server status messages. You can view the last 20, 50, 100, 200, 500, or 1000 entries in any available sub-type.

1. Select *Equalizer* > **Status** > **Event Log** to view the log:

```
max lines: 20   ▼  log type: equalizer-L7-HTTP   ▼   Refresh

Nov  4 10:41:33|        lbd|w|        L7-HTTP|        sv00|server down - fa
Nov  4 10:41:33|        lbd|w|        L7-HTTP|        sv01|server down - fa
Nov  5 04:25:29|        lbd|w|        L7-HTTP|        sv01|connect 192.168.
Nov  5 04:25:29|        lbd|w|        L7-HTTP|        sv00|connect 192.168.
```
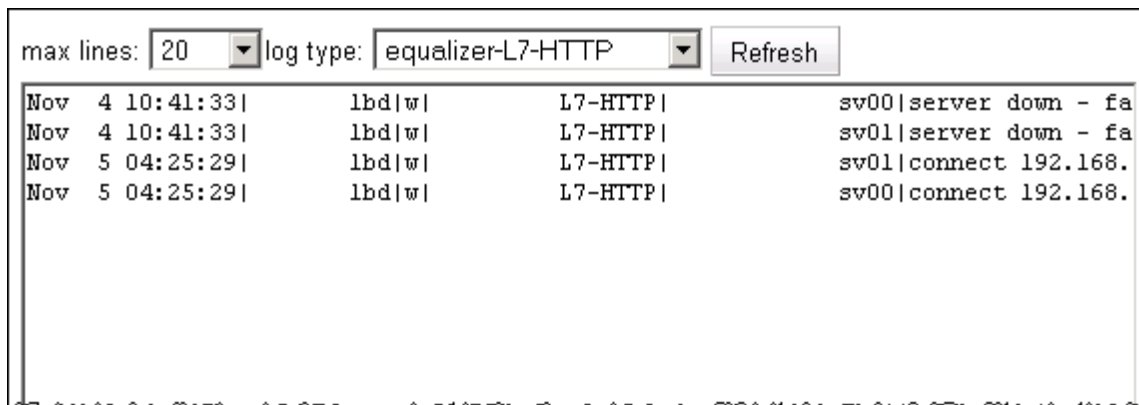
Figure 43   Viewing the system event log

Each log entry has the following general format:

```
time stamp | subsystem | type | cluster | server | event description
```

The type can be one of 'w' (warning), 'n' (notice), or 'e' (error). Use the scrollbar at the bottom of the log to see the entire event description.

2. Select the maximum number of lines to display from the bottom of the log in the **max lines** select box.

3. Select the type of messages to display in the **log type** select box: **equalizer** displays the Equalizer software log; **operating system** displays the log for Equalizer's host operating system; other entries display log entries for the appropriate cluster only.

4. Select the **Refresh** button to display the selected log entries.

To export the contents of a log, copy text from the **log viewer** screen and paste it into another application (such as Windows Notepad); then, save the text to a file.

Displaying the Virtual Cluster Summary

Select **Equalizer > Status > Cluster Summary** to open the Virtual Cluster Summary. This table displays basic status and statistics for the currently configured virtual clusters, their associated servers, and Layer 7 match rules, as shown in the example below:

Figure 44 Viewing cluster summary information

Click on a cluster name to open the summary for that cluster. For each server in a cluster, the table displays the following information:

Servers	The server name.
Status	Status indicators for each server in the cluster: **Up** (responding to health check probes), **Down** (not responding to health check probes), **Quiesced** (not accepting new connections), and **Hot Spare** (only responding to requests when no server is marked **Up**).
Initial Weight	The initial server weight asigned to the server by the administrator. This weight is used by Equalizer as it starts to load balance requests amongst the servers in the cluster. For all load balancing policies other than **static weight** and **round robin**, Equalizer adjusts the server initial weights to reflect the relative performance of the servers in the cluster over time. The dynamically adjusted server weight is displayed as **Current Weight** in the table.
Current Weight	The dynamically adjusted server weight used by equalizer when load balancing incoming requests (see Initial Weight, above). If the load balancing policy for the cluster is **static weight**, then this column shows the Initial Weight value. If the policy is **round robin**, then "**Round Robin**" is displayed instead of a value.
Processed	The total number of connections that have been processed by the server since the system was rebooted.
Active	The total number of currently active connections to this server.

Sticky	(Layer 4 clusters only): The number of "sticky records" currently held by Equalizer. Each sticky record represents a Layer 4 connection to a Layer 4 TCP or UDP cluster with a non-zero **sticky time**. See "Enabling Sticky Connections" on page 88.

For each match rule in a Layer 7 cluster, the summary displays the following information:

Processed	The number of requests that have been *evaluated* by the Match Rule since the system was last rebooted. This number includes both requests that matched and did not match the match rule expression.

If Envoy is enabled, GeoCluster names are also listed in the Cluster Summary table. For each site in a GeoCluster, the summary displays the following information:

Site	The site name.
Status	Status indicators for each site in the GeoCluster: **Up** (responding to health check probes), **Down** (not responding to health check probes), **Quiesced** (not accepting new connections), and **Hot Spare** (only responding to requests when no site is marked **Up**).
Weight	The site weights determine the relative proportion of connection requests that Equalizer routes to each site. These weights are the current, dynamically-adjusted values, not the initial weights initially assigned by the administrator.
Times Chosen	The number of times this site was selected by geographic load balancing to respond to a client request.
Times Down	The number of times this site was down when geographic load balancing was attempting to select a site to respond to a client request.

Displaying Global Connection Statistics

Click on the plus sign (+) next to **Connections** in the left frame to display the following statistics:

L4 processed	The total number of Layer 4 connections processed since the last reboot.
L4 peak	The peak (highest) number of Layer 4 connections processed *per second* since the last reboot.
L4 timeouts	The total number of partially established Layer 4 connections that were closed because either the **idle timeout** or **stale timeout** timers expired.
L7 active	The total number of currently active Layer 7 connections.
L7 processed	The total number of Layer 7 connections processed since the last reboot.
L7 peak	The peak (highest) number of Layer 7 connections processed *per second* since the last reboot.

Click on **Connections** in the left frame, or select **Equalizer > Status > Statistics**, to display the following global connection statistics. All statistics are reset when the system reboots.

Basic Statistics	
L4 total connections processed	The total number of Layer 4 connections processed.
L4 peak connections processed	The peak number of Layer 4 connections processed (in connections per second).
L4 stale connections timed-out	The total number of partially-established Layer 4 connections that were closed because the **stale timeout** for the connection expiired. See "Global Networking Parameters" on page 49.
L4 idle connections timed-out	The total number of partially-established Layer 4 connections that were closed because the **idle timeout** for the connection expiired. See "Global Networking Parameters" on page 49.
L7 current active connections	The total number of currently active Layer 7 connections. Includes partially established connections (client connections that have not yet been load balanced to a server).
L7 total connections processed	The total number of Layer 7 connections processed.
L7 peak connections processed	The peak number of Layer 7 connections processed (in connections per second).
Advanced Statistics	
L7 client connections acceptable	The number of Layer 7 client connections that were initiated.
L7 connections timed out	The number of Layer 7 connections that timed out because one of the connection timers (**client timeout**, **connect timeout**, or **server timeout**) expired.
L7 request bytes from clients	The number of bytes received in client requests.

L7 response bytes to clients	The number of bytes received in server responses.
L7 complete requests	The number of Layer 7 client requests that were completed (i.e., all headers were received before **client timeout** expired).
L7 min. usec to complete request	The minimum number of microseconds required to receive a complete client request.
L7 max. usec to complete request	The maximum number of microseconds required to receive a complete client request.
L7 avg. usec to complete request	The average number of microseconds required to receive a complete client request.
L7 maximum headers exceeded by client	The number of times a request was received that contained more than the maximum of 64 headers supported by Equalizer (connections that exceed 64 headers are dropped by Equalizer).
L7 total client connections	The total number of Layer 7 clients connections received (not necessarily processed).
L7 current client connections	The number of currently active client connections.
L7 requests processed	The total number of Layer 7 clients requests processed.
L7 responses processed	The total number of Layer 7 server responses processed.
L7 server conx reused	The number of times a server connection was kept open and re-used by Equalizer.
L7 cookies stuffed	The number of times Equalizer inserted a cookie into a Layer 7 packet.
requests in error	Number of requests that caused an error.
L7 responses in error	Number of Layer 7 responses that caused an error.
L7 client request timeouts	Number of Layer 7 requests that were dropped because the **client timeout** expired.
L7 server connect timeouts	Number of Layer 7 requests that were dropped because the **connect timeout** expired.
server response timeouts	Number of Layer 7 requests that were dropped because the **server timeout** expired.
L7 avg. usec to connect to server	The average number of seconds that Equalizer had to wait for a connection to a server.
L7 http compressed response count[1]	The total number of server responses compressed.
L7 http compressed current responses count[1]	The number of server responses currently being compressed.
L7 http bytes selected for compression[1]	The total number of input bytes from all server responses that were selected for compression.

L7 http compressed bytes output[1]	The total number of compressed bytes output from all server responses.
L7 http compression ratio[1]	The approximate current compression ratio (bytes selected for compression divided by the compressed bytes output).

[1] Note that compression statistics are only displayed if Express Hardware GZIP Compression is installed in Equalizer.

Displaying Cluster Statistics

To display statistics for a cluster, click on the cluster name in the left frame object tree, and then select the **Reporting > Statistics** tab in the right frame. The following statistics are displayed:

total number of servers	The number of servers defined for the cluster.
server active connections	The number of active (current) connections to this cluster.
total connections served	The total number of connections to this cluster since the last reboot.
time since last activity	The number of seconds since the last connection to this cluster.

Displaying Server Statistics

To display statistics for a server, click on the server name in the left frame object tree, and then select the **Reporting > Statistics** tab in the right frame. The following statistics are displayed:

server dynamic weight	The current dynamic weight for this server.
server active connections	The number of active (current) connections to this server.
total connections served	The total number of connections to this server since the last reboot.
time since last activity	The number of seconds since the last connection to this server.

Displaying Envoy Statistics

To display Envoy statistics, click on **Envoy** in the left frame object tree, and then open the **Status > Statistics** tab in the right frame. The following statistics are displayed:

DNS Requests received	The total number of DNS requests received.
Invalid DNS Requests received	The total number of Invalid DNS requests received.
Geocluster not found	The total number of DNS requests received that contained a GeoCluster name not defined on the local Equalizer.

Displaying Site Statistics

To display statistics for a Site in a GeoCluster, click on the Site name in the left frame object tree, and then select the **Reporting > Statistics** tab in the right frame. The following statistics are displayed:

total requests	The number of requests directed to this Site since the last reboot.
number queued	The number of requests queued for this site.
timed out	The number of agent-to-client triangulation probes that timed out before Equalizer received a response.
site had zero weight	The number of times the server was chosen but had a zero weight.
agent retries	The number of probes Equalizer re-sent to its agent.
agent misses	The number of Equalizer-to-agent probes that received no response. Interruptions in network connectivity between the Equalizer server and site agents and site failures can result in missed probes.
agent errors	The number of Equalizer-to-agent probes that returned a resource-unavailable error -- that is, Envoy on the remote site determined that the requested resource is unavailable.
unavailable	The number of times the server was chosen but was unavailable.
site returned	The number of clients directed to this site. You can compare this number with the values for other sites to determine the relative number of users sent to each site. If a value for one site is zero and the others are non-zero, consider why the zero site has no traffic.
returned default	The number of clients directed to the default site.
resource performance	The load on the above resource that the Equalizer agent calculates. The load incorporates data on resource response time, number of active requests, and load-balancing variables.

Plotting Global Performance History

1. Click on **Equalizer** (or the configured *Failover Peer Name* for this Equalizer) in the left frame, and open the **Status > Plots** tab in the right frame.

2. Select one or more of the following statistics to plot (all statistics are reset on reboot):

CPU Utilization	The average percent of non-idle CPU time over the selected time period.
Memory Utilization	The average percent of in-use memory over the selected time period.
L4 Connections Timed Out	The number of Layer 4 connections that were closed because a connection timeout expired (see Appendix B, "Timeout Configuration").
L7 Connections Timed Out	The number of Layer 7 connections that were closed because a connection timeout expired (see Appendix B, "Timeout Configuration").
L4 Total Connections Processed	The number of client connections to Layer 4 clusters processed.
L7 Total Connections Processed	The number of client connections to Layer 7 clusters processed.

3. Use the slider controls to select the following:

Refresh Rate	The amount of time between updates of the plot data.
Duration	The time interval displayed in the plot. Sets the horizontal time scale for the plot. For example, if **5 mins** is selected, all the data collected over the last 5 minutes is displayed in the plot.

4. The plot display is updated automatically with your settings the next time the display is refreshed.

Plotting Cluster Performance History

To display a graphical representation of the performance history of a cluster:

1. Click on the cluster name in the left frame object tree, and then select the **Reporting > Plots** tab in the right frame.

2. Select one or more of the following statistics to plot (all statistics are reset on reboot):

Servers	The average computed load of all the servers in the cluster. Because server computed loads are normalized by the cluster-wide average, the cluster-wide average should be 100. Certain events (for example, rapid fluctuations in the load, rebooting servers, and restarting application daemons such as httpd) can cause spikes in the computed load for the cluster.

|	
Service Time	The average service time of all of the servers in the cluster. The service time is the time it takes a server to start sending reply packets once it receives a client request. The average service time is a reasonable indication of the overall performance of the cluster.
Active Connections	The total number of active connections on the servers in the cluster.
Hit Rate	The number of connections served by the cluster each second. This is a good indication of how many "hits" the site is getting.
Server Agent	The average of the server agent return values for all servers in the cluster. If you have not configured server agents, -2 is displayed.

3. Use the slider controls to select the following:

Refresh Rate	The amount of time between updates of the plot data.
Duration	The time interval displayed in the plot. Sets the horizontal time scale for the plot. For example, if **5 mins** is selected, all the data collected over the last 5 minutes is displayed in the plot.

4. The plot display is updated automatically with your settings the next time the display is refreshed.

Plotting Server Performance History

To display a graphical representation of the performance history of a server:

1. Click on the server name in the left frame object tree, and then select the **Reporting > Plots** tab in the right frame.

2. Scroll down using the scrollbar at the right of the plot screen to display the plot controls:
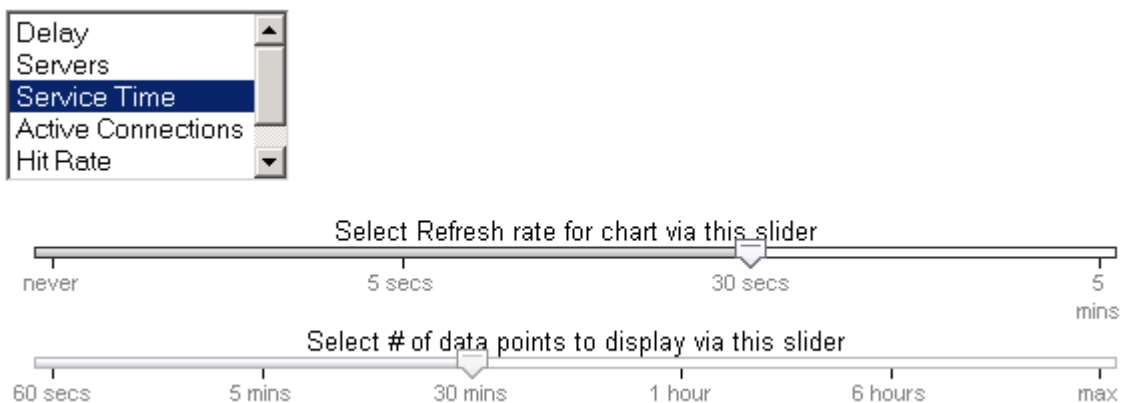
3. In the drop down box, use the **Ctrl** or **Shift** keys and the left mouse button to select one or more of the following statistics to plot:

Active Connections	The number of active connections on the server. Equalizer "smooths" the connection count using a sliding-window smoothing algorithm before being plotted. If you have enabled the Layer 4 sticky timer, note that the number of active connections on a server will be higher.

Service Time	The time it takes a server to start sending reply packets once it has received a client request. This value is very small for servers that are primarily serving static HTML pages—typically 100-200 milliseconds. If the server is serving many active pages and cgi-bins, this value will be much higher. The service time increases when the server is under heavy load because client requests are queued until the server can handle them.
Computed Load	A measure of the performance of the server relative to the overall performance of the cluster. Equalizer tries to normalize the cluster-wide computed load value to 100. If the server's computed load value is above 100, it is performing below the overall cluster performance.
	Equalizer derives a server's computed load value from its service time, number of active connections, and server agent value (if configured). It is also takes into account the load balancing policy used by the cluster.
	Ideally, a server's computed load should be around 100, though values in the range 85 to 115 are reasonable. If the server's computed load is higher than 115, the server is not performing well and you may need to add servers or upgrade to better servers. If you are using adaptive load balancing, Equalizer lowers the server's dynamic weight to reduce the number of connections sent to that server. If the server's computed load value is less than 85, the server is performing very well and Equalizer will attempt to improve cluster-wide performance by increasing the server's dynamic weight to direct more traffic to it. Such adjustments to the server's weight will in turn affect its computed load value.
Dynamic Weight	The percentage of incoming traffic that Equalizer dispatches to this server. For example, if the cluster has three servers with dynamic weights of 100, 80, and 120, the first server will get 100/(100+80+120) or 33.3% of the incoming traffic.
	If a server is down, its dynamic weight is zero. If a server crashes and reboots, the period that the server was down shows up as a gap in the dynamic weight plot.
	If you are not using adaptive load balancing (for example, the load balancing policy is set to *round robin* or *static weight*), Equalizer does not use dynamic weights. For more information about setting the load balancing policy and adaptive load balancing, refer to "Configuring a Cluster's Load-Balancing Options" on page 86.
Server Agent	The value that the server agent daemon returns. When queried, the server agent returns a value in the range -2 to 100. If you have not configured the cluster to use the server agent or the server agent daemon is not running on this server, the server agent value displayed is -2.
	Server agent values above 60 to 70 indicate that the server is overloaded. If this persists and you have enabled adaptive load balancing, Equalizer responds by reducing the server's dynamic weight so that fewer requests are routed to the server.

4. Use the slider controls to select the following:

Refresh Rate	The amount of time between updates of the plot data.

Duration	The time interval displayed in the plot. Sets the horizontal time scale for the plot. For example, if **5 mins** is selected, all the data collected over the last 5 minutes is displayed in the plot.

5. The plot display is updated automatically with your settings the next time the display is refreshed.

Plotting Match Rule Performance History

1. Click on the server name in the left frame object tree, and then select the **Reporting > Plots** tab in the right frame. The number of **Processed Connections** is the number of connections selected by the conditions of the match rule.

2. Use the slider controls to select the following:

Refresh Rate	The amount of time between updates of the plot data.
Duration	The time interval displayed in the plot. Sets the horizontal time scale for the plot. For example, if **5 mins** is selected, all the data collected over the last 5 minutes is displayed in the plot.

3. The plot display is updated automatically with your settings the next time the display is refreshed.

Plotting Responder Performance History

1. Do one of the following:

    •   Click on the Responder name in the left frame and open the **Reporting** tab in the right frame.

    •   Click **Responders** in the left frame and then click on the **Edit** icon in the **Action** column of the table, on the same row as the name of the Responder whose statistics you want to view.

Edit icon

2. The chart shows the number of **Processed Connections** for the Responder -- this is the number of times the Responder was executed by a match rule. This counter is incremented each time a Responder is executed by any match rule in any cluster.

3. Use the slider controls to select the following:

Refresh Rate	The amount of time between updates of the plot data.
Duration	The time interval displayed in the plot. Sets the horizontal time scale for the plot. For example, if **5 mins** is selected, all the data collected over the last 5 minutes is displayed in the plot.

4. The plot display is updated automatically with your settings the next time the display is refreshed.

Plotting GeoCluster Performance History

If you have licensed Envoy on your Equalizer, you can use the Plot feature to view a graphical representation of the performance history for the selected GeoCluster. To plot the performance history for a geographic cluster, follow these steps:

1. In the left frame, right-click the name of the geographic cluster whose history you want to view, and select **Plot GeoCluster** from the menu. The graphical history for the selected cluster appears in the right frame.

2. To change the information being plotted, scroll down using the scrollbar at the right of the plot screen to display the plot controls.

3. Choose the statistics to plot from the drop down box:

sites	The number of requests received for all sites in the GeoCluster since the last reboot.
network latency	The average ICMP triangulation time (if ICMP triangulation is enabled) when at least one site was able to respond. This value does not include clients for which the default site was selected.
global request rate	The number of requests received for the cluster per minute.

4. Use the slider controls to select the following:

Refresh Rate	The amount of time between updates of the plot data.
Duration	The time interval displayed in the plot. Sets the horizontal time scale for the plot. For example, if **5 mins** is selected, all the data collected over the last 5 minutes is displayed in the plot.

5. The plot display is updated automatically with your settings the next time the display is refreshed.

Plotting Site Performance History

If you have installed Envoy, the Plot Site feature enables you to view a graphical representation of the performance history for the selected site. To plot the performance history for a site, follow these steps:

1. In the left frame, right-click the name of the site whose history you want to view, and select **Plot Site** from the menu. The graphical history for the selected cluster appears in the right frame.

2. To change the information being plotted, scroll down using the scrollbar at the right of the plot screen to display the plot controls.

3. Choose the statistics to plot from the drop down box:

Site Chosen	The number of times that Equalizer returned this site in response to a client query.
No Agent Response	The number of requests in which an agent failed to reply to Equalizer's probes.
Resource Down	The number of times that the target resource failed to respond during the period plotted.
Default Chosen	The number of times the default site was chosen in response to a client query.
Agent Errors	The number of ICMP ECHO requests that the agent at this site sent to clients and for which the agent received no response.
Resource Performance	The relative workload of this site during the plotted period.

4. Use the slider controls to select the following:

Refresh Rate	The amount of time between updates of the plot data.
Duration	The time interval displayed in the plot. Sets the horizontal time scale for the plot. For example, if **5 mins** is selected, all the data collected over the last 5 minutes is displayed in the plot.

5. The plot display is updated automatically with your settings the next time the display is refreshed.

Exporting Usage Statistics

You can export usage statistics, including the data collected for plotting cluster and server histories, to a comma separated value (**.csv**) file that can be opened in any program (such as Excel) that accepts comma separated data as input. The data is exported to the browser in a file with the default name **export.csv**. All available statistical data is exported for the time period selected. To export usage statistics:

1. Select *Equalizer* **> Monitoring > Export to CSV**.

2. Select the **time period** for which you want to export the data from the drop-down box. (The size of the export file will depend on the time period selected and the number of clusters and servers in your configuration.)

3. Select **export** to download the file for saving via your browser.

The following statistics are reported in the exported file, with one line for every five seconds in the selected **time period**:

For each Cluster:	
Delay	The average service time of all of the servers in the cluster. The service time is the time it takes a server to start sending reply packets once it receives a client request.
Agent	The average of the server agent values returned for all servers in the cluster.
Connections	The average number of active connections for all servers in the cluster.
Load	The average computed load for all the servers in the cluster.
For each L7 Match Rule:	
Smoothed Processed Connections	The total number of incoming requests that were examined and matched the match rule expression.
For each Server:	
Delay	The average service time of the server. The service time is the time it takes a server to start sending reply packets once it receives a client request.
Agent	The average of the server agent values returned for the server.
Connections	The number of active connections for the server.
Load	The computed load for the server.

Total	The total number of connections processed by the server.
Time	The up time for the server.
Weight	The server's dynamic weight.
Global Statistics:	
Total Connections Processed	The total number of connections processed.
Peak Connections Processed	The peak number of connections per second processed.
Connections over last sec.	The number of connections over the last second.
Connections Timed Out	The number of connections that were dropped because one of the connection timeout counters expired.
CPU Utilization	A number indicating the percent of available CPU capacity being used.

Configuring Custom Event Handling

You can configure Equalizer to perform certain actions when a server fails or other critical events occur. You can forward Equalizer log information to another machine, and specify a command to run or email to be sent when a server event occurs.

Forwarding Equalizer Log Information

You can forward log entries from Equalizer's System Event Log (see "Displaying the System Event Log" on page 138), to another machine that is running a system logging daemon. When this option is enabled, each system event message is sent to the remote system via a UDP datagram by the **syslogd** daemon running on Equalizer. To specify a remote system logging host, follow these steps:

1. Log into the Equalizer Administration Interface (see "Logging In" on page 31).

2. Select **Equalizer > Monitoring > Events**.

Figure 45   The Events tab - logging field

3. In the **logging** field, enable the **use remote syslog** checkbox.

4. In the **syslog host** text box, type the hostname or IP address of the machine to which you want to forward **syslog** messages. The system you specify must be running a system logging daemon (such as **syslogd**) that is configured as a system logging host; see the documentation for the operating system running on that system for more information.

5. Click the **commit** button.

> **Note –** The remote **syslog** facility on Equalizer sends all **syslog** messages to the remote host, including low priority messages (e.g., information only messages) that are not displayed when viewing the operating system log via the Administrative Interface. You should edit the **syslog** configuration on the remote host to filter incoming messages so that only the priority levels in which you are interested are recorded on the remote system.

Specifying a Command to Run When a Particular Event Occurs

You can configure Equalizer to run a command that you specify (such as running a custom shell script) whenever certain events occur. The following events trigger the specified command:

- Failure of a server

- Restoration of a failed server

- Failure of a server agent

- Restoration of a server agent

- Failover in a high-availability Equalizer pair

When an event command is configured and one of the above events occurs, the command is executed and a one-line message describing the event that occurred is sent to the standard input of the specified command. This message can then be read and examined by the command to which it is passed. It is the same message that is sent via email notification for such events.

For example, the following shell script will append the current date and the event message to a file:

```
#!/bin/sh
read MSG
echo 'date': $MSG >> /tmp/echomsgs.txt
```

Once the above shell script is installed as (for example) */usr/bin/local/echomsgs* on Equalizer, you can then tell Equalizer to run the script by doing the following:

1. Log into the Equalizer Administration Interface (see "Logging In" on page 31).

2. Select **Equalizer > Monitoring > Events**

handling

Enter the full pathname of a command or script to run on any server event (server up/down, server agent up/down, failover).

command to run on server event

commit

3. In the **handling** field, enter the command that you want Equalizer to run when it detects a server event. For our example above, you would enter:

   /usr/local/bin/echomsgs

4. Click the **commit** button.

Note – Any program specified to run for a server event must complete its work and terminate within one or two seconds to avoid interrupting Equalizer's server failure detection facility.

Configuring Email Notification When a Particular Event Occurs

You can configure Equalizer to send an email notification whenever a server event occurs, for the same list of events shown in the previous section. You need to specify the sender and recipient email addresses, as well as the Simple Mail Transfer Protocol (SMTP) server for this feature to work. Any SMTP server will work with Equalizer, and usually will reside on another system on your network. The procedure below shows you how to use the **event notification** screen to configure, enable, and disable email notification.

1. Log into the Equalizer Administration Interface (see "Logging In" on page 31).

2. Select **Equalizer > Monitoring > Events**

3. In the **email notification** section, enter the sender of the email in the **from** field using the format required by your SMTP server.

   The address format to use depends on how your SMTP server is configured. For many servers, the *user@domain* (e.g.: **admin@example.com**) format will be acceptable. Some servers can be configured to require sender and recipient addresses that conform strictly to the RFC821 standard. For example, a **postfix** SMTP server has an option called **strict_rfc821_envelopes** that, when enabled, requires that all addresses must be enclosed in angle brackets, as in *<user@domain>* (e.g.: **<admin@example.com>**). If such a server receives an email whose sender or recipient addresses are not enclosed in angle brackets, the server will return an address syntax error.

   Check the settings on your SMTP server to determine the address format you need to use, or ask your network administrator.

   If you leave the **from** field blank, the default address **events@*hostname.domain*** (for example: **events@sv01.example.com**) will be used. (The hostname and domain used are part of the global parameters specified when you set up the Equalizer hardware.)

4. Enter the recipient of the email in the **to** field using the format required by your SMTP server, as described in the previous step. At least one recipient address must be specified. Specify multiple email addresses by including them in angle brackets, separated by whitespace, as in the following example:

   ```
   <recipient1@example.com> <recipient1@example.com> <recipient1@example.com>
   ```

5. Enter the SMTP address used for forwarding email using either dot notation (`10.0.0.10`) or the hostname in the **SMTP server** field. The SMTP server must be listening on port 25.

6. Check the **enable email notification** checkbox (this box allows you to turn off email notification later without removing your email configuration, as shown in the next section).

7. Click the **commit** button. (If the **to** or **SMTP server** fields are blank, or if you did not check the **enable email notification** check box, you will not be able to commit the changes.)

Disabling Email Notification When a Particular Event Occurs

To disable email notification:

1. Log into the Equalizer Administration Interface (see "Logging In" on page 31).

2. Select **Equalizer > Monitoring > Events**. On the **event configuration screen**, clear the **enable email notification** checkbox.

3. Select **commit**.

Browsing Equalizer Configurations using SNMP

The Simple Network Management Protocol (SNMP) is an internet standard that allows a management station to monitor the status of a device over the network. SNMP organizes information about the Equalizer and provides a standard way to help gather that information. Using SNMP requires:

- An SNMP agent running on the system to be monitored.

- A Management Information Base (MIB) database on the system to be monitored.

- An SNMP management station running on the same or another system.

An SNMP agent and MIB databases are provided on Equalizer Models E350GX and above, implemented for SNMPv1 and SNMPv2c.

A management station is not provided with Equalizer and must be obtained from a third party supplier. The management station is often used primarily to browse through the MIB tree, and so is sometimes called a MIB browser. One such management station that is available in a free personal edition is the iReasoning MIB Browser, available from `http://www.ireasoning.com`.

A MIB database is a hierarchical tree of variables whose values describe the state of the monitored device. A management station that want to browse the MIB database on a device sends a request to the SNMP agent running on the device. The agent queries the MIB database for the variables requested by the management station, and then sends a reply to the management station.

With SNMP, you can monitor the following information from the Equalizer MIBs:

Static configuration information, such as:

- Device name and Model
- Software version
- Internal and external IP addresses and netmasks
- Default gateway
- Failover alias

Equalizer's failover details

- Sibling Name
- Sibling Status (Primary or Secondary)

Dynamic configuration information, such as:

- Failover status
- NAT enabled
- L4 configuration state
- L7 configuration state
- Server Health check status
- Email status notification
- Cluster parameters (timeouts, buffers)
- Server parameters

Equalizer status

- L4 Statistics
- L7 Statistics

Equalizer cluster configuration

- L4 or L7 protocol of cluster
- Load balancing policy for cluster.

- IP address and port (or range)
- Sticky time and cross cluster sticky
- Cookie on or off

Enabling the SNMP Agent

The SNMP agent responds to outside SNMP requests, usually from an SNMP management station. To configure the SNMP agent, follow these steps from the Equalizer Administration Interface in Edit mode.

1. Log into the Equalizer Administration Interface (see "Logging In" on page 31).

2. Select *Equalizer* > **Monitoring > SNMP**:

Figure 46 The SNMP settings screen.

3. Enter values for the **system description**, **system location**, **system contact**, and **system name**. Description is the user-assigned description of the Equalizer. Location describes its physical location. Contact is the name of the person responsible for this unit. Name is the administrative name for the Equalizer.

4. Enter a value for the **community string**. Any SNMP management console needs to send the correct community string along with all SNMP requests. If the sent community string is not correct, Equalizer discards the request and will not respond.

5. Enter an address and port in **trap IP address:port**. This specifies the IP address and port to which trap messages should be sent. Usually this is the IP address of the machine running the SNMP management station application. The port number used by default is 162, which is the default port used by SNMP management stations; it must match the port on which the SNMP management station is listening for traps.

6. Use the check boxes to enable the corresponding traps. The following table shows the traps that are enabled or disabled using the check boxes.

Enable server up/down events	This checkbox controls two traps, `cpsSysEqServerDownEv` and `cpsSysEqServerUpEv`. Equalizer triggers these traps when it detects either a server failure or a response from a failed server.
Enable peer events	This checkbox controls two traps, `cpsSysEqSiblingContactLostEv` and `cpsSysEqSiblingContactOkayEv`. Equalizer triggers these traps whenever it is configured as part of a failover pair and it either loses or regains contact (respectively) with its peer.
Enable failover events	This checkbox controls one trap, `cpsSysEqAssumedPrimaryRoleEv`. Equalizer sends this trap whenever it assumes primary status.
Enable partition events	This checkbox controls one trap, `cpsSysEqPartitionDetectedEv`. Equalizer sends this trap whenever it is in failover mode and detects that both Equalizers have assumed primary status.

7. Make sure the **Enable SNMP Agent** checkbox is turned on to start SNMP. To disable SNMP without removing your configuration, turn off the **Enable SNMP Agent** checkbox.

8. Click **commit** to save your changes.

Setting Up an SNMP Management Station

An SNMP management station is not provided with Equalizer. In order to use SNMP to manage an Equalizer, a third-party management console must be installed and configured on a machine that can access the Equalizer system. Configuration procedures are specific to the management console used.

At a minimum, the SNMP management console needs to be configured to:

- Use the Equalizer's IP address and port 161 for SNMP requests.

- Use the **community string** specified in the above procedure.

- Use the address and port specified in the above procedure for SNMP traps (usually port 162 is used for this purpose, but this can be configured as shown in the above procedure).

- Use the Equalizer MIB definitions; these need to be loaded into the management console, following the instructions for the console. The Equalizer MIB source files are located at:

  ```
  http://<Equalizer-ip>/eqmanual/cpsreg.my
  http://<Equalizer-ip>/eqmanual/cpsequal.my
  ```

  In the above, `<Equalizer-ip>` is the IP address of the Equalizer. On the Equalizer, these are located in the directory */usr/local/www/eqmanual*.

MIB Description

Equalizer's Management Information Base (MIB) contains five major sections. These sections describe Equalizer's siblings (failover), configuration and status, clusters, servers, and events. Each object in the MIB contains a description field that describes the object's purpose. All of the MIB objects are read-only; that is, SNMP **Set** operations are not supported.

Note that Equalizer's MIB does *not* contain MIB objects for **system**, **interface**, and many other "standard" MIB object trees common to many SNMP-enabled devices. As a result, any management station or other SNMP-based software that queries for them will return an error. Only the objects defined in the *cpsreq.my* and *cpsequal.my* MIB definition files are supported by Equalizer.

The following is a summary description of the Equalizer MIB. The MIB source files contain detailed comments for each variable; these comments may also be displayed by the MIB browser when a variable is accessed.

Siblings

The main object that describes siblings is *cpsSysEqSiblings*. This describes any siblings for failover configurations.

Configuration and Status

The main object, *cpsSysEqualizer*, is the largest object in the MIB and contains many sub-objects. These sub-objects include:

eqStaticCfg - This group contains the static configuration information such as the name of the Equalizer, the software version, internal and external IP addresses and netmasks, default gateway, failover alias, etc.

eqDynamicCfg - This group consists of several sub-groups and contains no variables of its own. The sub-groups are:

> *eqGlobalDynamicCfg* - This group contains a number of global configuration items including failover status, whether or not outbound NAT is enabled, etc.

> *eqL4DynamicCfg* - This group contains configuration variables specific to Layer 4 load balancing, the state of passive FTP, idle timeout, stale timeout, etc.

> *eqL7DynamicCfg* - This group contains configuration variables specific to Layer 7 load balancing, including send and receive buffer sizes, the state of SSL encryption, etc.

eqStatus - This group consists of two sub-groups and contains no variables of it's own. The sub-groups are.

> *eqL4Status* - This group contains Layer 4 statistics such as number of connections processed, peak connections, and idle timeout count.

> *eqL7Status* - This group contains L7 statistics such as active connections, peak connections and total number of connections.

Clusters

The main object that describes clusters is *cpsSysEqClusters*. This consists of a set of tables describing the configuration of, and operational statistics for, all of the virtual clusters configured within the system.

Servers

The main object that describes servers is *cpsSysEqServers*. This consists of a set of tables describing the configuration of, and operational statistics for, all of the servers configured within each virtual cluster within the system.

Events

The main object that describes Equalizer events is *cpsSysEqEvents*. This contains variables that control whether or not traps are globally enabled and enable flags for each of the individual trap events.

Using Match Rules

Note: This chapter does not apply to the E250.

This chapter tells you all you need to know to create Layer 7 Match Rules that load balance requests based on the content in the payload of the requests, as well as the header information and other request characteristics.

Why Match Rules?

The ability to make load balancing decisions based on the content of a client request is what separates Layer 7 processing from the processing options available at Layer 4. For Layer 7 clusters, Match Rules provide fine-grained control over load balancing decisions based on the content of the client request. If you need to be able to route requests to the servers in a cluster based on the content of the request, Match Rules are the answer.

> **Note –** Match rules are supported on Equalizer Models E350 and higher models; they are *not* supported on E250 models.

Match Rules Overview

Layer 7 clusters can use logical constructs called "match rules" to control the processing of the incoming data stream from clients. Match rules extend the Layer 7 load balancing capabilities of HTTP and HTTPS clusters by allowing you to define a set of logical conditions which, when met by the contents of the request, trigger the load balancing behavior specified in the match rule.

Typically, a match rule selects the subset of servers that the load balancing algortihms will use for a particular request. By default, a request is load balanced over all the available non-spare servers in a cluster. Match rules allow you to select the group of servers that will be used to load balance the request.

For each virtual cluster, you can specify any number of match rules. For each match rule, you specify the subset of servers that can handle requests that meet the rule criteria.

A match rule provides for custom processing of requests within connections. Equalizer provides common and protocol-specific match functions that enable dynamic matching based on the request's contents. Protocol-specific match functions typically test for the presence of particular attributes in the current request.

For example, a Layer 7 HTTP virtual cluster can specify matching on specific pathname attributes to direct requests to subsets of servers so that all requests for images are sent to the image servers. The difference between load balancing with and without match rules in such a situation is illustrated in the following figure.
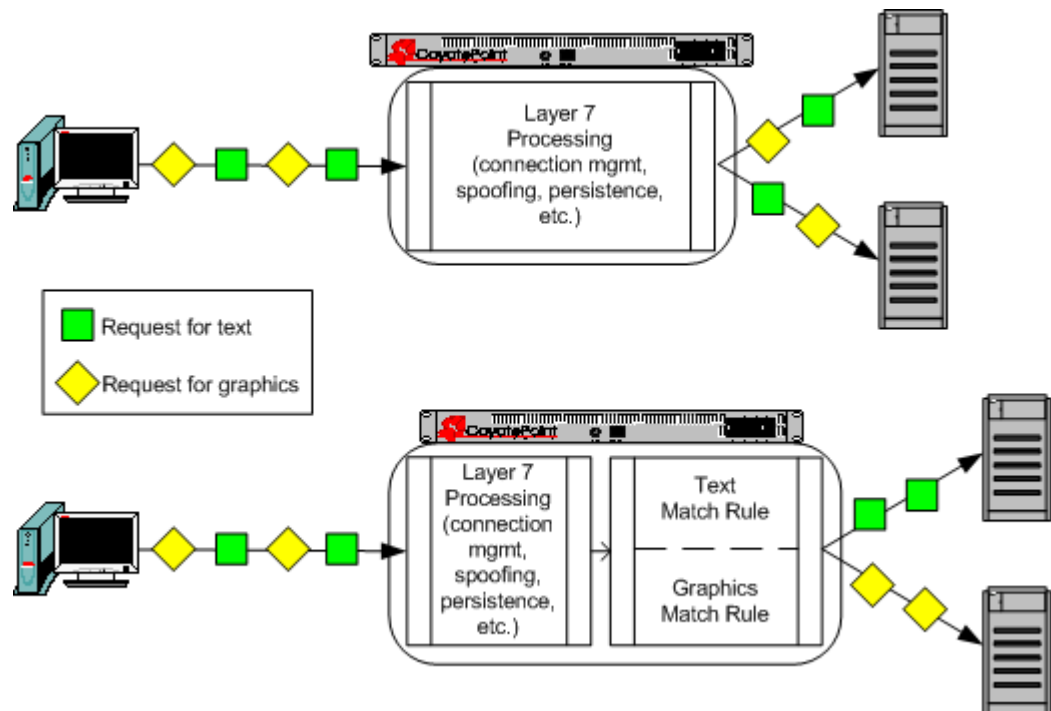
Figure 47   Conceptual Example of Match Rule Processing

Most client requests are a mix of requests for text and graphics. Layer 7 processing without Match Rules (top diagram in Figure 47) balances requests across all the available servers in the cluster, so that each server will see a mix of text and graphics requests. This means that all text and graphics must be available on each server.

Some sites may want to have one system serve only requests for graphics, and one system serve only text requests. By adding appropriate Match Rules (bottom diagram in Figure 47), Equalizer can examine each request to determine if the content requested is Text or Graphics, and send the request to the appropriate server. In this example, the servers need only hold the content they are serving, text or graphics.

Match Rule Processing

A match rule is like an if-then statement: an expression is evaluated and if it evaluates to true the body of the match rule applies to the request.

A match expression is a combination of match functions with logical operators, and can be arbitrarily complex. This allows for matching requests that have, for example:

```
(attribute A) AND NOT (attribute B)
```

If the match expression evaluates to *true*, then the data in the request has selected the match rule, and the match body applies. The *match body* contains statements that affect the subsequent handling of the request.

Multiple match rules are checked in order. Once the data in the request selects a match rule -- that is, the match rule expression evaluates to *true* -- no further match rules are checked against the request.

Equalizer makes a load balancing decision as follows:

1. If the request headers contain a cookie that specifies a server in the match rule's server list, Equalizer sends the request to the server in the cookie.

2. Otherwise, Equalizer sends the request to the server in the match rule's server list that is selected by the load balancing policy in effect for the match rule.

This process applies even if all the servers selected for the match rule are unavailable. In this case, when the match rule expression matches the request and all the servers in the match rule server list are unavailable, no reply is sent to the client. Eventually, the client sees a connection timeout.

If the match expression evaluates to *false*, then each subsequent match rule in the list of match rules for the virtual cluster is processed until a match occurs. All virtual clusters have a **Default Match** rule, which always evaluates to *true* and which will use the entire set of servers for load balancing. The Default Match rule is always processed last.

Each virtual cluster can have any number of match rules, and each match rule can have arbitrarily complex match expressions. Keep in mind that Equalizer interprets match rules for every Layer 7 cluster connection, so it is a good idea to keep match rules as simple as possible.

Match Rules, the Once Only Flag, and Cookies

Since multiple client requests may be received on a single TCP/IP connection, Equalizer has a flag (**once only**) that specifies whether to check the headers in every request received on a connection, or to load balance based solely upon the first set of headers received on a connection (and ignore the headers in subsequent requests on the same connection).

The **once only** flag is both a global and cluster parameter, and appears on the **Networking** tab. When using Match Rules, it is usually desirable to turn *off* the **once only** flag for the cluster so that Equalizer matches against each individual request on the stream, not just the initial one.

You can also enable or disable **once only** in a match rule, to override the setting on the cluster for any request that matches that rule. For example, if **once only** is enabled on a cluster and disabled on a match rule, any request that matches that match rule's expression will be load balanced as if **once only** were disabled on the cluster.

The following table shows how the setting of **once only** affects load balancing when a match rule hit occurs:

match rule hit on...	once only enabled	once only disabled
...the first request on a connection	If the request headers contain a cookie specifying a server in the match rule's server list, send the request to the server in the cookie. Otherwise, send the request to the server in the match rule's server list that is selected by the load balancing policy in effect for the match rule.	The request is load balanced as described for the first request with **once only** enabled.
...second and subsequent requests on the same connection	If the request headers contain a cookie specifying a server in the match rule's server list, send the request to the server in the cookie. Otherwise, send the request to the server that was selected by the first request.	The request is load balanced as described for the first request with **once only** enabled.

Note that Equalizer always honors a cookie that specifies a server in the match rule's server list, regardless of the setting of the **once only** flag: the request is sent to the server specified by the cookie. If, however, the cookie specifies a server that is *not* in the match rule's server list, the cookie is ignored.

General Match Expressions and Match Bodies

A match rule consists of a *match expression* and a *match body,* which identifies the operations to perform if the expression is satisfied by the request. Match syntax is as follows:

```
match name { expression } then { body }
```

Each match has a name, which is simply a label. The name must follow the same restrictions as those for cluster names and server names. All match names within a cluster must be unique.

Match Expressions

Match expressions affect the subsequent processing of the request stream using URI, host, or other information. Match expressions are made up of match functions, most of which are protocol-specific, joined by logical operators, optionally preceded by the negation operator, with sets of beginning and end parentheses for grouping where required. This may sound complex, and it can be, but typical match expressions are simple; it is usually best from a performance perspective to keep them simple.

The most simple match expression is one made up solely of a single match function. The truth value (*true* or *false*) of this expression is then returned by the match function. For example, a match function common to all Layer 7 protocols is the `any()` function, which always returns *true*, independent of the contents of the request data. So, the most simple match expression is:

```
any()
```

which will always result in the match rule being selected.

Use the logical NOT operator, (sometimes), to invert the sense of the truth value of the expression. So, you can use the NOT operator to logically invert a match expression, as follows:

```
NOT expression
```

giving rise to the next simplest example:

```
NOT any()
```

which always evaluates to *false* and always results in the match rule not being selected.

With the addition of the logical OR (`||`) and logical AND (`&&`) operators, you can specify complex expressions, selecting precise attributes from the request, as in this:

```
NOT happy() || (round() && happy())
```

Match expressions are read from left to right. Expressions contained within parentheses get evaluated before other parts of the expression. The previous expression would match anything that was not happy or that was round and happy.

Note – The the logical negation operator is displayed as "NOT", rather than "!".

Unlike the previous example, match functions correspond to certain attributes in a request header.

For example, a request URI for a web page might look like this:

```
Get /somedir/somepage.html   http/1.1
Accept: text/html, text/*, *.*
Accept-Encoding: gzip
Host: www.coyotepoint.com
User-Agent: Mozilla/4.7 [en] (Win98; U)
```

Various functions return true when their arguments match certain components of the request URI. Using the above request URI, for example, you could use several match functions:

- **pathname**() return*s true if its argument matches* `/somedir/somepage.html`

- **dirname**() retu*rns true if its argument matches* `/somedir/`

- **filename**() retu*rns true if its argument matches* `somepage.html`

Other functions can evaluate the contents of the `Host` header in the request URI above:

```
host (www.coyotepoint.com)
host_prefix (www)
host_suffix (coyotepoint.com).
```

Some function arguments can take the form of a regular expression[1]. Note that you cannot put regular expressions into match expressions except as an argument to a function whose definition supports regular expressions.

Note – Matching regular *expressions* (*using *_regex() functions*) is many times more processing-intensive than using other match functions. It is usually possible to avoid using regular expressions by carefully crafting match expressions using other functions. For example, the following regular expression match:

```
dirname_regex("(two|four|six|eight)")
```

Can be replaced by the more efficient:

```
dirname_substr("two") ||
dirname_substr("four") ||
dirname_substr("six") ||
dirname_substr("eight")
```

Match Bodies

Match bodies specify the actions to take if the match expression selects the request. This is specified in the form of statements that provide values to variables used by the load balancer to process the request. The most common (and most useful) match body selects the set of servers over which to apply the load balancing:

```
servers = all;
```

The `servers` assignment statement takes a comma-separated list of server names, which specifies the set of servers to be used for load balancing all requests that match the expression in the match rule. The reserved server names `all` and `none` specify respectively the set of *all* servers in the virtual cluster and *none* of the servers in the virtual cluster. If you do not assign servers, none will be available for load balancing; as a result, the connection to the client will be dropped.

In general, you can override most cluster-specific variables in a match body. (You can override protocol-specific variables as well, but that does not always make sense.) One useful example of overriding variables is as follows:

```
servers = s0, s1, s2;
flags = !once_only;
```

which would load-balance across the specified servers (which first must be defined in the virtual cluster) and also turn off the `once_only` flag for the duration of processing of that connection.

---

1. Regular expressions are specified according to IEEE Std 1003.2 ("POSIX.2").

Match Rule Definitions

Match rules are defined in the file */var/eq/eq.conf* with the definition of the cluster to which the match rule applies. A match rule as it appears in *eq.conf* looks like the following example:

```
match ma01 {
    client_ip("10.0.0.19")
    } then {
    flags = !spoof;
    servers = sv_01;
    }
```

In this example (the match rule is named "ma01"), the match function, `client_ip`, has an argument that matches all requests from IP address `10.0.0.19`, which are all sent to server sv_01. Additionally, this rule disables the `spoof` flag (that is, when the connection is made to the server, the server sees a connection to the Equalizer, not to the client). This rule looks as follow in the Administrative Interface:

Figure 48   Example match rule

The **If following expression matches** section of the screen shows the expression that is evaluated against the incoming request. If the expression evaluates to *true*, the **load balance with these settings** section specifies the servers that will be used to satisfy the incoming request, as well as the flags that will be set for the request. The next section of this document explains these settings in detail.

Managing Match Rules

The Administration Interface allows you to create and modify match rules, without requiring a detailed knowledge of the configuration language syntax used in the *eq.conf* file. The interface validates match rules before saving them so that all saved rules are syntactically correct. For this reason, we recommend you use the interface to create and edit match rules, rather than editing the configuration file.

The interface does *not*, however, test the behavior of match rules. Match rules must be tested against a flow of incoming requests in order to determine if the behavior of the rule is what you expect.

Before constructing a match rule, you should first understand the general concepts of match rules covered in "General Match Expressions and Match Bodies" on page 163.

The Match Rules Table

Click on a cluster name in the left frame and then click on the **Match Rules** tab to display a list of match rules defined for that cluster.
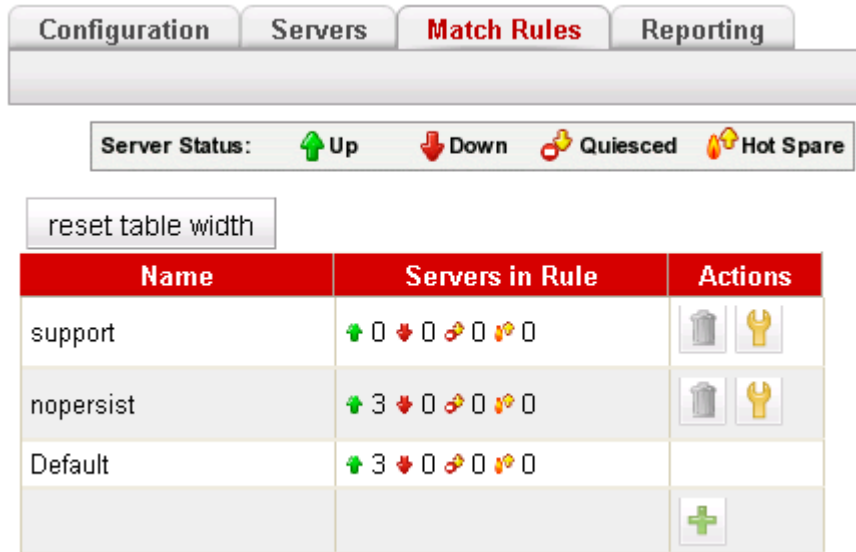
Figure 49   The match rules table

Name	The match rule name.
Server in Rule	Status indicators for all servers in the rule. Shows the number of servers in each of the following states: **Up** (responding to health check probes), **Down** (not responding to health check probes), **Quiesced** (not accepting new connections), and **Hot Spare** (only responding to requests when no other server is up).
Actions	Delete or Modify the match rule in the same row as the icon chosen. The Add icon at the bottom of the column opens the **Add New Match Rule** dialog.
reset table width	The columns on the table can be resized. If you extend a column too far to the right so that other columns are no longer visible, this button returns the table to its default proportions.

The Default Match Rule

All Layer 7 clusters created via the Equalizer Administration Interface start with a single match rule (named Default) that matches all requests and selects all servers.

```
match Default {
    any()
    } then {
    servers = all;
    }
```

The default rule specifies that all servers defined in the cluster should be used for load balancing the request, and that all flag settings for the request will be inherited from the cluster flag settings. This rule is always the last match rule in the ordered list of match rules for a cluster. You cannot modify, delete, or move this match rule.

The Default rule can be viewed by clicking in the left frame on **match Default** for any Layer 7 cluster. (If you have not created a Layer 7 cluster, see "Working with Virtual Clusters" on page 74). Figure 50 shows the default match rule for a cluster with two servers.

Figure 50   A Default match rule shown in the Match Rule dialog box

Note that although the Default match rule cannot be modified or deleted, it can be overridden. Do this by creating a new rule *immediately before* the Default that uses `any()` as the matching expression, so that the Default match rule is never processed. This effectively creates a new default match rule that you can configure with the desired load balancing options.

The following section shows you how to create a new Match Rule.

Creating a New Match Rule

To add a match rule to a virtual cluster, follow this general procedure:

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster (see "Logging In" on page 31).

2. In the left frame, right-click the name of the Layer 7 cluster to which you want to add a match rule, and select **Add Match Rule**.
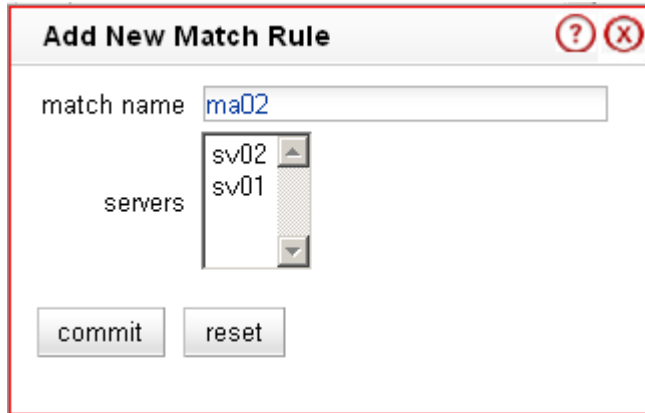
Figure 51 Example Add New Match Rule dialog box

3. Enter a name for the new rule in the **match name** field or accept the default. All match names within a cluster must be unique.

4. In the **servers** section, use the CTRL+left click and SHIFT+left click to select the names of the servers that you want to use to load balance requests that match the rule.

> **Caution –** If you do not enable a check box for at least one server, *Equalizer will drop the connection for any request that matches the rule*.

Select **commit** once you choose the servers for the match rule.

5. The Match Rule **Configuration** tab is displayed.

Figure 52   Match rule **Configuration** tab

The **Position rule:** field displays the name of the rule before which the currently displayed rule is evaluated. By default, a new rule is placed immediately before the Default rule. Change the placement of the new rule by choosing a rule from the **immediately before** list box. The evaluation order of the rules in a cluster is shown in the left frame.

The ordering of match rules is important, as they are processed from first to last until one of them evaluates to *true*, at which time the match body is processed. The initial match expression of a new rule, `any()` is one that will always evaluate to *true*, meaning that this match rule will always be selected. It is good practice to be cautious when adding new match rules to ensure that all the traffic to a cluster does not get mishandled. Use the **disable** flag (see Step 9) to skip a match rule that is still being developed.

6. Build your match rule expression in the **If following expression matches** section. To place or modify a match function, click the appropriate part of the expression. The part of the expression that the editor will directly affect is now displayed in a dialog box.

   a. From the drop-down list, select the match function and or expression with which you want to replace the selected part of the expression. Supply values for all arguments required by the function. To learn more about match functions, refer to "Match Functions" on page 171.

      The drop-down list of edit actions are different depending on what you select in the expression and whether the cluster is HTTP or HTTPS. All lists have some common match functions and structural editing operators. Some of the structural editing operators include the function you are replacing (for example, if you have selected the host() function, **replace with host AND any** will appear in the drop down box).

   b. Click the **continue** button. If there are any syntax errors, an error screen appears. This most likely occurs if there are missing arguments or syntax errors in the argument strings. Correct the error and click **continue**

again. If your changes are syntactically correct, Equalizer displays the new version of the match expression in the **Configuration** tab.

    c.   Repeat **a** and **b** until your expression is complete.

7.   The **load balance with these settings** section allows you to specify the following load balancing options for matching requests:

policy	Change these parameters to override the cluster setting. See "Modifying a Layer 7 Virtual Cluster" on page 75 for an explanation of these parameters.
compress mime-types	
cookie age	
cookie domain	
cookie path	
disable	Enable this flag to disable this match rule without deleting it. This can be useful when testing new match rules.
spoof	The two columns of check boxes to the right of these flags allow you to specify that the flag setting for a request that is selected by the match rule is either the same as the cluster setting, or overridden for this match.
once only	
abort server	The right-hand check box for each flag, if set, indicates that the flag setting will be inherited from the cluster setting -- in the screen above in Step 7, the **spoof** setting on the cluster (enabled) will be overridden for this match rule.
persist	
compress	
always	See "Modifying a Layer 7 Virtual Cluster" on page 75 for an

8.   Click **commit** to save the match rule definition.

Modifying a Match Rule

To edit a match rule, follow these steps:

1.   Log into the Administrative Interface using a login that has **write access** for the cluster (see "Logging In" on page 31).

2.   In the left frame, click the name of the match rule to be changed.

3.   Make the desired changes to the match rule, as shown in the procedure in the previous section, starting at Step 5 on page 169.

Removing a Match Rule

To delete a match rule, follow these steps:

1.   Log into the Administrative Interface using a login that has **add/del** access for the cluster (see "Logging In" on page 31).

2.   In the left frame, right-click the name of the match rule to be deleted and select **Delete Match Rule** from the local menu.

3.   Click **delete** to confirm that you want to delete the match rule.

Match Functions

To build or edit a match expression, click part of the expression to edit its arguments or to select a match function or logical expression from a dynamic drop-down list. The part of the expression that you click on is highlighted and determines the contents of the drop-down list. For instance, if the current selection is a match function, the arguments to the function are displayed so you can edit them, along with a list of items that can replace the function.

In the Administration Interface, logical operators and constructs are introduced using special entries in the drop-down list for expressions. These allow you to build complex boolean expressions in match rules. See the section "Logical Operators and Constructs in the GUI" on page 177.

The combination of match functions and logical operators provides a great deal of control over request processing based on the contents of the request's HTTP headers and the destination URI of the request.

The following table lists the non-URI functions supported by Equalizer match rules:

Table 53: non-URI Match Functions

non-URI Match Function	Description
any()	This function always evaluates to *true*.
client_ip(*string*)	This function evaluates to *true* only if the IP address of the client machine making the connection matches the *string* argument. The *string* can be a simple IP address (e.g., "192.168.1.110"), or an IP address in Classless Inter-Domain Routing (CIDR) notation (e.g., "192.168.1.0/24"). This function can be useful in restricting match expressions to a particular client or group of clients, which can aid in debugging a new match rule when a cluster is in production. Only the specified clients match the rule, leaving other clients to be handled by other match rules.
debug_message(*string*)	This function always evaluates to *true*. It writes the *string* argument to the Event Log for the cluster (**View > Event Log**). This function can be logically ANDed and ORed with other functions to write debug messages. *Use this function for testing and debugging only. Do not use it in production environments, since it has a negative impact on performance.*
ignore_case()	This function always evaluates to *true*, and is intended to be used to apply the **ignore_case** flag for comparisons when it is *not set* on the cluster. When this function is ANDed with other functions, it has the effect of forcing case to be ignored for any comparisons done by the match rule.

Table 53:   non-URI Match Functions

non-URI Match Function	Description
observe_case()	This function always evaluates to *true*, and is intended to be used to override the **ignore_case** flag for comparisons when it is *set* on a cluster. When this function is ANDed with other functions, it has the effect of forcing case to be honored for any comparisons done by the match rule.
http_09()	This function takes no arguments and evaluates to *true* if the HTTP protocol used by the request appears to be HTTP 0.9. This is done by inference: if an explicit protocol level is absent after the request URI, then the request is considered HTTP 0.9.
method(*string*)	This function evaluates to *true* if the *string* argument exactly matches the Request Method (e.g., GET, POST, etc.) specified in the request. Note that by default Equalizer forwards packets to servers without determining whether or not the method specified in the request is valid (i.e., is a method specified in Section 9 of RFC2616). One use of the **method**() function is to be able to override this default behavior and prevent invalid requests from being forwarded to a server.
[header match functions]	[No exact match **header()** function is supplied. See "Match Function Notes" on page 175, for the supported values for *header*.]
header_prefix(*header*, *string*)	This function evaluates to *true* if the selected *header* is present and if the string-valued argument *string* is a prefix of the associated header text.
header_suffix(*header*, *string*)	This function evaluates to *true* if the selected *header* is present and if the argument *string* is a suffix of the associated header text.
header_substr(*header*, *string*)	This function evaluates to *true* if the selected *header* is present and if the string-valued argument *string* is a sub-string of the associated header text.
header_regex(*header*, *string*)	This function evaluates to *true* if the selected *header* is present and if the string-valued argument *string*, interpreted as a regular expression, matches the associated header text.
ssl2()	HTTPS only. This function evaluates to *true* if the client negotiated the encrypted connection using SSL version 2.0.

Table 53:   non-URI Match Functions

non-URI Match Function	Description
ssl3()	HTTPS only. This function evaluates to *true* if the client negotiated the encrypted connection using SSL version 3.0.
tls1()	HTTPS only. This function evaluates to *true* if the client negotiated the encrypted connection using TLS version 1.0.

In addition to the functions in the preceding table, a set of functions is provided that allows you to process requests based on the various components of a request's destination URI.

A URI has the following parts (as defined in RFC1808):

```
<scheme>://<hostname>/<path>;<params>?<query>#<fragment>
```

In addition, Equalizer further breaks up the `<path>` component of the URI into the following components:

```
<directory><filename>
```

The following figure illustrates how Equalizer breaks up a URI into the supported components:
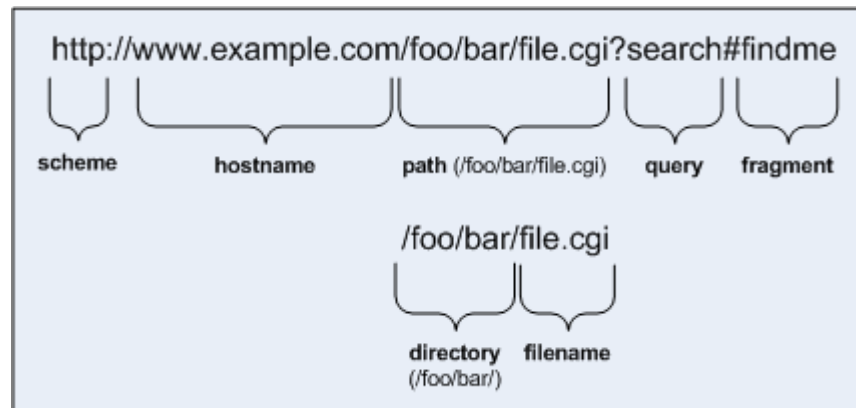
Figure 54   URI components

Note that the following components of the URI do not have corresponding match functions:

- Match functions for the `<scheme>` component are not necessary, since a cluster must be configured to accept only one protocol: HTTP *or* HTTPS.

- Match functions for the optional `<params>` component are not provided. Use the **pathname*()** and **filename*()** functions to match characters at the end of the **path** and **filename** components.

- Match functions for the optional `<fragment>` component are not provided. The fragment portion of a URI is not transmitted by the browser to the server, but is instead retained by the client and applied after the reply from the server is received.

The following table lists the URI matching functions that match text in the URI components shown in Figure 54.

URI Match Function	Description
host(*string*)	This function evaluates to *true* if the *string* argument exactly matches the hostname portion of the request. *In the case of HTTP 0.9, the host is a portion of the request URI. All other HTTP protocol versions require a Host header to specify the host, which would be compared to the string.*
host_prefix(*string*)	This function evaluates to *true* if the *string* argument is a prefix of the hostname portion of the URI path. The prefix of the hostname includes all text up to the first period ("www" in "www.example.com").
host_suffix(*string*)	This function evaluates to *true* if the *string* argument is a suffix of the hostname portion of the URI path. The suffix of the hostname includes all text after the first period in the hostname ("example.com" in "www.example.com").
pathname(*string*)	This function evaluates to *true* if the *string* argument exactly matches the path component of the request URI.
pathname_prefix(*string*)	This function evaluates to *true* if the *string* argument is a prefix of the path component of the request URI.
pathname_suffix(*string*)	This function evaluates to *true* if the *string* argument is a suffix of the path component of the request URI.
pathname_substr(*string*)	This function evaluates to *true* if the *string* argument is a substring of the path component of the request URI.
pathname_regex(*string*)	This function evaluates to *true* if the *string* argument, interpreted as a regular expression, matches the path component of the request URI.
dirname(*string*)	This function evaluates to *true* if the *string* argument exactly matches the directory portion of the path component of the request URI. The path component is the entire directory path, including the trailing slash (for example, "/foo/bar/" is the directory portion of "/foo/bar/file.html").
dirname_prefix(*string*)	This function evaluates to *true* if the *string* argument is a prefix of the directory portion of the path component of the request URI. The leading slash must be included in the *string* (for example, "/fo" is a prefix of "/foo/bar/file.html").
dirname_suffix(*string*)	This function evaluates to *true* if the *string* argument is a suffix of the directory portion of the path component of the request URI. The trailing slash must be included in the *string* (for example, "ar/" is a suffix of the directory portion of "/foo/bar/file.html").
dirname_substr(*string*)	This function evaluates to *true* if the *string* argument is a substring of the directory portion of the path component of the request URI.
dirname_regex(*string*)	This function evaluates to *true* if the *string* argument, interpreted as a regular expression, matches the directory portion of the path component of the request URI.

URI Match Function	Description
filename(*string*)	This function evaluates to *true* if the *string* argument exactly matches the filename portion of the URI path. *This portion includes only the text after the last trailing path component separator (/), as that is considered part of the directory* (for example, "file.html" is the filename portion of "/foo/bar/file.html").
filename_prefix(*string*)	This function evaluates to *true* if the *string* argument is a prefix of the filename portion of the URI path.
filename_suffix(*string*)	This function evaluates to *true* if the *string* argument is a suffix of the filename portion of the URI path.
filename_substr(*string*)	This function evaluates to *true* if the *string* argument is a substring of the filename portion of the URI path.
filename_regex(*string*)	This function evaluates to *true* if the *string* argument, interpreted as a regular expression, matches the filename portion of the URI path.
query(*string*)	This function evaluates to *true* if the *string* argument exactly matches the (optional) query component of the request URI. The query, if present, appears in a URI following a question mark (?). The syntax of a query is application specific, but generally is a sequence of key/value pairs separated by an ampersand (&).
query_prefix(*string*)	This function evaluates to *true* if the *string* argument is a prefix of the query portion of the URI path.
query_suffix(*string*)	This function evaluates to *true* if the *string* argument is a suffix of the query portion of the URI path.
query_substr(*string*)	This function evaluates to *true* if the *string* argument is a substring of the query portion of the URI path.
query_regex(*string*)	This function evaluates to *true* if the *string* argument, interpreted as a regular expression, matches the query portion of the URI path.

Match Function Notes

Please observe the notes in the following sections when constructing match rules.

Match Rule Behavior When Server Status is not 'Up'

When a match rule expression matches a client request, the request is load balanced using the servers, parameters, and flags specified in the match rule. The servers specified in the match rule may be in a number of "states" that affect the load balancing behavior: the server may be up or down, and may have one or both of the **quiesce** and **hot spare** options enabled.

server up	The request is routed to the selected server.
up/quiesce enabled	The request is routed to the selected server.
up/hot spare enabled	The request is routed to the selected server.
server down	If no Responder is selected in the match rule, then the request is sent to the selected server and, eventually, the client times out. If a Responder is selected, the Equalizer sends the configured response to the client.

The reason match rules behave as shown above is because the purpose of a match rule is to send a request that matches an expression to a particular server that can (presumably) better satisfy the request. In some cases, sending the request to a particular server may be required behavior for a particular configuration.

With this in mind, it does not make sense to skip a match rule because the server (or servers) named in the rule are down, hot spared, or quiesced -- rather, since the server in the rule is presumably critical to satisfying the request, it makes sense to route the request to the (for example) down server, and have the client receive an appropriate error -- so that the request can be retried.

If we instead were to skip a match rule because, for example, the server selected by the match rule is down, the request would be evaluated by the next match rule -- or the default match rule. The request, therefore, could potentially be sent to a server in the cluster that does not have the requested content. This means that the client would receive a "not found" error, instead of an error indicating that the appropriate server is not currently available.

Considering Case in String Comparisons

String comparisons performed by match functions honor the setting of the **ignore case** cluster parameter: if it is set on the cluster (the default), then all match rule functions used for that cluster are case insensitive; that is, the case of strings is ignored. For example, the string "ab" will match occurrences of "ab", "Ab", "aB", and "AB". If **ignore case** is *not* set on the cluster, then all string comparisons are by default case sensitive (the string "ab" will match only "ab").

To override the **ignore case** flag setting on the cluster for a match function or block of functions, you must logically AND the **observe_case()** or **ignore_case()** functions with the match function or block. For example, if **ignore case** is set on the cluster, you would use the following construct to force the **header_substr()** function to make case sensitive string comparisons:

```
(observe_case() AND header_substr("host", "MySystem"))
```

Regular Expressions

Some match functions have *prefix*, *suffix*, *substr*, or *regex* variants. The *regex* variants interpret an argument as a regular expression to match against requests. Regular expressions can be very costly to compute, so use the *prefix*, *suffix*, or *substr* variants of functions (or Boolean combinations of prefix and suffix testing), rather than the *regex* function variants, for best performance. For example, the following regular expression match:

```
dirname_regex("(two|four|six|eight)")
```

Can be replaced by the more efficient:

```
dirname_substr("two") OR

dirname_substr("four") OR

dirname_substr("six") OR

dirname_substr("eight")
```

Equalizer supports POSIX regular expression syntax only. See Appendix D, "Regular Expression Format" for a description.

Supported Headers

All of the **header_*(***header, string***)** match functions take a *header* argument, which selects the header of interest. If this header is not present in the request, the match function evaluates to *false*. Otherwise, the text associated with the header is examined depending on the particular function.

Although HTTP permits a header to span multiple request lines, none of the functions matches text on more than one line. In addition, Equalizer will only parse the first instance of a header. If, for example, a request has multiple **cookie** headers, Equalizer will only match against the first **cookie** header in the request.

The list of supported headers for the *header* argument are as follows:

Accept	From	Referer
Accept-Charset	Host	TE
Accept-Encoding	If-Match	Trailer
Accept-Language	If-Modified-Since	Transfer-Encoding
Authorization	If-None-Match	Upgrade
Cache-Control	If-Range	User-Agent
Connection	If-Unmodified-Since	Via
Content-Length	Max-Forwards	Warning
Cookie	Pragma	X-Forwarded-For
Date	Proxy-Authorization	
Expect	Range	

HTTPS Protocol Matching

Equalizer permits the construction of virtual clusters running the HTTPS protocol. HTTPS is HTTP running over an encrypted transport, typically SSL version 2.0 or 3.0 or TLS version 1.0. All of the functions available for load balancing HTTP clusters are available for HTTPS clusters. In addition, there are some additional match functions [`ssl2()`, `ssl3()`, and `tls1()`], that match against the protocol specified in an HTTPS request.

Supported Characters in URIs

The characters permitted in a URI are defined in RFC2396. Equalizer supports all characters defined in the standard for all Match Functions that have a URI as an argument. Note in particular that the ASCII space character is not permitted in URIs -- it is required to be encoded by all conforming browsers as "%20" (see Section 2.4 of RFC2396).

Logical Operators and Constructs in the GUI

In addition to the Match Functions listed in the previous section, the Equalizer Administrative Interface provides the following logical operators and constructs that allow you to combine the match functions into logical expressions, and manipulate the functions in the match expression. All of these operators and constructs affect the part of the match expression that is currently selected (highlighted in red) in the graphical interface.

negate function	This function negates (or reverses) the value of the expression that comes immediately after it in the match definition. When using the GUI to construct a match rule, choosing this function negates the currently selected function in the match rule expression and appears on screen as the string "NOT". In the *eq.conf* file, it negates the function immediately following it and appears as an exclamation point (!).
delete selection	Removes the currently selected portion of the match expression.
replace with AND	Replaces the currently selected logical operator with "AND".
replace with OR	Replaces the currently selected logical operator with "OR".

replace with any AND any	Replaces the currently selected logical construct with "`any() AND any()`".
replace with any OR any	Replaces the currently selected logical construct with "`any() OR any()`".
replace with self AND any	Replaces the currently selected logical construct with the current selection logically AND'ed with the "`any()`" function.
replace with self OR any	Replaces the currently selected logical construct with the current selection logically OR'ed with the "`any()`" function.
replace with any AND self	Replaces the currently selected function or logical construct with the "`any()`" function logically AND'ed with the current selection.
replace with any OR self	Replaces the currently selected function or logical construct with the "`any()`" function logically OR'ed with the current selection.
replace with any AND *function*	Replaces the currently selected function or logical construct with the "`any()`" function logically AND'ed with the current selection.
replace with any OR *function*	Replaces the currently selected *function* with the "`any()`" function logically OR'ed with the current selection.
replace with *function* AND any	Replaces the currently selected *function* with the current selection logically AND'ed with the "`any()`" function.
replace with *function* OR any	Replaces the currently selected *function* with the current selection logically OR'ed with the "`any()`" function.
swap left and right	When a logical operator is selected (i.e., AND or OR), switches the order of the left and right sides of the logical expression (e.g., "`A AND B`" becomes "`B AND A`").
replace with left	When a logical operator is selected (i.e., AND or OR), replaces the entire logical expression with the left side of the logical expression (e.g., "`A AND B`" becomes "`A`").
replace with right	When a logical operator is selected (i.e., AND or OR), replaces the entire logical expression with the left side of the logical expression (e.g., "`A AND B`" becomes "`B`").

Using Responders in Match Rules

Responders are used to send automated responses to clients when all the servers in a match rule are down. See the section "Automatic Cluster Responders" on page 109 for a complete description of Responders as well as examples of using Responders in Match Rules.

Example Match Rules

This section shows you how to create a few of the most commonly used types of match rules:

- "Parsing the URI" on page 179
- "Changing Parameters and Flags in Match Rules" on page 180
- "Dedicated Image and Content Servers" on page 184

Parsing the URI

In this example, we want to direct requests to a particular server based on the hostname used in the URI contained in the request. We want all requests for URIs that start with "support" to go to one server, and all other requests that do *not* match this rule to be load balanced across all servers in the cluster.

To do this, we will construct one match rule that parses the URI; if the URI contains the string "**support**", it forwards the request to the server **sv_support**. For this example, we assume that a cluster with four servers (**sv_support**, **sv01**, **sv02**, **sv03**) has already been defined.

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster (see "Logging In" on page 31).

2. In the left frame, right-click the name of the Layer 7 cluster to which you want to add the rule, and select Add Match Rule. The **Add Match Rule** dialog appears:

   a. Type **support** into the **match name** text box.

   b. Select **sv_support** in the servers list; make sure only this server is selected:

   c. Select **commit**.

The match rule is created, added to the object tree, and its **Configuration** tab is opened:

3. In the **If following expression matches** field, select **any()** to open the **Select function** dialog:

    a. Select **replace with host_prefix** from the drop-down box.

    b. Type "**support**" into the **hostname prefix** text box. The dialog should now look like this:
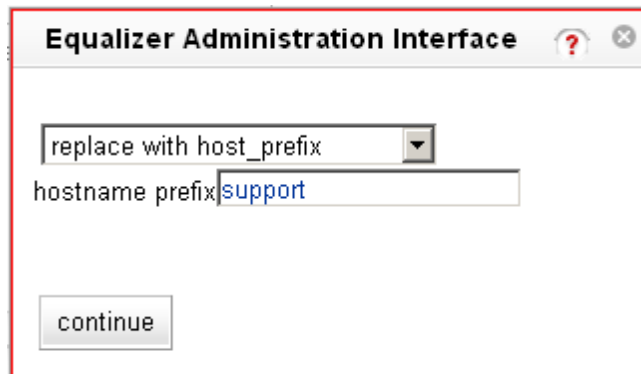
    c. Click **continue**.

4. Select the **commit** button to save your changes to the **support** rule.

Changing Parameters and Flags in Match Rules

By default, a client request that matches a match rule expression is load balanced using the same load balancing parameters and options that are currently set on the cluster. This section shows you how to change load balancing parameters and flags in a match rule.

For example, persistent connections to servers are enabled by the **persist** cluster flag, which is enabled by default when you create a cluster. While you could select the **dont persist** option for a server, this means that *all*

connections to that server will not be persistent. Let's assume that you only want to disable persistence for incoming requests that have a URI containing a hostname in the following format:

> *xxx*`.testexample.com`

We'll use the `host_suffix()` match rule function to test for the above hostname format. For this example, we assume that a cluster with three servers (**sv00**, **sv01**, **sv02**) has already been defined. We will construct a match rule that turn off **persist** for any request that contains the host suffix "**testexample.com**"; this request will be balanced across all three servers in the cluster.

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster (see "Logging In" on page 31).

2. In the left frame, right-click the name of the Layer 7 cluster to which you want to add the rule, and select **Add Match Rule**. The **Add Match Rule** dialog appears:

   a. Type **nopersist** into the **match name** text box.

   b. Select all the servers in the **servers** list using the **Ctrl** or **Shift** key and the left mouse button

   c. Select **commit**.

The match rule is created, added to the object tree, and its **Configuration** tab is opened:

3. In the **If following expression matches** field, select **any()** to open the **Select function** dialog:

    a. Select **replace with host_suffix** from the drop-down box.

    b. Type "**testexample.com**" into the **hostname suffix** text box. The dialog should now look like this:

    c. Click **continue**.

4. In the **load balance with these settings** field, disable both of the two check boxes to the right of the **persist** flag. The **Configuration** tab should now look like this:

5. Select the **commit** button to save your changes to the **nopersist** rule.

Dedicated Image and Content Servers

In this example, we want to direct all requests for images to a particular set of server, and balance the remainder of requests across the other servers in the cluster. The image servers are all connected to a common storage device that contains the images. The remaining servers are all dedicated to serving particular content for different web sites. For this example, we assume that a cluster with five servers as shown below has already been defined
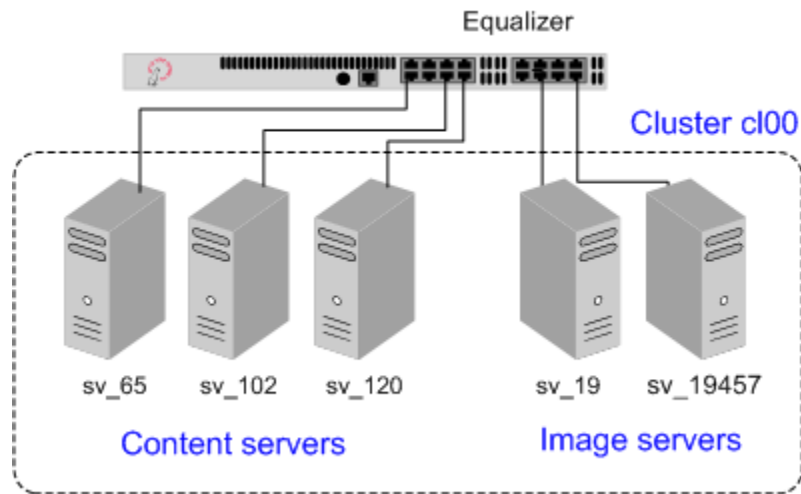
Figure 55 Match Rule Example: Dedicated Image and Content Servers

We want to maintain persistent connections for the web site servers, assuming that some of the websites may need to maintain sessions for applications such as shopping carts, email, etc. Persistent connections are not necessary for the image servers, since they access the images from common storage and have no need to maintain client sessions, so there is no need to incur the performance impact of maintaining session information.

To do this, we'll create two match rules, as follows:

1. Log into the Administrative Interface using a login that has **add/del** access for the cluster (see "Logging In" on page 31).

2. In the left frame, click the name of the Layer 7 cluster to which you want to add the rule. The cluster **Configuration** screen appears in the right frame:

   a. Make sure that the **once only** flag is not checked; otherwise, uncheck the **once only** flag and click **commit**.

   b. Open the **Persistence** tab and make sure the **persist** flag is not checked; otherwise, uncheck the **persist** flag and click **commit**.

   This is necessary because these flags, if enabled, cause only the first request in a connection to be evaluated. Since we want content to come from one set of servers and images from another, we want the servers that will have persistent connections to be chosen by the match rules.

3. Right-click the cluster name in the left frame and select **Add Match Rule**. The **Add Match Rule** dialog appears:

   a. Type **images** into the **match name** text box. In this match rule, we'll construct an expression that will match all the filename extensions of the images to be served. These requests will go to the image servers.

   b. In our example, we want all the images to be served from either **sv_19** or **sv_19457**. In the **servers** field, select **sv_19**, **sv_19457** using the **Ctrl** or **Shift**.

   c. Select **commit**.

   The match rule is created, added to the object tree, and its **Configuration** tab is opened:

4. In the **If following expression matches** field, click **any()** to open the **Select function** dialog:

   a. Select **replace with filename_suffix** from the drop-down box.

b. Type "**jpg**" into the **filename suffix** text box.

c. Select **continue**.

5. In the **If following expression matches** field, click **filename suffix("jpg")** to open the **Select function** dialog:

a. Select **replace with filename_suffix OR any()** from the drop-down box.

b. Select **continue**.

6. In the **If following expression matches** field, click **any()** to open the **Select function** dialog:

a. Select **replace with filename_suffix** from the drop-down box.

b. Type "**jpeg**" into the **filename suffix** text box.

c. Select **continue**.

7. Repeat Steps 5 and 6 for each of the other filename suffixes on our example servers -- **gif**, **bmp**, and **png**.

When you are done, the match expression should look like this:

```
─If following expression matches─────────
( filename_suffix(".jpg")
  OR filename_suffix("jpeg")
  OR filename_suffix("gif")
  OR filename_suffix("bmp")
  OR filename_suffix("png")
)
```

8. Select the **commit** button to save your changes to the **images** rule.

9. The **images** rule we created selects all the requests for image files; now we need a rule to determine which servers will receive all the other requests. The Default rule is not sufficient, and in fact we don't want it to be reached, since it could send a request for content to one of the image servers. So, we'll create another rule with the same match expression as the Default [**any()**], but a restricted list of servers. This effectively *replaces* the Default match rule with one of our own.

In the left frame, right-click the name of the cluster and select **Add Match Rule**. The **Add Match Rule** screen appears.:

a. Type "**content**" into the **match name** text box

b. In the **servers** field, select **sv_102**, **sv_65**, and **sv_120**.

c. Select **commit**.

The match rule is created, added to the object tree, and its **Configuration** tab is opened:

10. Select **Default** in the **immediately before** drop-down box.

11. Disable the right-hand check box for the **persist** flag; then, enable the left-hand check box next to **persist**. (Remember that in our example we're enabling **persist** for the content servers, so that persistent sessions can be maintained by the applications that run on these servers.) The **create match rule** screen should now look like this:

12. Select the **commit** button to save your changes to the **content** rule.

Chapter 8:

Administering GeoClusters

The Envoy geographic load balancer, an optional software add-on for the Equalizer product line, supports load balancing requests across servers in different physical locations or on different networks.

Overview of Geographic Load Balancing with Envoy

In non-Envoy Equalizer configurations, there is a one-to-one correspondence between a cluster and a website: when a client makes a request for a website (say, `www.example.com`), the client uses the Domain Name Service (DNS) to resolve the website name to an IP address. For a website that is load balanced by an Equalizer, the IP address returned is the IP address of an Equalizer cluster. After resolving the name, the client sends the request to the cluster IP. When Equalizer receives the client request, it load balances the request across the servers in the cluster, based on the current load balancing policy and parameters.

In an Envoy conversation, you have two or more Equalizers located in separate locations. Each Equalizer and its set of clusters and servers forms a *site (or Envoy site)*. With Envoy, the website name in the client request is resolved to a *GeoCluster IP*. A GeoCluster is analogous to a cluster, but one level above it: in other words, a GeoCluster actually points to two or more clusters that are defined on separate Equalizers.

In the same way that Equalizer balances requests for a cluster IP across the servers in the cluster, Equalizer load balances a request for a GeoCluster IP across the clusters in the GeoCluster configuration. Once a site is chosen and the client request arrives at that site, the request is load balanced across the servers in the appropriate cluster. In this way, you can set up geographically distant Equalizers to cooperatively load balance client requests.

Overview of Configuration Process

Follow this general procedure when setting up Envoy for the first time on two or more Equalizers running Version 8:

1. Configure appropriate clusters (and servers) on all of the Equalizers to be included as Envoy sites in the GeoCluster.

2. Configure the GeoCluster on each Equalizer; the parameters used should be the same on all sites.

3. Configure the authoritative DNS server for your website's domain with DNS records for all Equalizers in the GeoCluster. The DNS server returns these records to clients in response to DNS requests to resolve the website (GeoCluster) name.

> **Note –** While it is possible to mix Version 8 and Version 7 Equalizers in the same GeoCluster, we recommend that you run the same version of Equalizer software on all Equalizers in your GeoCluster. If you must run Version 8 and Version 7 Equalizers in an Envoy configuration, or if you are upgrading an existing Version 7 Envoy configuration to Version 8, see the section "Upgrading a Version 7 GeoCluster to Version 8" on page 194 for additional notes.

Overview of Envoy Site Selection

When a client uses DNS to resolve the address of a website name, it first contacts its local DNS server to resolve the name. The local DNS server then begins the process of resolving the website name by contacting other DNS servers to locate the authoritative DNS server for the website's domain. The authoritative DNS server for the website's domain returns a list of Envoy sites to the client's DNS server. Once it has this list, the client's local DNS server sends requests, one at a time, to each of the Envoy sites until it reaches an active site. (An overview of this process is given in the section "Distributing the Geographic Load" on page 7.) The local DNS server then waits for Envoy to resolve the website name into an IP address and return it.

Once an active Envoy site is reached, Envoy performs the following steps to determine the virtual cluster IP to return to the client's DNS:

1. Figure 56 is an illustration of a client in California whose local DNS server has contacted Envoy Site A to resolve the destination domain name for teh client request -- in this example, `www.coyotepoint.com`.
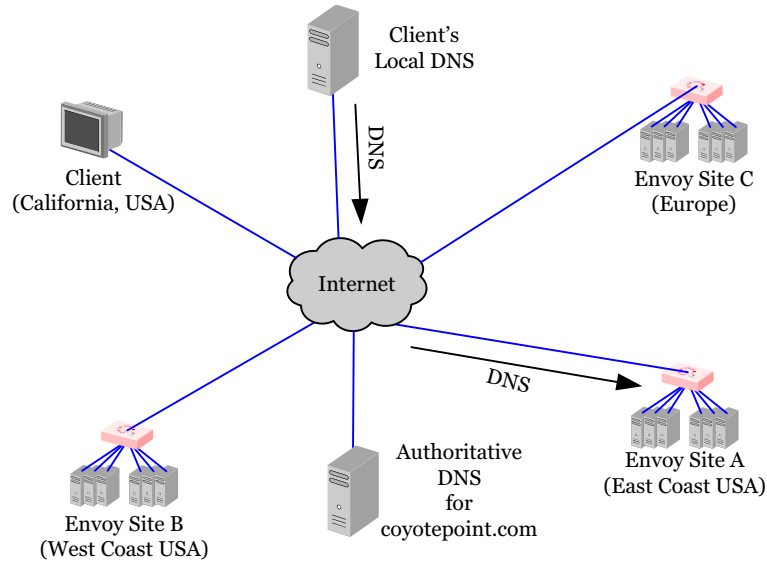

Figure 56 Sending name resolution requests to an Equalizer in a GeoCluster

2. Site A sends a *geographic query protocol probe (GQP)* to all the other Envoy sites in the GeoCluster for the requested domain (this GeoCluster has the same name -- **www.coyotepoint.com**). The probe is received by a special Envoy *agent* running at each site in the cluster (the agent for a site starts when you configure Envoy for the site). Site A also queries its local Envoy agent (see Figure 57).
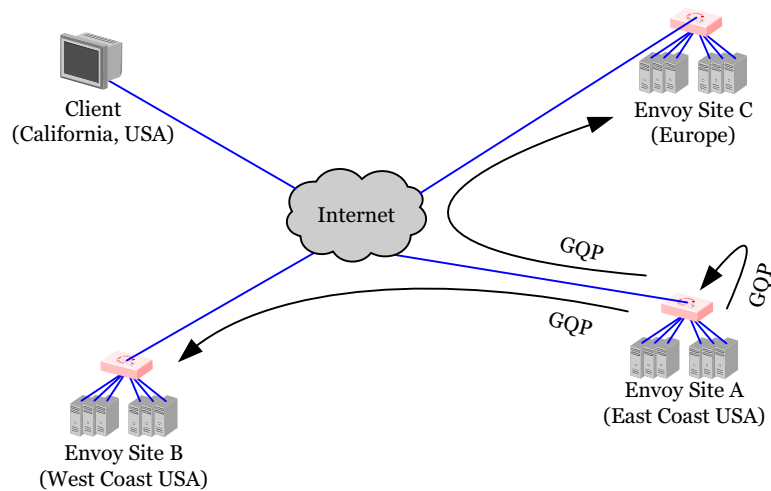
Figure 57 The selected Equalizer queries other Equalizers and its own servers in the GeoCluster

The GQP probes contain information about the requesting client and the local resource (i.e., local cluster) that is being requested by the client.

If **ICMP triangulation** is enabled, the GQP probes also tell the sites to send an ICMP echo request ('*ping*') to the client's local DNS server, and to return the DNS server's response time in their GQP response to Site A. This provides more accurate client location information to Envoy in the case where a resource is available at more than one site.

3. The Envoy agent at each site checks the availability of the requested resource and sends a GQP reply to the Envoy agent running at Site A (see Figure 58).
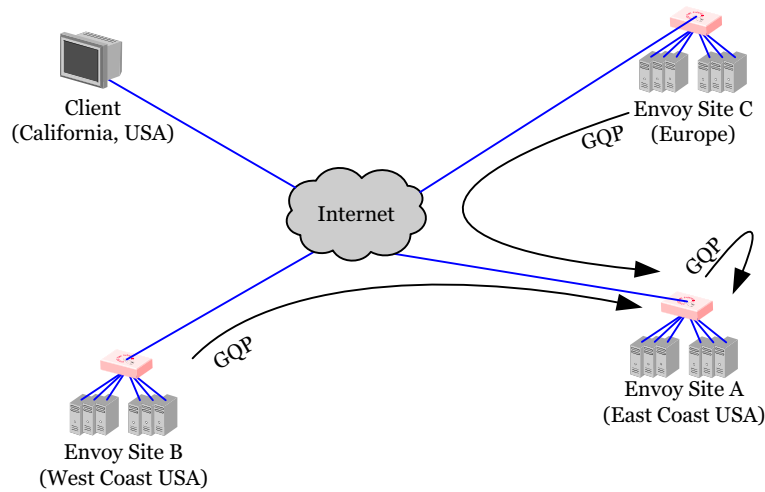

Figure 58 The selected Equalizer receives availability and triangulation (latency) information

The content of each response is as follows:

* If the resource (cluster) is not available at the site, the site sends an error message.

* If the resource (cluster) is available at the site, the site sends a message that the resource is available.

* If **ICMP triangulation** is enabled for the GeoCluster, and the requested resource is available at the site, the site will include the local DNS server's ICMP echo response time (or lack thereof) in its GQP response to Site A.

4. After all GQP responses are received (or the GQP probes time out), Site A determine the 'best available' site to return to the client's DNS server using this process:

   a. If at least one GQP probe is received from a site at which the resource is available, then Envoy uses the resource availability and network latency information (if present) in the GQP replies to select a site for the client based on the current load balancing policy, and returns the address of the 'best available' Envoy site to the requesting client's local DNS (see Figure 59).
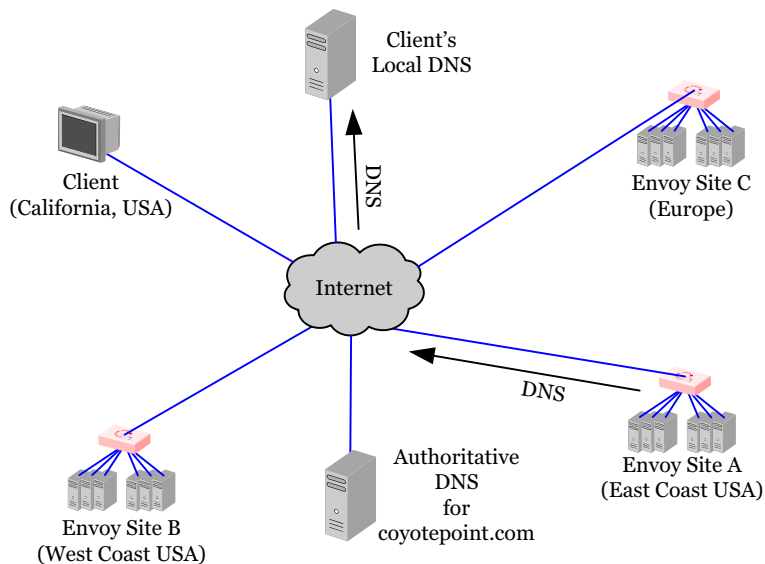

Figure 59 The client's local DNS receives the best Equalizer site

b. If no GQP responses are received, or if the requested resource (cluster) is not available at any of the sites that replied, then Site A returns a site to the client's DNS according to this algorithm:

```
If the site that has the default option enabled is up
   Then, send the IP address of the resource at this site (even if weight=0)
Else, if one or more sites are up
   Then, send the IP address of the resource at any site marked up
Else, if a site is marked as the default
   Then, send the IP address of the resource at that site, even if the site
   is marked down
Else, send a NULL response back to the client's DNS server.
```

5. Once the client's local DNS server sends the client the IP address of the selected site, the client sends the request to the site (see Figure 60). The site then responds to the client and the connection is thereafter managed by the chosen site (in our example, Site B).
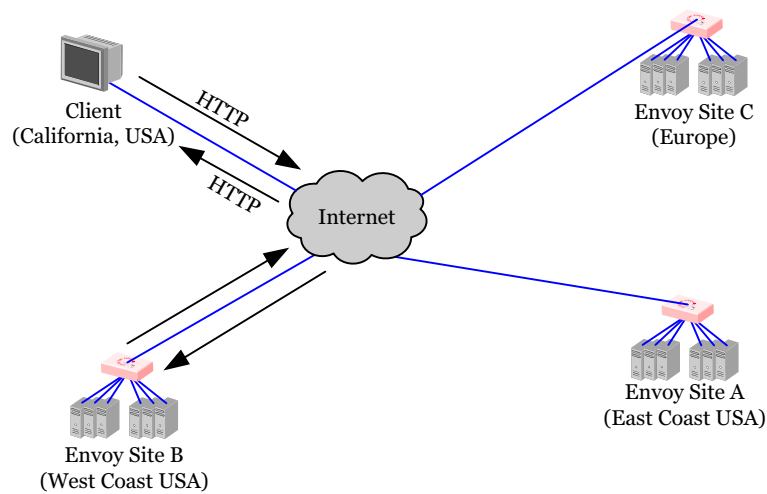
Figure 60   Site B handles the client's connection

Licensing and Configuring Envoy

Each site in an Envoy GeoCluster must have an Equalizer that is running Envoy, which must be licensed in order to run. Envoy software is pre-installed on each Equalizer and is enabled through the registration and licensing process.

After you have licensed Envoy and completed Envoy and DNS configuration described in this section, you can set up GeoClusters and define the available sites for each cluster.

Enabling Envoy

To license and enable Envoy, follow these steps:

1. Log into the Equalizer Administration Interface, and expand the **Equalizer System Information** box in the right frame:

   - If the line **Envoy geographic load balancing** shows that Envoy is **enabled**, stop now; Envoy is already licensed.

   - If the line **Envoy geographic load balancing** shows that Envoy is **disabled**, go to the next step.

1. Follow the registration procedure and make sure that you enter the serial number for your Envoy software on the registration website; see "Licensing Equalizer" on page 44 in Chapter 4, "Configuring Equalizer Operation".

2. Shut down the Equalizer and reboot the machine; see "Rebooting Equalizer" on page 65 in Chapter 4, "Configuring Equalizer Operation".

3. After the system reboots, confirm that Envoy is enabled. Log into the Equalizer Administration Interface and expand the **Equalizer System Information** box in the right frame. The line **Envoy geographic load balancing** should indicate that Envoy is **enabled**.

Configuring the Authoritative Name Server to Query Envoy

You must configure the authoritative name server(s) for the domains that are to be geographically load balanced to delegate authority to the Envoy sites. You need to delegate each of the fully-qualified subdomains to be balanced. If your DNS server is run by an Internet Service Provider (ISP), then you need to ask the ISP to reconfigure the DNS server for Envoy. If you are running your own local DNS server, then you need to update the DNS server's *zone file* for your Envoy configuration.

For example (see Figure 61), assume you must balance `www.coyotepoint.com` across a GeoCluster containing two Envoy sites, `east.coyotepoint.com` (at `192.168.2.44`) and `west.coyotepoint.com` (at `10.0.0.5`). In this case, you must configure the name servers that will handle the `coyotepoint.com` domain to delegate authority for `www.coyotepoint.com` to both `east.coyotepoint.com` and `west.coyotepoint.com`. When queried to resolve `www.coyotepoint.com`, `coyotepoint.com`'s name servers should return name server (NS) and alias (A) records for both Envoy sites.
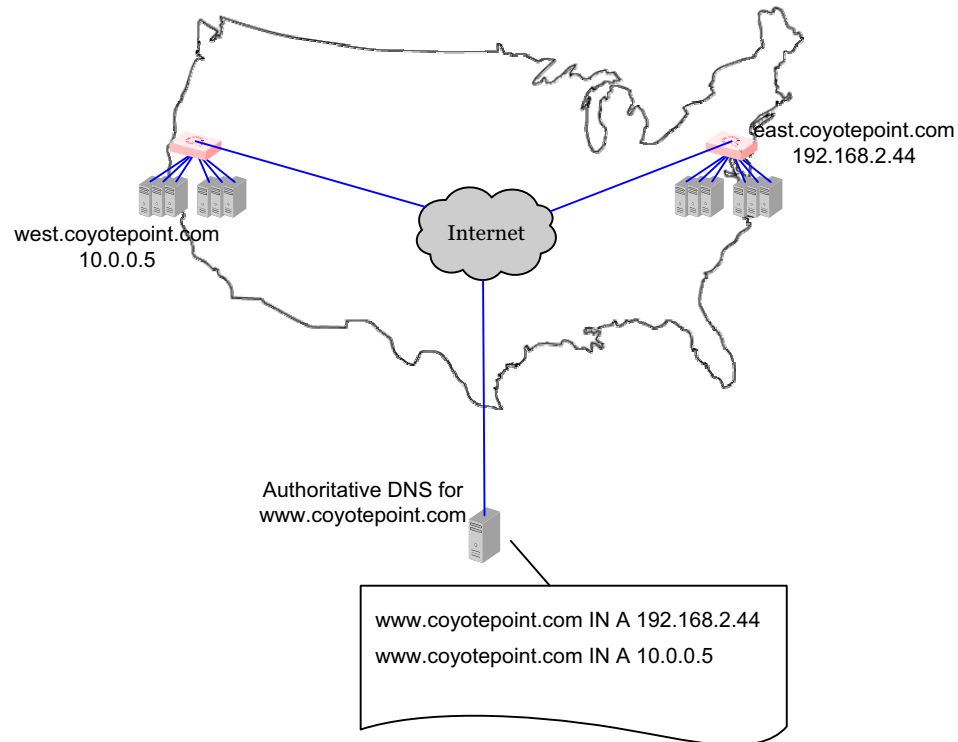
Figure 61 Two-site DNS example

An example of a DNS zone file for this configuration is shown below. In this example, the systems `ns1` and `ns2` are assumed to be the authoritative name servers (master and slave) for the `coyotepoint.com` domain.

```
$TTL 86400

coyotepoint.com. IN SOA ns1.coyotepoint.com. hostmaster.coyotepoint.com. (
                        0000000000
                        00000
                        0000
                        000000
                        00000 )
coyotepoint.com.     IN NS ns1.coyotepoint.com.
coyotepoint.com.     IN NS ns2.coyotepoint.com.
www.coyotepoint.com.  IN NS east.coyotepoint.com.
www.coyotepoint.com.  IN NS west.coyotepoint.com.
ns1   IN A ns1-IP-address
ns2   IN A ns2-IP-address
east  IN A 192.168.2.44
west  IN A 10.0.0.5
```

Figure 62 Example DNS Zone File

In the example above, we left the domain parameters as zeros, since these vary widely between DNS installations. Please see the documentation for the version of DNS that you are using for more information on the zone file content and format.

To ensure that you have properly configured DNS for Envoy, you can use the **nslookup** command (supported on most OS platforms) to confirm that the DNS server is returning appropriate records, as in this example:

```
nslookup www.coyotepoint.com
Server:  ns1.coyotepoint.com
Address:  ns1-IP-address

Name:    www.coyotepoint.com
Address:  192.168.2.44
```

Using Envoy with Firewalled Networks

Envoy sites communicate with each other using Coyote Point's UDP-based Geographic Query Protocol (GQP). Similarly, Envoy sites communicate with clients using the DNS protocol. If you protect one or more of your Envoy sites with a network firewall, you must configure the firewall to permit the Envoy packets to pass through.

To use Envoy with firewalled networks, you need to configure the firewalls so that the following actions occur:

- Envoy sites communicate with each other on UDP ports 5300 and 5301. The firewall must allow traffic on these ports to pass between Equalizer/Envoy sites.

- Envoy sites and clients can exchange packets on UDP port 53. The firewall must allow traffic on this port to flow freely between an Envoy site and any Internet clients so that clients trying to resolve hostnames via the Envoy DNS server can exchange packets with the Envoy sites.

- Envoy sites can send ICMP echo request packets out through the firewall and receive ICMP echo response packets from clients outside the firewall. When a client attempts a DNS resolution, Envoy sites send an ICMP echo request (ping) packet to the client and the client might respond with an ICMP echo response packet.

Using Envoy with NAT Devices

If an Envoy site is located behind a device (such as a firewall) that is performing Network Address Translation (NAT) on incoming IP addresses, then you must specify the public (non-translated) IP as the Site IP, and use the translated IP (the non-public IP) as the resource (cluster) IP in the Envoy configuration.

This is because Envoy must return the public cluster IP to a requesting client in order for the client to be able to contact that cluster -- since the request goes through the NAT device before it reaches Equalizer. The NAT device translates the public cluster IP in the request to the non-public cluster IP that is defined on Equalizer, and then forwards the packet to Equalizer.

The non-public cluster IP must still be specified as the resource IP for the site, as this is the IP that Envoy will use internally to probe the availability of the resource (cluster) on the site.

Upgrading a Version 7 GeoCluster to Version 8

Envoy in Version 8 is designed to work with existing sites running Version 7. You can upgrade a Version 7 site in-place to Version 8, and it will continue to operate seamlessly with other Version 7 sites in the GeoCluster. In order to work with resources located on other Version 8 sites, however, the configuration must be updated with the cluster name, as noted below:

1. Upgrade sites one at a time, starting with the non-default sites. Test thoroughly before upgrading the next site.

2. The resource (cluster) name for any resource that is located on a site running Version 7 of the Equalizer software must be left blank. Specify the cluster IP and port instead.

3. The resource (cluster) IP and port for any resource that is located on a site running Version 8 of the Equalizer software must be left blank. Specify the cluster name instead.

Working with GeoClusters

This section shows you how to add or delete a GeoCluster and how to configure a GeoCluster's load-balancing options. Configuring a GeoCluster and its sites is analogous to configuring a virtual cluster and its servers.

When Envoy is first enabled, there are no GeoClusters defined, so clicking on the **Envoy** icon in the left frame displays a blank GeoCluster Summary table. Below is an example of a GeoCluster Summary table with one GeoCluster and two GeoSites configured:

Figure 63   GeoCluster Summary

The table shows the GeoCluster name and the status of the sites in the cluster. The icons in the Actions column let you add, modify, and delete GeoClusters.

To see an exapandable list of existing GeoClusters, click **Envoy > Status**:

Adding a GeoCluster

To add a new GeoCluster, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for Global Parameters (see "Logging In" on page 31).

2. Do *one* of the following:
   - In the left-frame object tree, right-click on **Envoy** and select **Add GeoCluster** from the menu.
   - Click **Envoy** in the left frame and then click the Add icon in the GeoCluster Summary table.

3. The following dialog appears:

4. Enter the following information:

FQDN name	Enter the **GeoCluster name**, which is the fully-qualified domain name (FQDN) of the GeoCluster (for example, `www.coyotepoint.com`). The FQDN must include all name components up to the top level (com, net, org, etc). Do not include the trailing period.
DNS ttl	The cache time-to-live, which is the length of time (in seconds) that the client's DNS server should cache the resolved IP address. Longer times will result in increased failover times in the event of a site failure, but are more efficient in terms of network resources. The default is 120 (that is, 2 minutes).

5. Click the **commit** button to add the GeoCluster. An entry for the new GeoCluster appears in the left frame. The right frame displays the GeoCluster Configuration screen.

6. Continue with Step 3 in the next section to change the default GeoCluster parameters.

Viewing and Modifying GeoCluster Parameters

To view or modify a GeoCluster's load-balancing options, follow these steps:

1. Log into the Administrative Interface using a login that has **read** (to view only) or **write** (to view or change) permission on the GeoCluster (see "Logging In" on page 31).

2. Do *one* of the following:
    *   Click on the GeoCluster name.
    *   Click **Envoy** in the left frame and then click the Modify icon [icon] in the GeoCluster Summary table row for the GeoCluster you want to modify.

3. The GeoCluster Configuration tab is displayed:

The **default site warning** is displayed on this screen until you select a default site; see "Displaying and Modifying Site Information" on page 201.

The GeoCluster configuration parameters are explained in the table below:

responsiveness	This value controls how aggressively Equalizer adjusts the site's dynamic weights. Equalizer provides five response settings: **slowest**, **slow**, **medium**, **fast**, and **fastest**. Faster settings enable Equalizer to adjust its load balancing criteria more frequently and permit a greater variance in the relative weights assigned to sites. Slower settings cause site measurements to be averaged over a longer period of time before Equalizer applies them to the cluster-wide load balancing; slower settings also tend to ignore spikes in cluster measurements caused by intermittent network glitches. We recommend that you select the *medium* setting as a starting point.
DNS cache ttl	The cache time-to-live, which is the length of time (in seconds) that the client's DNS server should cache the resolved IP address. Longer times will result in increased failover times in the event of a site failure, but are more efficient in terms of network resources. The default is 120 (that is, 2 minutes).
MX exchanger	The fully qualified domain name (e.g., 'mail.example.com') to be returned if Equalizer receives a "mail exchanger" request for this GeoCluster. The mail exchanger is the host responsible for handling email sent to users in the domain. This field is not required.

policy	Three basic metrics are used by the policy to load balance requests among sites: the current load on the site, the initial weight setting of the site, and ICMP triangulation responses. The policy setting tells Envoy the realtive weight to assign to each metric when choosing a site. **round trip** weights the ICMP triangulation information received from each site more heavily than other criteria. **adaptive** give roughly equal weights to the site load and ICMP triangulation responses, and gives less weight to the initial weight for the site. This is the default setting. **site load** weights the current load at each site more heavily than other criteria. **site weight** weights the user-defined initial weight for each site more heavily than other criteria. **Note**: For all policies, the current site load metric is ignored for the first 10 minutes that the site is up, so that the metric value is a meaningful measure of the site load before it is used.
ICMP triangulation	When a request for name resolution is received by Envoy from a client's local DNS, this option (if enabled) tells Envoy to request network latency information from all sites in order to make load balancing decisions based on the proximity of each site to the client's DNS server. To do this, all Envoy sites send an ICMP echo request ('ping') to the client's DNS server. The reply from the DNS server allows Equalizer to select a site using the length of time is takes for the DNS server's reply to reach the site. (Consequently, this method assumes that the client's DNS server is geographically close to the client -- which is usually the case.) In order for triangulation data to be collected at each site, responding to ICMP echo requests must be enabled on the client DNS server and the DNS server must be allowed to respond through any firewalls between it and the Envoy sites. If you do not want Envoy sites to ping client DNS servers, disable this option (this is the default setting).

4. Click the **commit** button to save any changes you made to the GeoCluster parameters.

Deleting a GeoCluster

To delete a GeoCluster, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for the GeoCluster (see "Logging In" on page 31).

2. Do *one* of the following:

   - In the left frame, right-click the name of the GeoCluster to delete and select **Delete GeoCluster** from the menu.

   - Click **Envoy** in the left frame and then click the Delete icon 🗑 in the GeoCluster Summary table row for the GeoCluster you want to delete.

3. When prompted, click **delete** to confirm removing the cluster. Equalizer deletes the GeoCluster and all its sites.

Displaying Envoy Statistics

See "Displaying Envoy Statistics" on page 143.

Plotting GeoCluster History

See "Plotting GeoCluster Performance History" on page 148.

Working with Sites

GeoSites, or Sites, are defined within GeoClusters, so before you can configure your first site, you must first have added a site as shown in the section "Adding a GeoCluster" on page 195.

Once you define a GeoCluster, you can open the GeoSite Summary table to add, modify, and delete GeoSites: click the GeoCluster name in the left frame, and then click the Sites tab in the right frame. The following is an example of a GeoSite Summary table with two sites configured:

Add, modify, and delete Sites

Geo Site Name	AAA IP	Agent IP	Status	Actions
si00	172.16.0.81	172.16.0.80	⬆	🗑 🔧
si01	172.16.0.142	172.16.0.140	⬆	🗑 🔧
				➕

The table shows the GeoSite Name, the DNS AAA Record IP address, and the Agent IP address supplied when the sites were created. It also displays the current Status of each Site. The icons in the Actions column let you add, modify, and delete GeoSites.

Adding a Site to a GeoCluster

1. Log into the Administrative Interface using a login that has **add/del** access for the GeoCluster (see "Logging In" on page 31).

2. Do *one* of the following:
   - In the left frame, right-click the name of the GeoCluster to which you want to add a Site, and select **Add Site** from the menu.
   - Click the GeoCluster name in the left frame and open the **Sites** tab. Click the Add icon ➕ at the bottom of the GeoCluster Summary table **Action** column.

3. The following dialog is displayed:

4. The GeoSite Parameters are described in the following table:

Site Name	A symbolic name that represents this site. For example, the east coast site for `www.coyotepoint.com` might be `eastCOAST`.
A Record IP Address	The IP address returned by DNS to a client when the GeoCluster is accessed. For example, when a client open www.coyotepoint.com, the local DNS server returns an A record that contains the IP address for www.coyotepoint.com. This is usually the address of an Equalizer cluster and in this case is also used as the resource IP. However, the site's A record IP may be different from the cluster (resource) IP if the A record IP address is NAT'ed to an internal address (the actual cluster IP). In this case, you specify the A record IP as the site IP and the cluster IP as the resource IP.
Agent IP Address	The IP address of the site monitoring agent. This is the external interface address of the Equalizer at this site; if the Equalizer is in single network mode, this is the internal interface address.
Site Version	Click **V8** if the Site is running Version 8 or later of the Equalizer software. Enter a **Resource Name** that is same as the name of a cluster configured on the Equalizer at this site.
	Click **V7** if the Site is running Version 7 of the Equalizer software. Enter an **IP Address** and **Port** for a cluster configured on the Equalizer at this site.

5. Click the **commit** button to add the Site. An entry for the new Site appears in the left frame. The right frame displays the Site Configuration screen. Continue with the next section to update the default Site and Resource parameters.

Displaying and Modifying Site Information

To view or modify the information for a particular GeoSite, follow these steps:

1. Log into the Administrative Interface using a login that has **read** (to view only) or **write** (to view or change) permission on the Site's GeoCluster (see "Logging In" on page 31).

2. Do *one* of the following:
    - In the left-frame object tree, expand the Envoy tree until the name of the Site is visible. Click the Site name.
    - Click the GeoCluster name in the left frame and open the **Sites** tab. Click the Modify icon 🔧 on the row for the GeoSite you want to modify.

The Site Configuration tab is displayed.

The **Site Configuration** parameters are explained in the table below:

ip	The IP address returned by DNS when the GeoCluster is accessed. For example, when a client open www.coyotepoint.com, the local DNS server returns an A record that contains the IP address for www.coyotepoint.com. This is usually the address of an Equalizer cluster and in this case is also used as the resource IP. However, the site's A record IP may be different from the cluster (resource) IP if the A record IP address is NAT'ed to an internal address (the actual cluster IP). In this case, you specify the A record IP as the site IP and the cluster IP as the resource IP.
agent	The IP address of the site monitoring agent. This is the external interface address of the Equalizer at this site; if the Equalizer is in single network mode, this is the internal interface address.

weight	An integer that represents the site's capacity. (This value is similar to a server's initial weight.) Valid values range between 10 and 200. Use the default of 100 if all sites are configured similarly; otherwise, adjust higher or lower for sites that have more or less capacity.
	Equalizer uses a site's initial weight as the starting point for determining what percentage of requests to route to that site. Equalizer assigns sites with a higher initial weight a higher percentage of the load. The *relative* values of site initial weights are more important than the actual values. For example, if two sites are in a GeoCluster and one has roughly twice the capacity of the other, setting the initial weights to 50 and 100 is equivalent to setting the initial weights to 100 and 200.
	Dynamic site weights can vary from 50% to 150% of the assigned initial weights. To optimize GeoCluster performance, you might need to adjust the initial weights of the sites in the cluster based on their performance.
	Site weights can range from 10 to 200. When you set up sites in a GeoCluster, you should set each site's initial weight value in proportion to its capacity for handling requests. It is not necessary for all of the initial weights in a cluster to add up to any particular number.
default site	Designates this site as the default site for the GeoCluster. Envoy load balances to the default site whenever it cannot choose a site based on the GQP probe information it gets from the sites. This can happen, for example, when GQP probe responses are not received from any site, when the resource (cluster) is down at all available sites, etc. If no default site is selected for a GeoCluster and all sites are down, then Envoy sends a null response to the client DNS. See the explanation in Step 4b on page 191.

3. Click the **commit** button to save any changes you made to the Site configuration.

4. Click on the **Resources** tab, to update the following resource parameters:

If **V8** is selected, the following parameter is displayed:

name	If the Equalizer at this site is running Version 8 or higher of the Equalizer software, specify the cluster name. Equalizer will query the Envoy agent at that site for the cluster's IP address and port. Leave blank if the site is running Version 7.

If **V7** is selected, the following parameters are displayed:

ip	If the Equalizer at the site is running Version 7 of the Equalizer software, specify the cluster's IP address (and port, below). It is generally the same value as the site IP address, unless the site address is NAT'ed to a cluster IP. Leave blank if the site is running Version 8.

port	If the Equalizer at the site is running Version 7 of the Equalizer software, specify the cluster's TCP port number (and IP address, above). Leave blank if the site is running Version 8.

5. Click the **commit** button to save any changes you made to the resource configuration.

Deleting a Site from a GeoCluster

To delete a Site from a GeoCluster, follow these steps:

1. Log into the Administrative Interface using a login that has **add/del** access for the GeoCluster (see "Logging In" on page 31).

2. Do *one* of the following:
   - In the left frame, right-click the name of the Site to delete and select **Delete Site** from the menu. (You may need to expand the GeoCluster first to see the Sites.)
   - Click the GeoCluster name in the left frame and open the **Sites** tab. Click the Delete icon on the row for the GeoSite you want to delete.

3. When prompted, click **delete** to confirm removing the Site. Equalizer deletes the Site and removes it from the object tree.
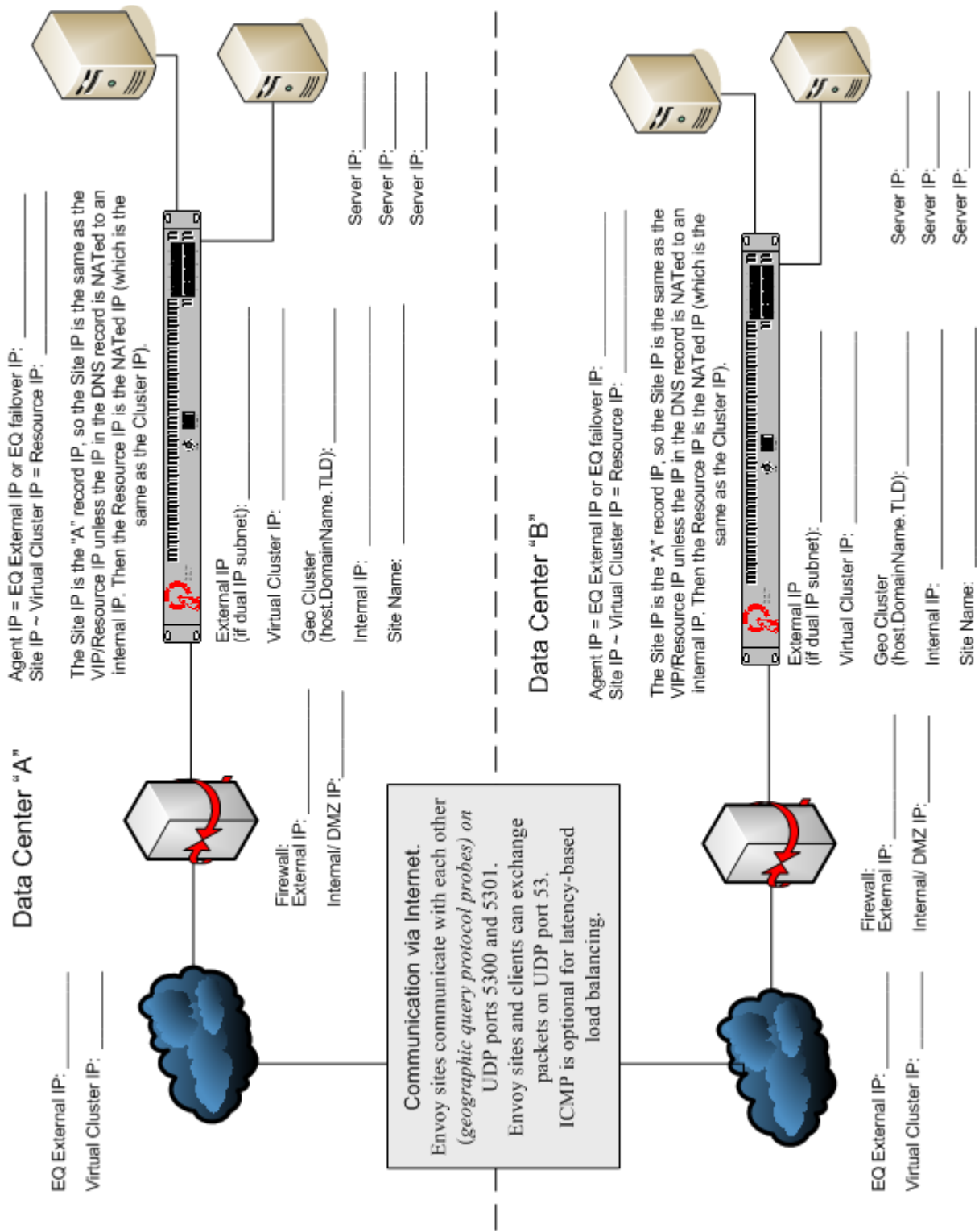
Displaying Site Statistics

See "Displaying Site Statistics" on page 144.

Plotting Site History

See "Plotting Site Performance History" on page 149.

Envoy Configuration Worksheet

Data Center "A"

Agent IP = EQ External IP or EQ failover IP: _____
Site IP ~ Virtual Cluster IP = Resource IP: _____

The Site IP is the "A" record IP, so the Site IP is the same as the VIP/Resource IP unless the IP in the DNS record is NATed to an internal IP. Then the Resource IP is the NATed IP (which is the same as the Cluster IP).

External IP
(if dual IP subnet): _____

Virtual Cluster IP: _____

Geo Cluster
(host.DomainName.TLD): _____

Internal IP: _____

Site Name: _____

Server IP: _____
Server IP: _____
Server IP: _____

Firewall:
External IP: _____
Internal/ DMZ IP: _____

EQ External IP: _____
Virtual Cluster IP: _____

Communication via Internet.
Envoy sites communicate with each other *(geographic query protocol probes) on* UDP ports 5300 and 5301.
Envoy sites and clients can exchange packets on UDP port 53.
ICMP is optional for latency~based load balancing.

Data Center "B"

Agent IP = EQ External IP or EQ failover IP: _____
Site IP ~ Virtual Cluster IP = Resource IP: _____

The Site IP is the "A" record IP, so the Site IP is the same as the VIP/Resource IP unless the IP in the DNS record is NATed to an internal IP. Then the Resource IP is the NATed IP (which is the same as the Cluster IP).

External IP
(if dual IP subnet): _____

Virtual Cluster IP: _____

Geo Cluster
(host.DomainName.TLD): _____

Internal IP: _____

Site Name: _____

Server IP: _____
Server IP: _____
Server IP: _____

Firewall:
External IP: _____
Internal/ DMZ IP: _____

EQ External IP: _____
Virtual Cluster IP: _____

Appendix A:

Server Agent Probes

A server agent is a custom written program that runs on a server and provides direct feedback to Equalizer that is used by the load balancing algorithms. This feedback is ontained by the agent by any means available on the target server; the only requirement from Equalizer's point of view is that the agent response is in the form of an integer between -2 and 100 that represents the status of the server and/or the application that is runningon it.

100 to 0	100 indicates that the server and/or application is lightly loaded.
	0 indicates that the server and/or application is heavily loaded.
-1	The application, or a required resource (such as a database), is unavailable.
-2	The server agent cannot determine the status of the application. This is the default return value used by Equalizer when an agent does not respond.

Note: Server agent code written prior to Version 8.1.0a must be adjusted to reflect the server agent return values and interpretations shown above. In particular, in previous releases the meanings of the 0 to 100 range of values were documented as the reverse of the meanings shown above.

Note that there is no return code for 'server down' -- Equalizer relies on the other health check probes to determine whether the server is up (ICMP, TCP, and ACV probes). If you want Equalizer to regard a server as down if there is no response from the server's agent, then enable the **require agent response** flag; see "Modifying Global Parameters" on page 47.

After returning a value to Equalizer, the agent closes the port and waits for another connection.

You configure server agents on a cluster-wide basis—all the servers in a virtual cluster must be running agents for server agents to be used for adaptive load balancing. When you have enabled server agents, Equalizer periodically probes the agent at each server's IP address through the configured agent port. Equalizer uses the collected server agent values when performing adaptive load balancing calculations.

Agents work with all load balancing policies (see "Equalizer's Load Balancing Policies" on page 86), except for **round robin** (which simply ignores any agent defined for the cluster). All the other policies use the integer returned by the agent as one factor in determining the server to which a new request is sent.

The **server agent** policy gives primary importance to the value returned by an agent over other load balancing factors (server weight, number of current connections, etc.).

Enabling Agents

Agents are enabled for a cluster by turning on the **server agent** cluster flag. The default **agent port** is **1510**. Make sure that any agent you deploy is listening and able to respond to TCP connections on the same port number on all the servers in the cluster.

The time between server agent probes is determined by the **agent delay** global parameter (default is 10 seconds).

Equalizer will open up a connection to the server agent's IP/port, and wait for a response. If no response is received, then the Equalizer performs load balancing without the server agent value for that server.

Some agents, particularly those written in Java, may require that a string be sent to the agent before a response is sent back to Equalizer. The **agent probe** field is provided for this purpose. If a string appears in this field, it is sent to the agent when an agent probe occurs.

Sample Agent in Perl

You can write custom agents as shell scripts, or in Java, Perl, C, or other languages. The code below is a simple server agent example written in Perl. This code prompts for a constant value when the server agent program is started, and returns that value when a connection is made on port 1510 (configurable via the `$port` variable).

```perl
#!/usr/bin/perl -w
# serveragent.pl
#--------------------
#(c) Copyright 2008 Coyote Point Systems, Inc.

use strict;
use Socket;

# use port 1510 as default
my $port = 1510;
my $proto = getprotobyname('tcp');

# take the server agent response value from the command line
my $response = shift;

# response has to be a valid server agent response
$response==-1 or ($response > 0 and $response<101)
or die "Response must be between -1 and 100";

# create a socket and set the options, set up listen port
socket(SERVER, PF_INET, SOCK_STREAM, $proto) or die "socket: $!";
setsockopt(SERVER, SOL_SOCKET, SO_REUSEADDR, 1) or die "setsock: $!";
my $paddr = sockaddr_in($port, INADDR_ANY);

# bind to the port, then listen on it
bind(SERVER, $paddr) or die "bind: $!";
listen(SERVER, SOMAXCONN) or die "listen: $!";
print "Server agent started on port $port\n";

# accepting a connection
my $client_addr;
while ($client_addr = accept(CLIENT, SERVER)) {

# find out who connected
my ($client_port, $client_ip) = sockaddr_in($client_addr);
my $client_ipnum = inet_ntoa($client_ip);

# print who has connected -- this is for debugging only
print "Connection from: [$client_ipnum]\n";

# send the server agent response value
print CLIENT $response;

# close connection
close CLIENT;
}
```

Here is the output of the server program when it is started on the server:

```
$ ./serveragent.pl 50
Server agent started on port 1510
Connection from: [10.0.0.32]
```

Here is what you see if you **telnet** to the agent IP/port:

```
$ telnet 10.0.0.120 1510
50
Connection to host lost.
```

This program is only an example because it doesn't make any useful calculations of what the server agent response should be. Such calculations need to be made by the customer depending on what the server agent program is monitoring.

Appendix B:

Timeout Configuration

Timeouts ensure that certain operations are carried out within a finite period of time, and the resources that they use are returned for re-use. This document describes the various timeout parameters used by Equalizer, which can be divided into two major groups:

- **connection timeouts** -- used by Equalizer to manage connections to the clients on the network and the servers in clusters

- **probe timeouts** -- used by Equalizer to manage the various server health check mechanisms that assess server availability

Most parameters are global and apply to all clusters; many can be overridden in the cluster settings.

Connection Timeouts

Layer 7 clusters (HTTP / HTTPS) and Layer 4 clusters (TCP / UDP) each use a different set of timeout parameters. These are discussed in the sections below.

HTTP and HTTPS Connection Timeouts

Connections to HTTP and HTTPS clusters are managed closely by Equalizer from the client request to the response from the server. Equalizer needs to manage two connections for every Layer 7 connection request: the client connection from which the request originates, and the connection to the server that is the final destination of the request (as determined by the load balancing policy).

> This section does not apply to the E250

Equalizer has an idle timer for the established client connection, a connect timer to establish a server connection, and an idle timer for the established server connection. Only one timeout is in use at any given time. This is a summary of how timeouts are used when a client connects to Equalizer:

1. When a client successfully connects to a Virtual Cluster IP, the **client timeout** applies from the time the connection is established until the client request headers are completely transmitted. Equalizer parses the client's request, and verifies that the request is a valid HTTP request and that the information needed for load balancing is obtained. In general, this happens at the time that the client headers are completed -- which is indicated by the client sending two carriage-returns for HTTP 1.0 or 1.1; one carriage-return for HTTP 0.9. Once the headers are completely transmitted to Equalizer, the **client timeout** is no longer used.

2. As soon as the Equalizer is done examining the header data, it makes a connection to a server, as determined by the load balancing policy, persistence, or a match rule hit. The amount of time that the Equalizer tries to establish a connection to the server is the **connect timeout**. Once the server connection is established, the **connect timeout** is no longer used.

3. After Equalizer establishes a connection with a server, the **server timeout** is the amount of time Equalizer waits for the next bit of data from the server. Any response from the server restarts the **server timeout**.

The important distinction between the **client timeout** and the **server timeout** is that the **client timeout** is a "hard" timeout -- the client has the number of seconds specified to transmit all of its headers to Equalizer before Equalizer times out. This is done mainly for security considerations to prevent malicious clients from creating a large number of partial connections and leaking data slowly over the connection, possibly causing resource exhaustion or other undesirable effects on Equalizer.

The **server timeout** by contrast is a "soft" timeout -- the server has the number of seconds specified to send *the next piece of information* (e.g., the next packet in the sequence). Whenever the client or the server sends a piece of data on the connection, the **server timeout** is reset. This allows the server to send large data streams in small pieces without timing out, and then close the connection once all the data is sent.

For example, when a client sends a POST operation in a request, the **client timeout** is used up until the time that the POST *headers* have all been received. The **connect timeout** is used until a connection with the server is established. Then, once the connection is established, the **server timeout** is used for the POST data itself and the subsequent response from the server.

Note that there is the chance that a client will connect, send its headers, and then send continuous data to Equalizer that repeatedly resets the **server timeout**. This vulnerability is usually avoided by setting a hard client timeout on the application server itself (see "Application Server Timeouts" on page 214).

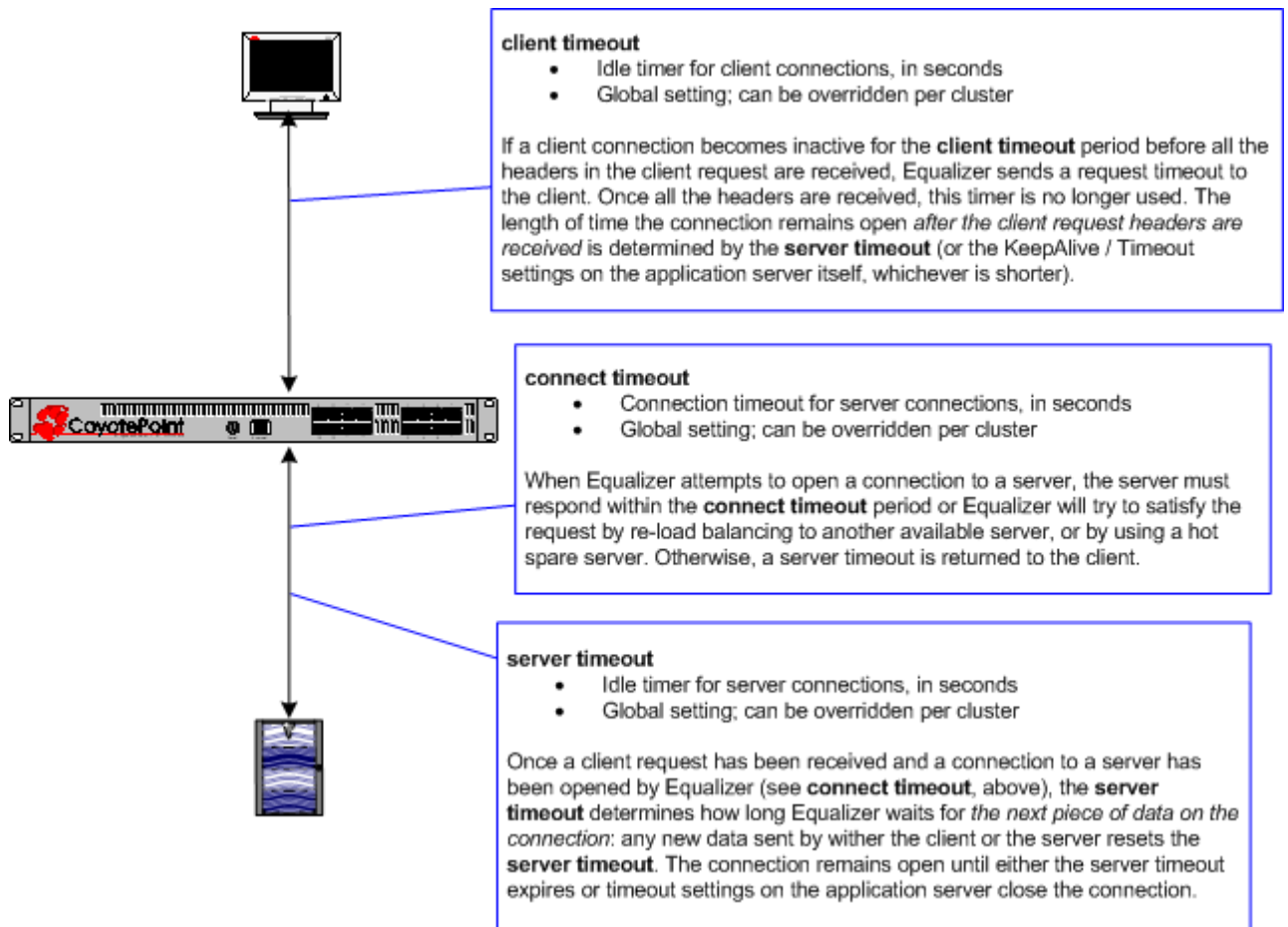Figure 64 summarizes the connection timeout parameters Equalizer uses for Layer 7 client and server connections.

client timeout
- Idle timer for client connections, in seconds
- Global setting; can be overridden per cluster

If a client connection becomes inactive for the **client timeout** period before all the headers in the client request are received, Equalizer sends a request timeout to the client. Once all the headers are received, this timer is no longer used. The length of time the connection remains open *after the client request headers are received* is determined by the **server timeout** (or the KeepAlive / Timeout settings on the application server itself, whichever is shorter).

connect timeout
- Connection timeout for server connections, in seconds
- Global setting; can be overridden per cluster

When Equalizer attempts to open a connection to a server, the server must respond within the **connect timeout** period or Equalizer will try to satisfy the request by re-load balancing to another available server, or by using a hot spare server. Otherwise, a server timeout is returned to the client.

server timeout
- Idle timer for server connections, in seconds
- Global setting; can be overridden per cluster

Once a client request has been received and a connection to a server has been opened by Equalizer (see **connect timeout**, above), the **server timeout** determines how long Equalizer waits for *the next piece of data on the connection*: any new data sent by wither the client or the server resets the **server timeout**. The connection remains open until either the server timeout expires or timeout settings on the application server close the connection.

Figure 64   Layer 7 connection timeout parameters

The timeline below shows the sequence of timeout events when a new connection is received by Equalizer.
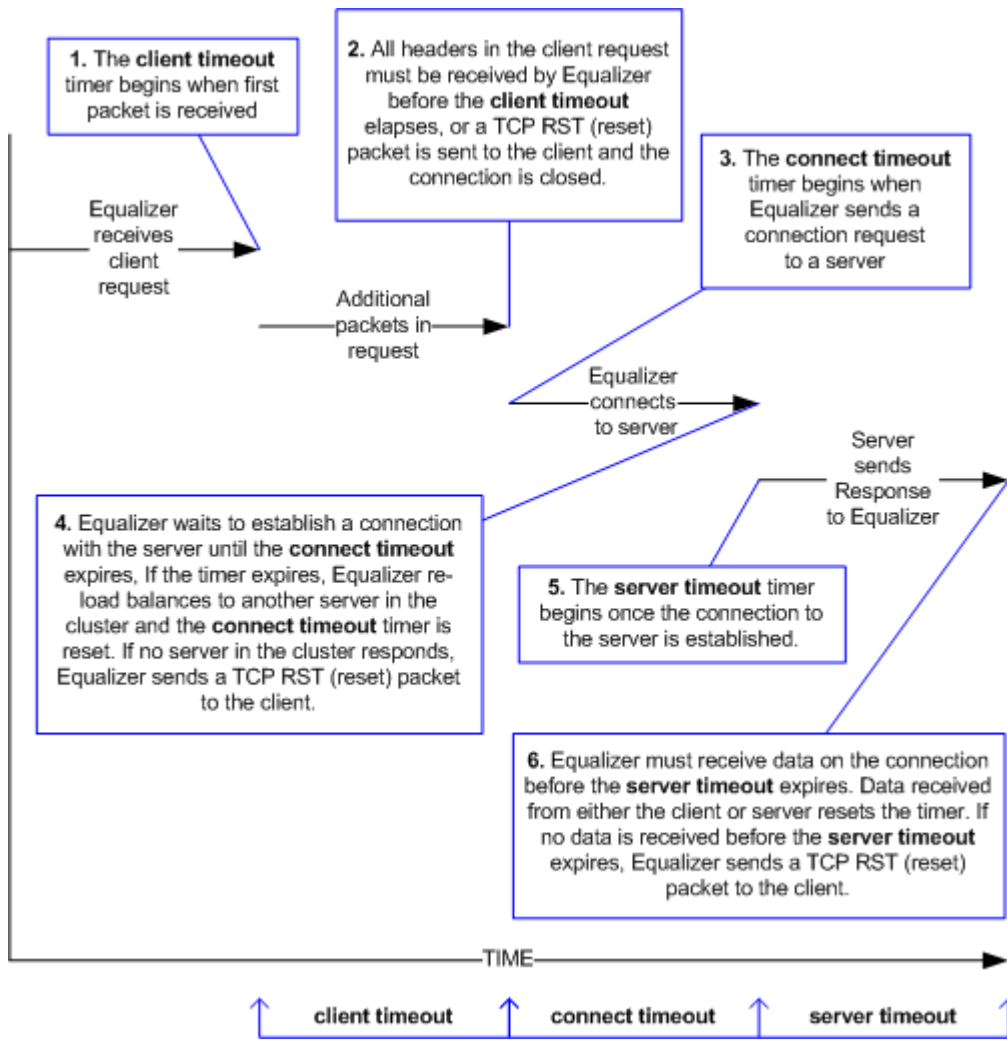
Figure 65   Layer 7 connection timeline

The following table shows the value range for the Layer 7 HTTP / HTTPS connection timeouts.

Parameter	Minimum	Default	Maximum	Units
client timeout	1.0	5.0	64535.0	seconds
server timeout	1.0	60.0	2147483647.0	seconds
connect timeout	1.0	10.0	60.0	seconds

The default timeout values are sufficient for many common applications. If timeouts are occurring using the default values, adjust the **server timeout** to the amount of time you expect your application server to respond to a client request, plus 1 second. If there is high latency between Equalizer and the servers in your cluster, then you may need to increase the **connect timeout**. The **client timeout** usually does not need to be changed; if you do need to increase it, use the lowest value possible for your configuration to work. High values for **client timeout** increase the risk of denial of service (DoS) attacks.

The Once Only Option and HTTP / HTTPS Timeouts

The previous sections describe how the connection timeouts work when the **once only** flag is *disabled* on a cluster; that is, when Equalizer is examining *every* set of headers received on a connection. The **once only** option, when enabled, specifies that Equalizer will examine only the *first* set of headers received on a connection. This has the following effects on connection timeouts:

- If you have **once only** *enabled*, as soon as the initial transaction (client request and server response) on a connection completes, the connection goes into "streaming" mode and the **client timeout** is no longer used for this connection. Equalizer does not parse any additional client requests received on the connection. The **server timeout** is used for the remainder of the connection, and is reset whenever data is received from either side of the connection.

- If you have **once only** *disabled* as described in the previous sections, and multiple requests are being sent on the same connection, the **client timeout** starts counting down again as soon as a new request is received from the client.

Layer 4 Connection Timeouts

Connections to Layer 4 clusters are received by Equalizer and forwarded with little processing. Equalizer simply rewrites the source and/or the destination IP addresses, as appropriate for the cluster, and sends the packet to the server specified by the cluster's load balancing policy. A *connection record* is kept for each connection so that address translation can be done on the packets going between the servers and clients. The Layer 4 connection timeouts specify how long a connection record is kept by Equalizer.

Layer 4 clusters use the **idle timeout** and **stale timeout** parameters. The **idle timeout** can be set at the global and cluster levels, while **stale timeout** can be set at the global level only. The parameters affect how Equalizer manages Layer 4 connection records:

- Connection records need to be removed in cases where the connection is not closed by the client or server, and is left idle. If no data has been received on a connection from either the client or the server after the time period specified by the **idle timeout** has elapsed, then Equalizer removes the connection record for that connection.

  Note that when using Direct Server Return (DSR), the time that a connection record is maintained is determined by adding the **idle timeout** for the cluster to the **sticky time** (see "sticky time" on page 85). This additional time is necessary when using DSR, since no server responses are routed through Equalizer (and therefore cannot restart the **idle timeout** to keep the connection open). For more information on DSR, see "Configuring Direct Server Return" on page 128.

- In other cases, a connection may be initiated but never established, so the connection record goes "stale" and must be removed. If a client fails to complete the TCP connection termination handshake sequence or sends a SYN packet but does not respond to the server's SYN/ACK, Equalizer marks the connection as *incomplete*. The **stale timeout** is the length of time that a connection record for an incomplete connection is maintained.

When Equalizer reclaims a connection, it sends a TCP RST (reset) packet to the server, enabling the server to free any resources associated with the connection. (Equalizer does *not* send a TCP RST to the client when reclaiming a connection.)

Reducing the **stale timeout** can be an effective way to counter the effects of SYN flood Denial of Service attacks on server resources. A **stale timeout** of 10.0 (see table below) would be an appropriate value for a site under SYN flood attack.

Parameter	Minimum	Default	Maximum	Units
idle timeout	0	0	2147483647.0	seconds
stale timeout	1.0	15.0	120.0	seconds

Note that if you change the **stale timeout** setting while partially established Layer 4 connections are currently in the queue, those connections *will* be affected by the new setting.

Application Server Timeouts

Keep in mind that the application server running on the physical servers in your cluster may have its own timeout parameters that will affect the length of time the server keeps connections to Equalizer and the client open. For example, an Apache 2 server has two related timeout directives: **TimeOut** and **KeepAliveTimeout**:

- The **TimeOut** directive currently defines the amount of time Apache will wait for three things:
    - The total amount of time it takes to receive a GET request.
    - The amount of time between receipt of TCP packets on a POST or PUT request.
    - The amount of time between ACKs on transmissions of TCP packets in responses.

- The **KeepAliveTimeout** directive specifies the number of seconds Apache will wait for a subsequent request before closing the connection. Once a request has been received, the timeout value specified by the **Timeout** directive applies.

In general, if you want Equalizer to control connection timeouts, you must make sure that any timeouts set on the application server are of longer duration than the values set on Equalizer.

For example, with respect to the Apache server timeouts above, the **client timeout** (for Layer 7 connections) or the **idle timeout** (for Layer 4 connections) should be of shorter duration than the timeouts set for Apache.

Similarly, the Layer 7 **server timeout** and **connect timeout** on Equalizer should be of shorter duration than the TCP connection timeouts set on the servers.

Connection Timeout Kernel Variables

Equalizer uses a number of kernel variables to track connection timeouts, as shown in the table below. You can use the `sysctl` command to display kernel variables. The two basic formats of this command are:

sysctl *variable_name*	Displays the kernel variable variable_name.
sysctl -a > *file*	Displays all kernel statistics. This is a long list, so we recommend capturing the list to a file.

eq.idle_timeout	The current setting of the Layer 4 global networking **idle timeout** parameter.
eq.idle_timedout_count	A Layer 4 counter incremented when a connection is terminated because the **idle timeout** expired.
eq.stale_timeout	The current setting of the Layer 4 global networking **stale timeout** parameter.
eq.l7lb.timeouts	The total number of Layer 7 connections dropped because a connection timer expired.
eq.l7lb.http.client_timeouts	The total number of Layer 7 (HTTP and HTTPS) connections that were terminated because the **client timeout** expired.
eq.l7lb.http.connect_timeouts	The total number of Layer 7 (HTTP and HTTPS) connections that were terminated because the **connect timeout** expired.
eq.l7lb.http.server_timeouts	The total number of Layer 7 (HTTP and HTTPS) connections that were terminated because the **server timeout** expired.

Note that there are also some kernel variables associated with Secure Socket Layer (ssl) client connections, such as when someone logs into Equalizer over an SSH connection. These variables are *not* incremented by HTTPS connections:

```
eq.l7lb.ssl.total_clients
eq.l7lb.ssl.current_clients
eq.l7lb.ssl.max_clients
eq.l7lb.ssl.requests
```

Server Health Check Probes and Timeouts

There are four levels of server health check probes supported by Equalizer:

- ICMP probes; all cluster types, enabled by default

- High Level TCP Probes; all cluster types, enabled by default

- High Level ACV (Active Content Verification) Probes; all cluster types except Layer 4 UDP clusters, disabled by default

- Server Agent Probes; all cluster types, disabled by default

ICMP Probes

By default, Equalizer sends an Internet Control Message Protocol (ICMP) echo request (commonly called a "ping") to the IP address of every server in every cluster.

The delay between successive ping requests to the same server is determined internally, but can be as short as one second on a server that is not responding to ICMP requests. On a lightly loaded Equalizer it may be 5 seconds or longer.

If a server does not respond to an ICMP echo request, Equalizer continues to issue any other probes (TCP, ACV, server agent) configured for the cluster. This means, for example, that if TCP and ICMP probes are both configured (the default), then a server can fail any number of ICMP probes and will still be marked *up* as long as it continues to respond to TCP probes.

If a server does not respond to an ICMP echo request and no other probes are configured, the server is marked *down*, and Equalizer continues to send ICMP requests to the server's IP address. If an ICMP echo response is subsequently received, the server is marked *up* again.

ICMP probing can be turned off by disabling the **ICMP probe** flag in the global parameters. This turns ICMP echo requests off for all clusters. (ICMP probes do not use any of the timeouts and parameters defined in the following section for High Level Probes.)

> **Note –** Responding to ICMP echo requests is an option on most server platforms. If ICMP echo reply is disabled on one or more of the servers your configuration, then you may want to disable ICMP echo requests on Equalizer to reduce traffic between Equalizer and the servers, and rely solely on the other probing mechanisms.

High Level TCP and ACV Probes

Equalizer sends High Level Probes to every server at the interval specified by **probe delay** (default: 10 seconds). By default, TCP probes are enabled for all servers, and ACV probes can be enabled for individual clusters. Both probes must complete within the same **probe timeout** period, and are controlled by the same set of parameters, as summarized in the following figure.

High Level TCP Probes
- sent to all servers by Equalizer
- sent to **probe port** on server
- can be disabled by **dont probe** server flag

High Level ACV Probes
- sent to servers in a cluster for which **ACV Probe** and **ACV Response** strings are defined
- sent to **probe port** on server
- can be disabled by **dont probe** server flag

ACV Probe / ACV Response
- required cluster settings for ACV
- **ACV Probe** is the string sent to the server
- **ACV Response** is the string expected in the server response

cluster

probe timeout
- set in seconds
- global setting
- time during which TCP and ACV probes must complete, or the server is marked '*failing*'

probe delay
- set in seconds
- global setting; can be overridden per cluster
- minimum time between *successful* probes of servers in the cluster

probe port
- cluster setting; can also be set per server
- by default, probe port is 0; probes are sent to the same port defined for server traffic
- if **probe port** is non-zero, Equalizer connects to the **probe port** for High Level Probes

dont probe
- server setting that disables TCP and ACV probes for a server

strikeout threshold
- global setting; the total number of TCP/ACV probes that must fail before a server is marked '*down*'
- the default value is 3; this means that after the first probe fails and the server is marked '*failing*', Equalizer sends up to two additional probes before it marks the server '*down*'

probe interval
- set in seconds
- global setting
- minimum time between additional probes of a server marked '*failing*' (see **strikeout threshold**, above)

Figure 66   Probe timeout parameters

The parameters shown in the figure above determine how high level TCP and ACV server probes are handled, as follows:

1. Equalizer begins a TCP probe by sending a TCP SYN packet to the server:
   - If the server and Equalizer negotiate a three-way TCP handshake (SYN, SYN/ACK, ACK) within the **probe timeout** period, go to **Step 2**.
   - If the server and Equalizer do not handshake within the **probe timeout** period, Equalizer marks the server *failing*; go to **Step 3**.

2. Equalizer then determines whether or not to send the server an ACV probe:
   - If **ACV Probe** and **ACV Response** are *not* defined for the cluster to which the server belongs, Equalizer marks the server *up* and waits for the **probe delay** period before it starts the HLP process again at **Step 1**.

- If **ACV Probe** and **ACV Response** are defined for the cluster to which the server belongs, the **ACV Probe** string is sent to the server. This is done as part of the same connection as the TCP probes, so the same **probe timeout** period also applies to the ACV probe (i.e., the **probe timeout** timer is not reset):

  - If the server responds to the ACV probe before the **probe timeout** expires, it is marked *up,* and Equalizer waits for the **probe delay** period before it starts the HLP probing process again at **Step 1**.

  - If the server does not respond to the ACV probe before the **probe timeout** expires, Equalizer marks the server *failing* and starts sending strikeout probes; go to **Step 3**.

3. This step is only performed when a server does not respond to a TCP or ACV probe within the **probe timeout**, and is marked *failing.* Before marking a *failing* server as *down,* Equalizer sends additional probes until the number of probes sent equals the value of the **strikeout threshold** parameter. The time between these additional probes is specified by the **probe interval** parameter:

   - If the failing server does not respond to any of the strikeout probes, it is marked *down*. Equalizer then continues sending TCP probes to the server using **probe interval** as the minimum delay between probes. If a response is ever received, Equalizer marks the server *up* and waits for the **probe delay** period before it starts the probing process again at **Step 1**.

   - If a failing server responds to one of the strikeout probes, Equalizer marks the server *up* and waits for the **probe delay** period before it starts the probing process again at **Step 1**.

The following figure shows the relationship between the **probe timeout** and **probe delay** parameters in a successful probing sequence.
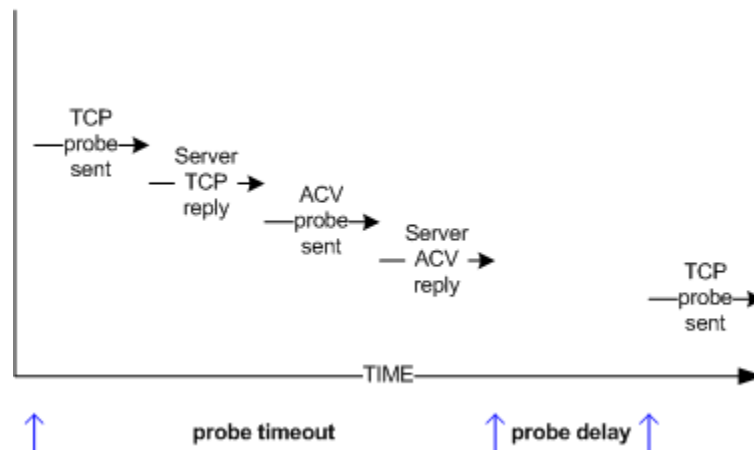
Figure 67   Successful probe sequence timeline

A server must respond to both TCP and ACV probes before the **probe timeout** elapses. Therefore, when ACV probes are enabled and probe failure messages are showing up in the log, you may need to increase the **probe timeout** so that the server has sufficient time to respond to both probes.

Assuming that a server responds successfully to TCP and ACV probes, Equalizer then waits for the **probe delay** time period before it sends the next TCP probe to the same server. (Note that the **probe delay** value is the *minimum* time between successful probes; the observed time may be longer for large configurations with many servers, during periods of high traffic, or due to Equalizer adjusting the delay internally to prevent server probes from consuming too much bandwidth on the network interface.)

In a network configuration where there is high latency between server probes and responses, the probe mechanism may falsely report that a server is down; this is indicated by messages in the event log indicating that a server is down and then comes back up again after a short period of time. In such cases you may need to increase the **probe timeout** or the **probe delay** parameters (or both) to reduce the number of false server down conditions reported by the probing mechanism.

The figure below shows the relationship between the **probe timeout** and **probe interval** parameters when a server does not respond to a High Level Probe.
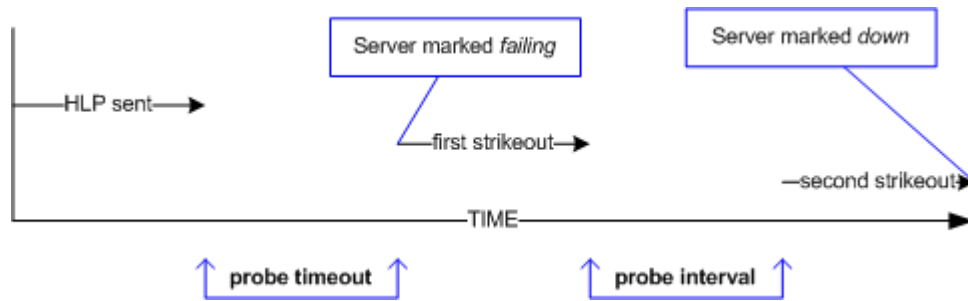
Figure 68   Unsuccessful probe timeout timeline

In the figure above, a High Level Probe (HLP) is sent to a server, which does not respond before the **server timeout** elapses. Equalizer marks the server as *failing* and sends two additional HLP probes (the default value of **strikeout threshold** is 3).

The **probe interval** specifies the time between these additional probes. If the server does not respond to either of these probes, it is marked *down*.

Note that the time periods between probes specified by the **probe interval** and **probe delay** values are *minimum* times. The observed time may be longer for large configurations with many servers, during periods of high traffic, or due to Equalizer adjusting the delay internally to prevent server probes from consuming too much bandwidth on the network interface. In addition, settings below 5 have the same effect as a setting of 5, since the Equalizer probe daemon cycles through the server probes every 5 seconds.

The range of values for each HLP parameter is shown in the table below. These apply to TCP and ACV probes only:

Parameter	Minimum	Default	Maximum	Units
probe timeout	1.0	10.0	60.0	seconds
probe delay	0.0	10.0	60.0	seconds
probe interval	0.5	20.0	25.0	seconds
strikeout threshold	1	3	6	integer

TCP Probe Aggregation

If a server is defined in more than one cluster and ACV probing is not enabled on any of the clusters to which the server belongs, then probes for that server are aggregated -- meaning, Equalizer only sends the server one TCP probe during each probe cycle, instead of sending one probe for each cluster. This reduces redundant probing.

Once ACV probing is enabled in a cluster, however, probe aggregation is disabled for all the servers in that cluster, and for all instances of these servers in other clusters.

For example, assume we have three clusters: **A**, **B**, and **C**. Let's say the same server, **sv01**, is defined in all three clusters. If ACV probing is not enabled in any of the three clusters, then **sv01** will be probed only once during each probe cycle, and the probe status will be reflected in all the clusters. Let's say we then enable ACV probing in cluster **A**. Thereafter, **sv01** will be probed independently for each cluster -- that is, Equalizer will probe **sv01** three times, once for each cluster **A**, **B**, and **C**.

It is possible, therefore, that defining a single server in many clusters and subsequently enabling ACV probes in any one of these clusters may result in a noticeable increase in probe traffic on the network.

Server Agent Probes

A server agent is a custom written application that runs on a server and listens on a specific port (default: 1510). When a connection request is received on that port, the server agent returns an integer value between -1 and 100 that indicates the relative load on the server (-1 meaning the server should be considered unavailable, 0 meaning very lightly loaded, and 100 meaning heavily loaded). Server agents can be used with any cluster type, and have an effect on all load balancing policies except **round robin**, which ignores server agent return values.

By default, server agents are disabled on all new clusters. To enable server agents for a cluster, you need to write the agent, install and run it on each server in the cluster, and then enable server agents for the cluster on Equalizer.

Agent Probe Process

When Equalizer connects to the port on which the server agent is running, it uses the number returned by the agent in its load balancing calculations, with the **server agent** policy giving highest preference to the server agent's return value over other factors.

The number returned by the agent to Equalizer is intended to indicate the current load on the server. The agent application that runs on the server can be written in any available scripting or programming language and can use any appropriate method to determine server load. The result must be an integer between -1 and 100 returned on the **server agent port**.

When enabled, server agents should be running on all servers in the cluster; however, by default, a server is not marked *down* when an agent value is not returned. Equalizer continues load balancing without the server agent return value unless the cluster parameter **require agent response** is enabled; if it is, Equalizer must receive an agent response or the server is marked *down*.

Note that server agent probing does not use any of the timeout values defined in the previous sections for High Level Probes. For example:

- The period of time between server agent probes to a server can be as short as one second. To introduce a timed delay, introduce a delay into the server agent code (for example, sleeping for 20 seconds). This does have the disadvantage of leaving the server agent port connection open for at least the length of the delay, but does reduce the frequency of agent probes.

- The period of time that Equalizer will wait for an agent response before marking it down is determined internally by Equalizer and cannot be adjusted by the administrator.

Enabling and Disabling Server Agents

Server agents are enabled for a cluster by turning on the **server agent** cluster flag, which sets the **server agent port** parameter to the default value of port **1510**. A connection to the server agent is opened on the **server agent port** specified up to every second -- depending on the cluster configuration, system load, and whether or not the server agent itself introduces a delay.

The **agent probe** cluster parameter specifies an optional string that is sent to the **server agent port** by Equalizer when it open a connection. This is not used by default, but is provided for those agents (such as agents written in Java) that require input before they reply to the probe. Agents written in C or perl, for example, usually don't require input in order to return the agent value.

Server agent probing is disabled by setting the **server agent port** parameter to **0**. Disabling the **server agent** flag automatically sets the port to 0.

Appendix C:

Using Reserved IP Addresses
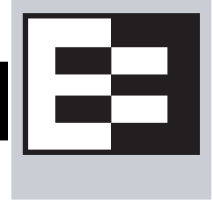
Equalizer supports placing servers on *reserved*, non-routable networks such as the class A network `10.0.0.0` and the class C network `192.168.0.0`. In environments in which the conservation of IP addresses is important, using reserved IP addresses can minimize the number of "real" IP addresses needed.

For example, an ISP hosting several hundred unique web sites replicated on three servers might not want to assign real IP addresses for all of them because each virtual cluster would consume four addresses: three on the back-end servers and one for the virtual cluster. In this case, the ISP might use `10.0.0.0` (the now-defunct Arpanet) as the internal network and assign virtual server addresses out this network for the servers. Figure 67 illustrates a typical reserved internal network.
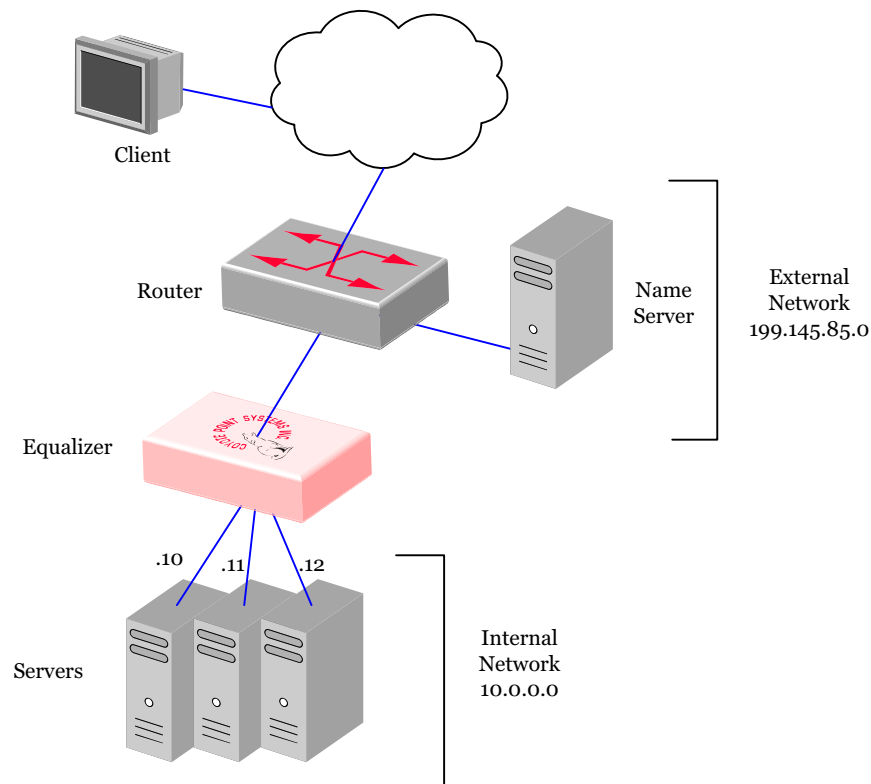
Figure 69   Reserved Internal Network

If servers placed on a non-routable network need to communicate with hosts on the Internet for any reason (such as performing DNS resolution or sending email), you must configure Equalizer to perform *outbound NAT* (network address translation). When you have enabled outbound NAT, Equalizer translates the source IP address in all packets

originating from the servers on the reserved network to Equalizer's external IP address. This way, clients will not see packets originating from non-routable addresses.

> **Note –** Enabling outbound NAT requires additional processing for each server response. If your servers do not need to initiate connections with hosts on the internet, disabling outbound NAT will improve performance.

To enable Equalizer to perform outbound NAT, follow these steps:

1. Open the Equalizer Administration Interface and log in under edit mode.

2. In the left frame, click the **Equalizer** (or system name) entry near the top of the object tree. In the right frame, select the **Networking** tab.

3. Enable the **enable outbound NAT** check box.

4. Click the **commit** button.

> **Note –** If you have two Equalizers deployed in a dual network Failover configuration, be sure to use the same outbound NAT setting on both Equalizers.

You will find a worksheet for configuring and using reserved IP addresses in "Sample Configuration Worksheets" on page 17.

Regular Expression Format

Equalizer supports only IEEE Std 1003.2 (POSIX.2) regular expressions in Match Rules. There are many other variants and extensions of regular expressions, including those found in Perl, Java, various shell languages, and the traditional Unix **grep** family of utilities; these variants are not supported in Match Rules.

Regular expressions can be difficult to create and debug, and can use significant system resources to process. We recommend you use regular expressions only when no other method will provide the functionality you require.

To aid in creating correct and efficient regular expressions, you can use a regular expression evaluator; many of these are available for download on the internet. Two free online regular expression evaluators are also available at the following websites:

```
http://www.rexv.org/   (choose POSIX tab)

http://www.projects.aphexcreations.net/rejax/   (choose PHP POSIX Language)
```

Terms

The terms in this section describe the components of regular expressions.

- A *regular expression* (RE) is one or more non-empty branches, separated by pipe symbols (|). An expression matches anything that matches one of the branches.

- A *branch* consists of one or more concatenated pieces. A branch matches a match for the first piece, followed by a match for the second, and so on.

- A *piece* is an atom optionally followed by a single *, +, or ?, or by a bound.
    - An atom followed by an asterisk matches a sequence of 0 or more matches of the atom.
    - An atom followed by a plus sign matches a sequence of 1 or more matches of the atom.
    - An atom followed by a question mark matches a sequence of 0 or 1 matches of the atom.

- A *bound* consists of an open brace ({) followed by an unsigned decimal integer, between 0 and 255 inclusive. You can follow the first unsigned decimal integer with a comma, or a comma and a second unsigned decimal integer. Close the bound with a close brace (}). If there are two integers, the value of the first may not exceed the value of the second.

Learning About Atoms

An *atom* followed by a bound that contains one integer i and no comma matches a sequence of exactly i matches of the atom. An atom followed by a bound that contains one integer i and a comma matches a sequence of i or more matches of the atom. An atom followed by a bound containing two integers i and j matches a sequence of i through j (inclusive) matches of the atom. An *atom* can consist of any of the following:

- A regular expression enclosed in parentheses, which matches a match for the regular expression.

- An empty set of parentheses, which matches the null string.

- A bracket expression.

- A period (.), which matches any single character.

- A carat (^), which matches the null string at the beginning of a line.

- A dollar sign ($), which matches the null string at the end of a line.

- A backslash (\) followed by one of the following characters: ^.[$()|*+?{\, which matches that character taken as an ordinary character.

- A backslash (\) followed by any other character, which matches that character taken as an ordinary character (as if the \ had not been present).

- A single character with no other significance, which simply matches that character. **Note that regular expressions are case-insensitive.**

- An open brace ({) followed by a character other than a digit is an ordinary character, not the beginning of a bound. It is illegal to end a real expression with a backslash (\).

Creating a Bracket Expression

A *bracket expression* is a list of characters enclosed in brackets ([...]). It normally matches any single character from the list. If the list begins with ^, it matches any single character not from the rest of the list. Two characters in a list that are separated by '-' indicates the full range of characters between those two (inclusive) in the collating sequence; for example, '[0-9]' in ASCII matches any decimal digit. It is illegal for two ranges to share an endpoint; for example, 'a-c-e'. Ranges are very collating-sequence-dependent, and portable programs should avoid relying on them.

- To include a literal ']' in the list, make it the first character (following an optional '^').

- To include a literal '-', make it the first or last character, or the second endpoint of a range.

- To use a literal '-' as the first endpoint of a range, enclose it in '[.' and '.]' to make it a collating element (see below).

With the exception of these and some combinations using '[' (see next paragraphs), all other special characters, including '\', lose their special significance within a bracket expression.

Within a bracket expression, a collating element (a character, a multi-character sequence that collates as if it were a single character, or a collating-sequence name for either) enclosed in '[.' and '.]' stands for the sequence of characters of that collating element. The sequence is a single element of the bracket expression's list. A bracket expression containing a multi-character collating element can thus match more than one character; e.g., if the collating sequence includes a 'ch' collating element, then the real expression '[[.ch.]]*c' matches the first five characters of 'chchcc'.

Within a bracket expression, a collating element enclosed in '[' and `]' is an equivalence class, representing the sequences of characters of all collating elements equivalent to that one, including itself. (If there are no other equivalent collating elements, the treatment is as if the enclosing delimiters were '[.' and '.]'.) For example, if 'x' and 'y' are the members of an equivalence class, then '[[x]]', '[[y]]', and '[xy]' are all synonymous. An equivalence class may not be an end-point of a range.

Within a bracket expression, the name of a character class enclosed in '[:' and ':]' stands for the list of all characters belonging to that class.

There are two special cases of bracket expressions: the bracket expressions '[[:<:]]' and '[[:>:]]' match the null string at the beginning and end of a word respectively. A word is defined as a sequence of word characters that is neither preceded nor followed by word characters. A word character is an alnum character (as defined by ctype(3)) or an underscore. This is an extension, compatible with but not specified by IEEE Std 1003.2 ("POSIX.2"), and should be used with caution in software intended to be portable to other systems.

Matching Expressions

If a real expression could match more than one substring of a given string, the real expression matches the one starting earliest in the string. If the real expression could match more than one substring starting at that point, it matches the longest. Subexpressions also match the longest possible substrings, subject to the constraint that the whole match be as long as possible, with subexpressions starting earlier in the real expression taking priority over ones starting later. Note that higher-level subexpressions thus take priority over their lower-level component subexpressions.

Match lengths are measured in characters, not collating elements. A null string is considered longer than no match at all. For example, 'bb*' matches the three middle characters of 'abbbc', '(wee|week)(knights|nights)' matches all ten characters of 'weeknights', when '(.*).*' is matched against 'abc' the parenthesized subexpression matches all three characters, and when '(a*)*' is matched against 'bc' both the whole real expression and the parenthesized subexpression match the null string.

Appendix E:

Using Certificates in HTTPS Clusters

The sections below tell you how to get your Layer 7 HTTPS clusters running with certificates. Please read these sections completely before beginning to work with certificates on Equalizer.

While this document tells you all you need to know to use certificates with HTTPS clusters, it is *not* a primer on HTTPS, SSL, or certificates. There are many resources on the Internet, in trade publications, and in books on these topics. Most SSL certificate vendors offer basic SSL overviews on their websites.

Using Certificates in HTTPS Clusters

The HTTPS protocol supports encrypted, secure communication between clients and servers. It requires that a Secure Sockets Layer (SSL) authentication handshake occur between a client and a server in order for a connection request to succeed.

When a client requests an HTTPS connection to a web server, the server (which has already been set up to support SSL connections) sends a *server certificate* to the client for verification. The client checks the content of the certificate against a local database of *Certificate Authorities*, and if it finds a match the connection is made. If no match is found (as is often the case with self-signed certificates), the browser will display a warning and ask if you want to continue with the connection.

A further level of trust can be enabled by setting the server up to request a *client certificate* in addition to the server certificate. Copies of the client certificate are pre-installed on both client and server. When the server sends the server certificate to the client, it also sends a request for a certificate from the client. Once the client accepts the server certificate as described above, it sends the client certificate to the server for verification. The server compares the client certificate it receives with its local copy of the client certificate, and if they match the connection is made.

A server certificate is required for an HTTPS connection; a client certificate is optional.

About Layer 7 HTTPS Clusters

In the typical HTTPS scenario described above, the client and server are communicating directly, and the server is doing all the work of encrypting and decrypting packets, comparing certificates, and authenticating clients. If you have many systems servicing requests for the same website, you'll need to install certificates on each server.

With Equalizer, you do not need to install a certificate on every server in a Layer 7 HTTPS cluster. Since certificates are associated with host names and not IP addresses, you only need a server certificate for each HTTPS cluster and the certificates are installed only on Equalizer -- not on each server. This reduces maintenance by reducing the number of certificates required for a group of systems serving content for the same host name.

When a client requests a connection to an HTTPS cluster, Equalizer establishes the HTTPS connection with the client, off loading SSL processing from all the servers in the HTTPS cluster. Equalizer communicates with the clients via HTTPS; the traffic between Equalizer and the servers in an HTTPS cluster is HTTP (i.e., unencrypted). Compared to the typical scenario where each server is establishing direct HTTPS connections with clients, encrypting and decrypting packets, and serving content as well, SSL offloading improves the overall performance of the cluster.

For even better performance, some Equalizer models are equipped with Xcel SSL Hardware Acceleration. With Xcel, all SSL processing is done by dedicated Xcel hardware, enhancing overall HTTPS throughput. For more information on Xcel, please visit the Coyote Point website (`www.coyotepoint.com`).

Note that HTTPS and certificates can be used on servers in Layer 4 TCP and UDP clusters, but you *will* need to install a server and client certificate on *each* server in the cluster (since Equalizer is not doing any HTTPS/SSL processing in Layer 4). In this scenario, no certificates are installed on Equalizer. Using a Layer 4 cluster is the preferred method for passing HTTPS traffic through Equalizer when you do not need to take advantage of features that are specific to Layer 7, such as cookie persistence, match rules, etc.

About Certificates

Each Layer 7 HTTPS cluster requires a *server* certificate; a *client* certificate is optional.

Web servers (such as Apache) and browsers (such as Internet Explorer and Firefox) are delivered with pre-installed Trusted Root Certificates. Trusted Root Certificates are used to validate the server and client certificates that are exchanged when an HTTPS connection is established.

Equalizer supports self-signed certificates, as well as signed certificates from Trusted Root Certificate Authorities and from Certificate Authorities (CAs) without their own Trusted Root CA certificates. If a CA without its own Trusted Root CA certificate issues your certificate, you will need to install at least two certificates: a server certificate and a chained root (or intermediate) certificate for the CA. The intermediate certificate associates the server certificate with a Trusted Root certificate.

Similarly, if you want to use client certificates with an HTTPS cluster, you'll need to get a signed client certificate from a CA, or create a self-signed certificate. A client certificate needs to be installed on each client that will access the Equalizer cluster, as well as on Equalizer. The same client certificate can be used on all clients (i.e., you don't need to buy or create a separate certificate for each client system).

Just as with server certificates, you may need to install a client certificate and a chained root certificate, if you obtain your certificates from a CA without its own Trusted Root CA certificate. Some sites prefer to use self-signed certificates for clients, or set up their own local CA to issue client certificates.

For several good tutorials on how to get your certificates signed, please see:

```
http://sial.org/howto/openssl/
```

Whichever method you choose, follow these general guidelines for certificates you want to use with Equalizer:

- Equalizer accepts both the **x509 PEM** or **PKCS12** certificate formats; PEM files usually have a *.pem* extension; PKCS12 files usually have a *.pfx* extension. Most CA vendors provide certificates in PEM format.

- Some older Equalizer models are equipped with an Xcel I Hardware SSL Acceleration, which requires a **private key length** of 1024 bits. This key length restriction does not apply to the newer generation Xcel II hardware, though a private key length of 1024 is recommended for best performance. (Note that all Equalizer GX hardware models that have Xcel are equipped with Xcel II.)

- When uploading certificates to Equalizer in **PEM** format, the certificates and private key must be contained in a single plain-text file, in the following order:
  - server certificate
  - private key
  - chained root (intermediate) certificates (if any)

Software vs. Hardware Encryption/Decryption

Without Xcel hardware SSL acceleration, all Layer 7 HTTPS encryption and decryption is performed by software, using Equalizer's CPU and memory. With Xcel, all SSL operations for Layer 7 HTTPS clusters are performed on dedicated hardware, thus offloading both the servers behind Equalizer and Equalizer itself -- freeing more resources for traffic and application management.

In terms of configuration, both software and hardware SSL operations require a list of cipher suites (encryption algorithms) to be used to encrypt and decrypt HTTPS traffic. The supported cipher suites for each SSL processing mode (software, Xcel I, Xcel II) are described in the section "Configuring Cipher Suites" on page 239.

Also see the section "Private Key Storage for Cluster Certificates" on page 238 for a discussion of how Equalizer stores the private keys for your cluster certificates, and keeping private keys secure on Equalizer.

Using Certificates in a Failover Configuration

In failover configurations, client and server certificates are *not* part of the configuration settings that are transferred between the failover peers when configuration changes are made on one of the failover systems. For this reason, you must install the server certificates (and the client certificates, if used) on *both* of the failover peers.

Enabling HTTPS with a Server Certificate

The following are the steps to follow to obtain and install a server certificate, and verify that it works.

1. Generate a Server Certificate Signing Request or a Self-Signed Server Certificate.

    To get a server certificate, do *one* of the following:

    a. **Create a Certificate Signing Request (CSR) and send it to a Certificate Authority for signing.** This provides the highest level of trust to the client, as the client can be assured that the certificate it receives from the server (in this case, Equalizer) was approved (i.e., digitally signed) by a trusted third party. Thus, the client has the assurance of a third party that the server to which it is connecting is identifying itself legitimately (and is not impersonating the legitimate server's identity). See the section "Generating a CSR and Getting It Signed by a CA" on page 231.

    b. **Create a certificate and sign it yourself.** This provides a lower level of trust, since the client is essentially trusting the server to identify itself. Self-signed certificates are relatively easy to counterfeit, and are only recommended for use on internal, non-production, or test configurations. See the section "Generating a Self-Signed Certificate" on page 232.

2. Create the HTTPS cluster.

    When creating an HTTPS cluster, the default flags and parameters are acceptable for most server certificate configurations.

    For more information on SSL parameters, see the section "Layer 7 Security > SSL Tab (HTTPS only)" on page 81.

3. Install the Server Certificate on Equalizer.

    Use the Equalizer Administration Interface to install the server certificate. See the section "Installing Certificates for an HTTPS Cluster" on page 234.

4. Try connecting to the Cluster via HTTPS.

    From a client browser, open **https://***cluster*, where ***cluster*** is the network node name or IP address of the HTTPS cluster. The browser may notify you that it is accepting a certificate from the server and ask for confirmation. Once you accept the certificate, the requested page should be displayed.

Enabling HTTPS with Server and Client Certificates

The following are the steps to follow to obtain and install both server and client certificates, and verify that they work.

1. Perform the procedure in the previous section ("Enabling HTTPS with a Server Certificate" on page 230) to enable HTTPS with a server side certificate.

2. Generate a Client Certificate Signing Request or a Self-Signed Client Certificate.

    In Step 1, you created a server certificate. Now, follow the same procedure to generate a client certificate; do *one* of the following:

    a. **Create a Certificate Signing Request (CSR) and send it to a Certificate Authority for signing**. See the section "Generating a CSR and Getting It Signed by a CA" on page 231.

    b. **Create a certificate and sign it yourself**. See the section "Generating a Self-Signed Certificate" on page 232.

    Many organizations choose to use third-party signed certificates for their HTTPS clusters, and use self-signed certificates for their clients.

3. Modify the HTTPS cluster to request a client certificate.

   a. Select the HTTPS cluster in the left frame of the Equalizer Administrative Interface and then select the **SSL** tab in the right frame.

   b. Enable the **certify_client** flag; this tells Equalizer to request a client certificate when a client attempts to connect to this cluster.

   c. By default, the **client certificate verification depth** is set to 2. This number indicates the number of levels in a certificate chain that the Equalizer will process before stopping (and refusing the connection). This default will need to be raised if you received more than one chained root certificate in addition to a client certificate from your Certificate Authority. Note that this setting has an impact on performance, since SSL operations are resource intensive.

   d. By default, Equalizer requests a client certificate, but does not *require* the client to provide one. Enable the **require certificate** flag to require that a client return a valid certificate before connecting.

   e. By default, the client's certificate will be re-validated if the SSL connection needs to be renegotiated. (Renegotiation is a feature of SSL, can occur for any of a number of reasons, and may be initiated by Equalizer or the client browser.) Enable the **verify once** flag to tell Equalizer *not* to re-evaluate the client certificate even if SSL renegotiation occurs. This can have a positive performance impact if many SSL renegotiations are occurring during normal operations.

   f. Select **commit** to save your changes to the cluster definition.

   For more information on SSL parameters, see the section "Layer 7 Security > SSL Tab (HTTPS only)" on page 81.

4. Install the Client Certificate on Equalizer.

   Use the Equalizer Administration Interface to install the client certificate. See the section "Installing Certificates for an HTTPS Cluster" on page 234.

5. Install the Client Certificate on all clients.

   Import the client certificate into the client browser's list of certificates. On Firefox, open **Tools > Options > Advanced > View Certificates**. On Internet Explorer, open **Tools > Internet Options > Content > Certificates**. Refer to the documentation for your browser for instructions.

6. Try connecting to the Cluster via HTTPS.

   From a client browser, open **https://*cluster***, where ***cluster*** is the network node name or IP address of the HTTPS cluster. The browser may notify you that it is accepting a certificate from the server and ask for confirmation. Once you accept the certificate, the server should ask for a client certificate; your browser may ask you to choose one. After the client certificate is sent to the server and accepted, the requested page should be displayed.

Generating a CSR and Getting It Signed by a CA

Most CA vendors provide a means of generating a Certificate Signing Request (CSR) on their websites, and we recommend that you use the CA website to generate the CSR.

A CSR can also be generated using the OpenSSL tools on any system, including Windows. The examples below were executed on a Windows system with the OpenSSL tools installed.

Note that only the most basic **openssl** command options are shown. See the **openssl**(1) and **req**(1) manual pages at `http://www.freebsd.org/cgi/man.cgi` for more information. Many certificate vendors also provide tools on their websites for entering a CSR.

Generating a CSR using OpenSSL

1. Navigate to an appropriate directory on your system, and create a new directory to hold your CSR, certificate, and private key.

2. Generate the CSR by entering this command:

```
openssl req -new -newkey rsa:1024 -out cert.csr
```

This begins an interactive session to generate a CSR, and also generates a new private key to be output into a file named *privkey.pem*. The key length you use (1024 in this example) can be any multiple of 8. If you already have a private key, use **-key** *filename* (instead of **-newkey rsa:1024**) to specify the file containing the private key. The key length you use (i.e., 1024 in this example) can be any multiple of 8.

After generating the private key, the following prompts are displayed (example responses shown):

```
Enter PEM pass phrase: <password>
Verifying - Enter PEM pass phrase: <password>
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:New York
Locality Name (eg, city) []:Millerton
Organization Name (eg, company) [Internet Widgits Pty Ltd]:CPS Inc.
Organizational Unit Name (eg, section) []:Engineering
Common Name (eg, YOUR name) []:mycluster.example.com
Email Address []:admin@example.com
```

Make sure you remember the **password** you specify, as you will need it to install and use the certificate.

For a *server certificate*, the **Common Name** provided must be the DNS-resolvable fully qualified domain name (FQDN) used by the Equalizer cluster. When a client receives the certificate from the server, the client browser will display a warning if the **Common Name** does not match the hostname of the request URI.

For a *client certificate*, the **Common Name** in the client's copy of the certificate is only compared to the **Common Name** in the copy of the client certificate on the server, so **Common Name** can be any value.

3. Visit the website of an SSL Certificate Authority (CA) to submit the *cert.csr* file to the CA.

4. Once the CA returns your signed certificate (usually in email), go to the section "Preparing a Signed CA Certificate for Installation" on page 233.

Generating a Self-Signed Certificate

To generate a self signed certificate in PEM format:

1. Generate a self-signed x509 format certificate by entering this command:

```
openssl req -new -x509 -newkey rsa:1024 -out selfcert.pem -days 1095
```

This creates a self-signed certificate (*selfcert.pem*) that will be valid for 1095 days (about three years) and also generates a new private key to be output into a file named *privkey.pem*. The key length you use (1024 in this example) can be any multiple of 8. If you already have a private key, use **-key** *filename* instead of **-newkey rsa:1024** to specify the file containing the private key. The key length you use (i.e., 1024 in this example) can be any multiple of 8.

After generating the private key, the following prompts are displayed (example responses shown):

```
Enter PEM pass phrase: <password>
Verifying - Enter PEM pass phrase: <password>
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:New York
Locality Name (eg, city) []:Millerton
Organization Name (eg, company) [Internet Widgits Pty Ltd]:CPS Inc.
```

```
Organizational Unit Name (eg, section) []:Engineering
Common Name (eg, YOUR name) []:myclient.example.com
Email Address []:admin@example.com
```

Depending on the tool you use to create the certificate, you may also be asked for a challenge password and other optional information. Make sure you remember the **password** (and, if prompted, the challenge password) you specify, as you will need it to install the certificate.

The **Common Name** provided must be the DNS-resolvable fully qualified domain name (FQDN) used by the Equalizer cluster. For a *server certificate*, when the client receives the certificate from the server, the browser will display a warning if the **Common Name** does not match the hostname of the request URI. For a *client certificate*, the **Common Name** in the client's copy of the certificate is only compared to the **Common Name** in the copy on the server, so this can be any value.

2. Combine the private key and certificate into one file, using a command like the following:

```
cat selfcert.pem privkey.pem > clustercert.pem
```

3. You can now install your self signed certificate and private key file, *clustercert.pem*, on Equalizer and your clients, as appropriate.

Preparing a Signed CA Certificate for Installation

When you receive your signed certificate back from your CA, you'll get one or more *.pem* files in return, or you'll get one or more mail messages from the CA. The files or messages contain your signed certificate and any necessary intermediate certificates required by the CA's chain of trust.

If you get your certificates in the mail, save each one to an ASCII text file with a *.pem* extension. Make sure you use a text editor such as **Notepad** (Windows) or **vi** (Unix/Linux) to save the files as text files.

Note that if you are using IIS, see the section "Using IIS with Equalizer" on page 236.

If you get only *one* certificate (the signed server certificate) from your CA, then:

1. Save it to a text file (e.g., *servcert.pem* for a server certificate, or *clientcert.pem* for a client certificate).

2. Open a new text file and read both the signed certificate and your private key (in this order) into the file. (The private key was created previously when you generated your CSR.) Save the file as a plain text file. On a Unix system, like Equalizer, you can do this with a command like one the following:

```
cat servcert.pem privkey.pem > clustercert.pem
```

```
cat clientcert.pem privkey.pem > clientprivcert.pem
```

Whatever method you use, the file should look like this when you are done:

```
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----
-----BEGIN RSA PRIVATE KEY-----
...
-----END RSA PRIVATE KEY-----
```

Make sure you save the file as a plain text file.

3. Install the file into Equalizer as instructed in the section "Installing Certificates for an HTTPS Cluster" on page 234.

If the CA uses chained root, or intermediate, certificates, then you'll receive (or need to download from the CA) more than one *.pem* file: the server certificate, plus any intermediate certificates needed to establish the chain of trust back to a Root CA certificate installed on your web server or client browser.

If you get *more than one* certificate (the signed server certificate plus one or more intermediate certificates) from your CA, then:

1. Save each certificate to a separate text file (e.g., *servcert.pem*, *intmcert.pem*).

2. Open a new text file and read the signed certificate, your private key, and any intermediate certificates (in this order) into the file. (Your private key was created previously, when you generated the CSR.) Save the file as a plain text file. On a Unix system, like Equalizer, you can do this with a command like one of the following:

   ```
   cat servcert.pem privkey.pem intmcert.pem > clustercert.pem
   ```

   ```
   cat clientcert.pem privkey.pem intmcert.pem > clientprivcert.pem
   ```

   Whatever method you use, the file should look like this when you are done:

   ```
   -----BEGIN CERTIFICATE-----
   ...
   -----END CERTIFICATE-----
   -----BEGIN RSA PRIVATE KEY-----
   ...
   -----END RSA PRIVATE KEY-----
   -----BEGIN CERTIFICATE-----
   ...
   -----END CERTIFICATE-----
   Add more certificates here if needed in the chain...
   ```

   Make sure you save the file as a plain text file.

3. Install the file into Equalizer as instructed in the section "Installing Certificates for an HTTPS Cluster" on page 234.

Installing Certificates for an HTTPS Cluster

Your certificate authority may issue you either a single signed client or server certificate, or a signed certificate plus one or more chained root certificates (also called "intermediate" certificates). The certificate or certificates you receive establish a chain of trust that ends at a trusted root certificate installed on your web server (and on every client that interacts with the web server).

If all of your clients use the same certificate to authenticate to the server, load the entire chain onto Equalizer. If each client uses a unique certificate, you can instead load all the intermediate and root certificates (minus the unique client certificate) onto Equalizer, and any client certificate presented that uses that chain will be accepted.

You must install all the certificates you receive on Equalizer to complete the installation process for HTTPS clusters. To install them on Equalizer, certificates must be in a single file, in either PEM (*.pem*) or PKCS12 (*.pfx*) format; see the section "Preparing a Signed CA Certificate for Installation" on page 233.

Caution – The private key for your *server* certificate is kept on Equalizer (in the directory */var/eq/ssl*) and will be accessible to anyone who can log into Equalizer. It is therefore essential that you restrict the ability of non-authorized personnel to access Equalizer, since any user can log in and copy or remove your private key. All Equalizer logins should be password protected with non-trivial passwords to restrict access to your private keys, and passwords should be given only to trusted personnel. Note that the private key for a *client* certificate (if used) is not stored on Equalizer, only the client certificate.

To install a certificate for an Equalizer cluster, follow these steps:

1. Copy the file containing the certificate and private key information (*clustercert.pem* in the examples above; *clustercert.pfx* if you used IIS) to the machine from which you will log into the Equalizer Administrative Interface. Note the location.

2. Log into the Administrative Interface using a login that has **add/del** access on the cluster that requires the certificate (see "Logging In" on page 31).

3. In the left frame, click the name of the HTTPS or SSL cluster for which you want to install a certificate and select the **Security > Certificates** tab in the right frame:

```
┌ select client or cluster certificate ─────────────────────────────────────────────┐
│                                                                                    │
│   For client verification, upload a single client certificate to authenticate all clients. For server verification, upload a │
│   single server certificate for the cluster.                                       │
│                                                                                    │
│   client  ○                                                                        │
│   cluster  ◉                                                                       │
│                                                                                    │
└────────────────────────────────────────────────────────────────────────────────────┘

┌ select SSL certificate file ───────────────────────────────────────────────────────┐
│                                                                                    │
│   The certificate file must be in PEM (.pem) or PKCS12 (.pfx) format, and must contain the private key and the entire │
│   certificate chain.                                                               │
│                                                                                    │
│   [                    ]   Browse...                                               │
│                                                                                    │
│   [ upload ]                                                                       │
│                                                                                    │
└────────────────────────────────────────────────────────────────────────────────────┘
```

Figure 69   The cluster Certificates tab

Note: If your Equalizer has **Xcel I** Hardware SSL Acceleration installed, a check box labeled **use secure key storage** will appear at the top of the **select client or cluster certificate** field. Checking this box tells Equalizer to store the private key for the server/cluster certificate in write-only memory on the Xcel hardware so that no one can access it. By default, this check box disabled and Equzalizer stores the private key in its file system, under */var/eq/ssl*. See the section "Private Key Storage for Cluster Certificates" on page 238, for more information. This option does not appear if your Equalizer is equipped with **Xcel II**, or does not have Xcel; in these cases, private keys are stored in Equalizer's file system, under */var/eq/ssl*.

4. If you are installing a *server* certificate, leave the **cluster** radio button selected; if you are installing a *client* certificate, make sure that the **client** radio button is selected.

5. Enter the full path name of the certificate file (or click **Browse** to select the file).

6. Click **upload** to install the certificate on Equalizer. You'll be prompted for a password, which is the password (or pass phrase) you provided when you generated the CSR for the certificate (or created the self-signed certificate).

   Note: If you select a file that is not in PEM or PKCS12 format (or select no file at all), the following error message is displayed:

   ```
   Certificate must be in PEM or PKCS12 format.
   ```

   Following the error message is the output the SSL tools returned after they were run on the uploaded file. The output may be unreadable and poorly formatted; this is normal, because the file you uploaded was not in the correct format. Click **dismiss** on the error popup and then go back to the previous step to select a file that is in PEM or PKCS12 format.

After the upload is complete, the **Certificates** tab displays the certificate details (serial number, key length, etc.) at the bottom of the tab.

Additonal SSL settings, including the cipher suites permitted, appear on the **SSL** tab. See "Layer 7 Security > SSL Tab (HTTPS only)" on page 81 and "Default Cipher Suites" on page 239 for more information.

7. If the certificate you just installed on Equalizer is a client certificate, you'll also need to install the certificate on each client. This usually involves converting the PEM format certificate into PKCS12 format; see the section "Converting a Certificate from PEM to PKCS12 Format" on page 237.

Using IIS with Equalizer

Using Internet Information Services (IIS) is optional when creating and managing certificates for Equalizer Layer 7 HTTPS clusters and clients. In fact, one of the advantages of using Equalizer is that only one server certificate is required for an HTTPS cluster. The cluster certificate is installed on Equalizer, *not* on the servers in the HTTPS cluster. So, you do not need to use IIS on each server to create and install certificates. This reduces the amount of effort spent administering server certificates.

For Layer 4 TCP and UDP clusters, certificates are *not* installed on Equalizer, and you *will* need to install a server certificate on *each* server in the cluster (since Equalizer is not doing any HTTPS/SSL processing in Layer 4). Generating a CSR and installing a signed certificate on Windows using IIS is shown in the procedure below.

Note that IIS does not support the creation of self-signed certificates. You must create the self-signed certificate on Equalizer (see "Generating a Self-Signed Certificate" on page 232) or another system that supports the OpenSSL tools; then, use IIS to import the certificate into the proper certificate store (usually, the **Personal** store) on Windows.

For more information on using IIS, please refer to the IIS documentation from Microsoft.

Generating a CSR and Installing a Certificate on Windows Using IIS

1. If you have not already installed Internet Information Services (IIS), use the **Add and Remove Programs** wizard (under **Control Panel**) to install it. Click on **Add/Remove Windows Components** and turn on the check box next to **Internet Information Services (IIS)**; click **Next** and follow the wizard's instructions.

2. Select **Control Panel > Administrative Tools > Internet Information Services**.

3. For a cluster (server) certificate, navigate to the website for which the CSR is intended. For a client certificate, navigate to any website or the default. Right click on the website and select **Properties**.

4. Select the **Directory Security** tab and click the **Server Certificate** button.

5. Select **Next**, and follow the Certificate Wizard prompts:

   a. Select **Create a new certificate**, and then **Next**.

   b. Select **Prepare the request now, but send it later**, and then **Next**.

   c. Type a **Name** for the certificate and select a **Bit Length** that is a multiple of 8. For most purposes, a bit length of 1024 is adequate. Longer bit lengths increase security at the expense of more SSL processing. Select **Next**.

   d. Type in an **Organization** (e.g., **MyCompany, Inc.**) and **Organizational Unit** (e.g., **Marketing**); then select **Next**.

   e. Type in the **Common name** for the certificate, and then select **Next**.

   For a *server certificate*, the **Common Name** provided must be the DNS-resolvable fully qualified domain name (FQDN) used by the Equalizer cluster. When a client receives the certificate from the server, the client browser will display a warning if the **Common Name** does not match the hostname of the request URI.

For a *client certificate*, the **Common Name** in the client's copy of the certificate is only compared to the **Common Name** in the copy of the client certificate on the server, so **Common Name** can be any value.

    f.   Type in a **Country/Region**, **State/province**, and **City/locality**; then select **Next**.

    g.   The last step in the wizard is to name and locate the new CSR. The default name and location will be *c:\certreq.txt* unless you choose otherwise.

6.   Visit the SSL vendor's website to submit your certificate request.

7.   Once the SSL vendor has mailed the new signed certificate back to you, do one of the following:

    a.   If you are using this certificate with a Layer 4 cluster, copy the new certificate onto the system on which you generated the request and double-click to install. If this is a server certificate for a server in a Layer 4 TCP or UDP cluster, make sure you attach it to the appropriate web site. If this is a client certificate, make sure you place the certificate in the **Personal** certificate store.

    b.   If you are using the certificate with a Layer 7 cluster, export your new SSL certificate with your private key, so that it can be installed on Equalizer:

    a.   In IIS, right click on the website for which the certificate was generated and navigate through **Properties > Directory Security > View Certificate > Details**.

    b.   Select **Copy to File**, then **Next**.

    c.   Select **Yes**, export the private key; then **Next**.

    d.   Select **PKCS #12 (.PFX)**; check **Enable strong protection**; then **Next**.

    e.   Type and confirm the password; then **Next**.

    f.   Enter a file name, e.g. *C:\clustercert.pfx*; then click **Next**.

    g.   Click **Finish**.

    h.   Click **Ok** if the export was successful.

    i.   The certificate is now ready to be uploaded to the cluster via the Equalizer Administration Interface; see "Installing Certificates for an HTTPS Cluster" on page 234.

Converting a Certificate from PEM to PKCS12 Format

Many browsers, such as FireFox and Internet Explorer, require private keys and certificates in PKCS12 format for installation. In order to install client and intermediate certificates into these browsers, you will first have to convert them from PEM format to PKCS12 format. (Note: if you created your certificate using IIS as explained in the previous section, then your certificate is already in PKCS12 format; it can be installed directly into a browser without conversion.)

Like PEM format, PKCS12 format supports having all your certificates and your private key in one file, as discussed above in the section "Preparing a Signed CA Certificate for Installation" on page 233. If you followed the instructions in that section and created the file *clientprivcert.pem* (containing the client certificate, the private key, and any intermediate certificates), then converting the file to PKCS12 is simple:

```
openssl pkcs12 -export -in clientprivcert.pem -out clientprivcert.pfx
```

The resulting file, *clientprivcert.pfx*, can now be installed into all client browsers that will be accessing the cluster that requires a client certificate.

In **Internet Explorer**, certificates are installed by selecting **Tools > Internet Options** from the main menu, selecting the **Content** tab, and pressing the **Certificates** button. Select the **Personal** tab and then the **Import** button.

In **FireFox**, certificates are installed by selecting **Tools > Options** from the main menu, selecting **Advanced**, selecting the **Encryption** tab, and pressing the **View Certificates** button. When the **Certificate Manager** appears, select the **Your Certificates** tab and then the **Import** button.

Private Key Storage for Cluster Certificates

When you upload a *cluster* certificate to Equalizer, the uploaded file contains:

- the cluster certificate
- zero or more intermediate certificates
- the private key for the cluster certificate (chosen by you when you created the certificate signing request or self-signed certificate)

The private key should be guarded carefully and access to it restricted to those who administer Equalizer. If you do not have Xcel enabled, or if you have Xcel II, private keys are kept in Equalizer's file system. Xcel I also provides the option to store provate keys in dedicated write-only memory.

Note that you should *not* check the **sks** check box when uploading *client* certificates, which are always stored on Equalizer *without* a private key.

Equalizer Xcel SSL Hardware Acceleration provides hardware-based SSL encryption and decryption. There are two versions of Xcel, Xcel I and Xcel II. The older version, Xcel I, optionally supports storing private keys for cluster certificates in dedicated write-only memory -- this is called **secure key storage** (SKS). All private keys uploaded to write-only memory can only be accessed by the accelerator hardware, thus preventing unauthorized access to your private keys.

If your Equalizer has Xcel I, a check box labeled **use secure key storage** will appear on an HTTPS cluster's **Certificates** tab (see Figure 69). Checking this box tells Equalizer to store your private key in Xcel I's write-only memory so that no one can access it.

Xcel I provides 128 kilobits of memory for private keys. This will hold up to 128 one-kilobit (1024-bit) keys, the only key length supported by Xcel I. (Be sure to use only 1024-bit private keys with Xcel I, regardless of whether SKS is used.)

For Xcel II, and on Equalizer models without Xcel, a key length of 1024 bits or less is recommended. While larger private keys are supported, 2048-bit and larger private keys can have a significant impact on performance.

> **Caution –** With Xcel II and on Equalizer models without Xcel, a cluster certificate's private key is stored on Equalizer in the directory */var/eq/ssl* and therfore will be accessible to anyone who can log into Equalizer. It is therefore essential that you restrict access to the Equalizer console via the serial line and SSH, since any user logged into the console can copy or remove your private key. **All Equalizer logins should be protected with non-trivial passwords, and logins should be given only to trusted personnel.**

Clearing Secure Key Storage on Xcel I

Over time, it is possible for the SKS memory on the Xcel I hardware to become full. When SKS is full, the following error is returned when you try to add another key (or replace an existing key):

```
Call to 'cert2sks' failed.
Error initializing RSA material
Using stdin
Could not allocate RSA key (N8_NO_MORE_RESOURCE).
Died at /usr/local/sbin/cert2sks line 286.
```

When this happens, you can do one of two things:

- Uncheck the **use secure key storage** check box when adding the SSL certificate; the private key will be kept on the Equalizer instead of in SKS.

- Clear SKS memory (using the procedure below); this removes all keys from SKS and will free up any space taken by keys that are no longer used (assuming you have not already used all 128kb of space on the

Xcel hardware with valid keys). After you clear SKS, you'll need to re-add all the certificates for all the HTTPS clusters whose keys were kept in SKS.

To clear SKS memory on Xcel I:

1. Log into Equalizer as *root* over the serial line, or login via SSH and use the **su** command to switch to the *root* login.

2. Enter the following command:

   ```
   SKSManager -R -u 0
   ```

3. After the operation completes (which should take about 1 minute), re-add all certificates for all HTTPS clusters.

Configuring Cipher Suites

The **cipher suite** HTTPS cluster parameter lists the supported encryption algorithms for incoming HTTPS requests. If a client request comes into Equalizer that does not use a cipher in this list, the connection is refused. If this field is blank, then any cipher suite supported by Equalizer's SSL implementation (or by Xcel Hardware SSL Acceleration, when enabled) will be accepted.

To view or set the **cipher suite** field for a cluster, click on the cluster name in the left frame and then select the **Security > SSL** tab in the right frame.

Default Cipher Suites

For an Equalizer with no Xcel SSL Hardware Acceleration installed and for systems with Xcel II enabled, the following default setting for **cipher suite** is used:

```
AES128-SHA:DES-CBC3-SHA:RC4-SHA:RC4-MD5:AES256-SHA
```

For an Equalizer with Xcel I enabled, the following default value is used:

```
DES-CBC3-SHA:RC4-SHA:RC4-MD5:AES256-SHA
```

Updating the Cipher Suites Field

This field can be used to specify a custom cipher suite required by the servers in a cluster. In general, to add a cipher suite, you spefiy a plus sigh (+) and then the name of the suite. To specifically exclude a cipher suite, use an exclamation point (!).

For example, SSLv2 encryption is supported by default. If your servers are required to support medium and high encryption using SSLv3 *only*, you can add "`!SSLv2`" to **cipher suite**. For example, the following cipher suite string will cause all non-SSLv3 client requests to be refused:

```
AES128-SHA:DES-CBC3-SHA:RC4-SHA:RC4-MD5:AES256-SHA:!SSLv2:+SSLv3
```

The **cipher suite** field requires a string in the format described in the OpenSSL cipher suite documentation, at:

```
http://www.openssl.org/docs/apps/ciphers.html
```

The tables in the following sections list the cipher suites supported by Equalizer. Also see the discussion of the cluster parameter "**cipher suite**" on page 81.

No Xcel (Software) and Xcel II Cipher Suites

The following cipher suites are supported by the base Equalizer software and by the Xcel II (newer generation) SSL Acceleration Hardware:

OpenSSL Cipher Suite Name	TLS/SSL Cipher Suite Names
AES128-SHA	TLS_RSA_WITH_AES_128_CBC_SHA
DES-CBC3-SHA	TLS_RSA_WITH_3DES_EDE_CBC_SHA SSL_RSA_WITH_3DES_EDE_CBC_SHA
RC4-SHA	TLS_RSA_WITH_RC4_128_SHA SSL_RSA_WITH_RC4_128_SHA
RC4-MD5	TLS_RSA_WITH_RC4_128_MD5 SSL_RSA_WITH_RC4_128_MD5
AES256-SHA	TLS_RSA_WITH_AES_256_CBC_SHA
The cipher suites below are supported but are not recommended. (In earlier releases, the EXP-RC4-MD5 ciphers were included by default in **cipher suite** for older browsers that only support 40-bit encryption. If some clients for your web services support only 40-bit encryption, then add EXP-RC4-MD5 to the **cipher suite** list.)	
EXP-RC4-MD5	TLS_RSA_EXPORT_WITH_RC4_40_MD5 SSL_RSA_EXPORT_WITH_RC4_40_MD5 SSL_CK_RC4_128_EXPORT40_WITH_MD5

Xcel I Cipher Suites

The following cipher suites are supported by the older generation Xcel I SSL Acceleration Hardware.

OpenSSL Cipher Suite Name	TLS/SSL Cipher Suite Names
DES-CBC3-SHA	TLS_RSA_WITH_3DES_EDE_CBC_SHA SSL_RSA_WITH_3DES_EDE_CBC_SHA
RC4-SHA	TLS_RSA_WITH_RC4_128_SHA SSL_RSA_WITH_RC4_128_SH
RC4-MD5	TLS_RSA_WITH_RC4_128_MD5 SSL_RSA_WITH_RC4_128_MD5

Appendix F:
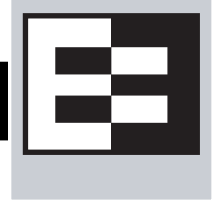
Equalizer VLB

Equalizer VLB™ is Coyote Point's virtualization enabled load balancing solution for VMware Infrastructure® virtual server configurations. It is available with either a Basic or Advanced license.

Equalizer VLB Basic

Equalizer VLB Basic uses VMware's management API to retrieve real-time virtual server performance information from a VMware Virtual Center console that manages virtual machines running on ESX Server (or from a single ESX Server directly). The additional server availability and resource utilization information obtained from VMware allows Coyote Point's Equalizer™ traffic management appliance to more efficiently direct the traffic flowing to VMware virtual machines. The diagram below illustrates how Equalizer VLB works:
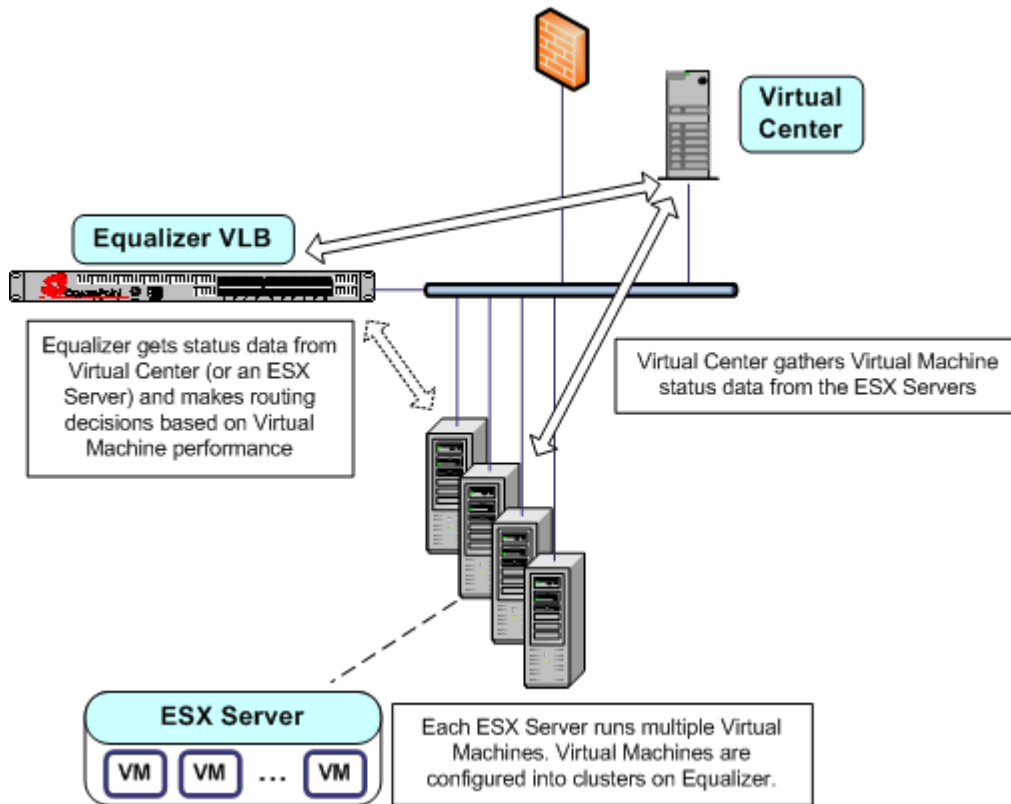
Equalizer extracts information from the Virtual Center console via the VMware API and load balances requests across virtual machines using knowledge of what is going on inside each virtual machine. If there is only one ESX Server in your configuration, Equalizer can also be set up to communicate directly with the ESX Server (instead of Virtual Center), and load balance among the virtual machines defined on that ESX Server only.

Equalizer uses statistics such as the amount of memory in use by a virtual machine, the amount of memory in use by all virtual machines on the physical host, and CPU utilization to automatically distribute incoming cluster requests to the virtual machines added to the cluster. Response to changes in VMware configuration is dynamic. If the virtual server performance in the pool is uneven, Equalizer automatically detects the uneven latency and sends new traffic to the best available virtual machine. If a server is overloaded and reboots, Equalizer simply detects that the server is available again, and automatically resumes sending traffic to it.

Equalizer VLB Advanced

In addition to the Equalizer VLB Basic functionality described above, Coyote Point has developed additional virtualization functionality, provided as Equalizer VLB Advanced. This functionality allows VLB to be tightly coupled with the Smart Control feature (see "Configuring Smart Events" on page 119). With Equalizer VLB Advanced, users can create Smart Events that read the status and control the functionality of virtual machines. To

accomplish this, there are certain Smart Control functions which are only available with an Equalizer VLB Advanced license. These functions are shown in Figure 36 on page 120 as "VLB only" functions.

Installation and Licensing

Equalizer VLB is installed automatically when you upgrade to Equalizer 8.0.1a, or a later release. It is supported on all current Equalizer appliance models, except the E250, and is a separately licensed product.

The following table summarizes the availability of Equalizer VLB and Smart Control:

Model	VLB Basic	VLB Advanced	Smart Control
E250	Not Available	Not Available	Not Available
E350	Included	Licensable	Included
E450	Included	Licensable	Included
E550	Included	Licensable	Included
E650	Included	Included	Included

When properly licensed, the bottom of the expanded **Welcome** screen displays the **Virtualization Load Balancing** version, as shown in following example.

```
              failover mode  standalone
Envoy geographic load balancing  disabled
            SSL acceleration  Xcel II
   Hardware GZIP compression  disabled
Virtualization Load Balancing  VLB Advanced
```

Enabling Equalizer VLB

In order to obtain VMware virtual machine information, Equalizer needs access information for the Virtual Center console managing the virtual machines. To enable communication between Equalizer and a Virtual Center console, do the following:

1. Follow the instructions in the VMware SDK & API documentation to install the VMware SDK on the system running Virtual Center (or on a single ESX Server). The SDK must be installed in order for Equalizer to be able to use VMware Infrastructure API calls and obtain virtual machine status. For instructions, see the VMware documentation at:

   `http://www.vmware.com/support/pubs/`

2. Log into Equalizer using an account that has **add/del** permission on global parameters.

3. Click **Equalizer** in the left frame, and then select the **Clusters > VLB** tab:

4. Enter the following information:

URL	The URL configured on the system running Virtual Center (or on an ESX Server) for VMware API connections. By default, this is an **https://** URL using the IP address of the Virtual Center system followed by **/sdk**, as in the example shown in the screen above.
Username	The VMware user account that you normally use to log into the Virtual Center or ESX Server that manages your VMware configuration.
Password	The password for your VMware user account. (Note that this text box is blank when you open the tab, even if a password has been previously saved.)

5. Click **enable VLB** to have Equalizer log into VMware automatically and enable virtual machines in clusters. (See "Disabling Equalizer VLB for all Clusters" on page 245 for a description of the **disable VLB** option.)

6. Click the **test** button to attempt to login to VMware using the information provided above.

7. Click **commit** to save your settings.

8. Optionally set the **agent delay**, the number of seconds between probes of VMware Virtual Center (or ESX Server) for the status of all virtual machines in all clusters (default: 10 seconds). To change the default:

   a. Select the **Equalizer > Probes** tab.

   b. Specify a new value in the **agent delay** text box.

   c. Click **commit** to save the new value.

Enabling VLB Agents on a Cluster

Once you have enabled VLB on Equalizer as shown in the previous section, you can configure clusters with VLB Agents. Doing so enables Equalizer to communicate with the Virtual Center and get detailed information on all the virtual machines configured in the cluster.

To enable VLB Agents on a cluster:

1. Log into Equalizer using an account that as **add/del** permission on the cluster to be modified.

2. Click the cluster name in the left frame. In the **Configuration > Required** tab, select **server agent** in the **policy** drop down box. The server agent policy gives preference to the values returned by the VLB agent, and is the recommended setting for VLB clusters.

3. Click **commit** to save the policy change.

4.   Select the **Configuration > Probes** tab:

5.   Select **VLB** in the **agent type** field.

6.   Click **commit** to save your settings.

Disabling VLB Agents for a Cluster

Disabling VLB Agents for a cluster means that Equalizer will no longer query the VMware virtual machine manager for status information for virtual machines associated with servers in that cluster. Traffic to the cluster is still load balanced across all servers associated with VMware virtual machines *without* the VLB Agent return value. Smart Rules defined for the cluster that use VLB-specific functions to query VMware will continue to be executed. You can still also add servers and associate virtual machines with them, as long as the VMware login information on the **VLB** tab is correct (see "Enabling Equalizer VLB" on page 243).

1.   Log into Equalizer using an account that as **add/del** permission on the cluster to be modified.

2.   Click the cluster name in the left frame, then select the **Configuration > Probes** tab in the right frame.

3.   Select **none** or in the **agent type** field.

4.   Click **commit** to save your settings.

Disabling Equalizer VLB for all Clusters

Disabling VLB globally means that Equalizer will no longer query the VMware virtual machine manager for status information for virtual machines associated with servers in any cluster (in other words, VLB Agents are disabled for all clusters). In addition, any Smart Rules that use functions that query VMware will *not* be executed. Cluster traffic is still load balanced across all servers associated with VMware virtual machines *without* the VLB Agent return value. You can still also add servers and associate virtual machines with them, as long as the VMware login information on the **VLB** tab is correct (see "Enabling Equalizer VLB" on page 243).

1. Log into Equalizer using an account that as **add/del** permission on global parameters.

2. Click **Equalizer** in the left frame, and then select the **VLB** tab in the right frame.

3. Click the **disable VLB** button near the bottom of the tab.

4. Click **commit** to save your settings.

Associating a Server with a Virtual Machine

Before you can associate a server with a virtual machine, VLB Agents must be enabled for the cluster, as described in the previous section. Associations are established either when a server is added to a VLB cluster, or later by editing an existing server's configuration:

1. Do *one* of the following:

    • If you are adding a new server to a VLB cluster, enable the Virtual Machine check box on the **Add New Server** dialog, and click **Next** (>). On the following screen, click the **Associate with Virtual Machine** button to query VMware.

    • To associate an existing Equalizer server with a virtual machine, click the server name in the left frame and open the **Virtual Machine** tab in the right frame. Click the **Associate with Virtual Machine** button to query VMware.

2. If the query is successful, a list of available vitrual machines is displayed. Choose the virtual machine you want to associate with the Equalizer server definition and click the **Associate** button. (Note that in order for VM selection to work, VLB must be enabled as described in the section "Enabling Equalizer VLB" on page 243.)

You can add both virtual machines and non-virtual machines (physical servers) to a VLB cluster. The non-virtual machines will be load balanced without any VLB server agent value.

Similarly, you can mix virtual and non-virtual machines as servers in a non-VLB cluster. The virtual machines will be load balanced as if they were physical servers, using no VMware data.

Finally, whether you are using VLB as the probe method for the cluster or not, you can use servers which are associated with virtual machines within Smart Control Events, and these servers will be monitored and controlled as specified in the events.

Smart Control Event Examples Using VLB

VLB Advanced provides VLB-specific extentions to the Smart Events feature -- additional functions are provided that let you query a VMware configuration for status information and perform VMware specific operations, such as powering down a virtual server. See the section "Configuring Smart Events" on page 119 for a complete list of supported actions and triggers.

The following examples show you how to use Smart Events in an Equalizer VLB configuration. We assume that you have already configured Equalizer to work with an existing VMware installation, by supplying the appropriate login information on the **Equalizer > Clusters > VLB** tab.

Configuring Multiple Hot Spares (VLB Only)

The **hot spare** server flag is used to designate a single hot spare server in a cluster. In some cases, you may want to designate multiple hot spare servers for a cluster. This can be done using Smart Events.

For example, assume a cluster that has three servers: **sv00**, **sv01**, and **sv02**. As long as it is up, we want **sv00** to actively pass traffic and the other two servers act as successive hot spares. That is, **sv01** and **sv02** will have the

quiesce option enabled while **sv00** is running. If **sv00** fails, we want **sv01** to take over. If both **sv00** and **sv01** are unavailable, we want **sv02** to take over.

To accomplish this, we'll create three Smart Events to be evaluated whenever Equalizer processes cluster events:

Event 1: If the first server is running, enable the quiesce option on the other two servers; otherwise, do nothing.

Event 2: If the first server is not running and the second server is, enable the quiesce option on the first and third servers, and disable it on the second server; otherwise, do nothing.

Event 3: If both the first server and the second server are not running, enable the quiesce option on the first and second servers, and unquiesce the third server; otherwise, do nothing.

To create Event 1:

1. Right-click on the cluster name in the left frame and select **Add Event** from the menu:

2. Type in an event name, such as **activate-sv00**, or accept the default. Click the next icon (**>**)at top to open the **Event Trigger** editor.

3. In the **functions** field, click **running**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **running**. Select **sv00** and click **accept**. The **expression workbench** should now look like this:

4. Click the next icon (**>**)at top to open the **Event Action** editor.

5. In the **functions** field, click **quiesce**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **quiesce**. Select **sv01** and click **accept**.

6. In the **operators** field, click **&&**.

7. In the **functions** field, click **quiesce**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **quiesce**. Select **sv02** and click **accept**. The **expression workbench** should now look like this:

8. Click the next icon (**>**)at top, and then **commit** to create the event. The Configuration tabs for the event open in the right frame.

To create Event 2:

1. Right-click on the cluster name in the left frame and select **Add Event** from the menu:

2. Type in an event name, such as **activate-sv01**, or accept the default. Click the next icon (**>**)at top to open the **Event Trigger** editor.

3. In the **operators** field, click on the NOT operator (**!**).

4. In the **functions** field, click **running**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **running**. Select **sv00** and click **accept**.

5. In the **operators** field, click **&&**.

6. In the **functions** field, click **running**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **running**. Select **sv01** and click **accept**. The **expression workbench** should now look like this:

7. Click the next icon (**>**)at top to open the **Event Action** editor.

8. In the **functions** field, click **quiesce**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **quiesce**. Select **sv00** and click **accept**.

9. In the **operators** field, click **&&**.

10. In the **functions** field, click **quiesce**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **quiesce**. Select **sv02** and click **accept**.

11. In the **operators** field, click **&&**.

12. In the **functions** field, click **unquiesce**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **unquiesce**. Select **sv01** and click **accept**. The **expression workbench** should now look like this:

13. Click the next icon (**>**)at top, and then **commit** to create the event. The Configuration tabs for the event open in the right frame.

To create Event 3:

1. Right-click on the cluster name in the left frame and select **Add Event** from the menu.

2. Type in an event name, such as **activate-sv02**, or accept the default. Click the next icon (**>**)at top to open the **Event Trigger** editor.

3. In the **operators** field, click on the NOT operator (**!**).

4. In the **functions** field, click **running**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **running**. Select **sv00** and click **accept**.

5. In the **operators** field, click **&&**.

6. In the **operators** field, click on the NOT operator (**!**).

7. In the **functions** field, click **running**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **running**. Select **sv01** and click **accept**. The **expression workbench** should now look like this:

8. Click the next icon (**>**)at top to open the **Event Action** editor.

9. In the **functions** field, click **quiesce**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **quiesce**. Select **sv00** and click **accept**.

10. In the **operators** field, click **&&**.

11. In the **functions** field, click **quiesce**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **quiesce**. Select **sv01** and click **accept**.

12. In the **operators** field, click **&&**.

13. In the **functions** field, click **unquiesce**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **unquiesce**. Select **sv02** and click **accept**.

14. In the **operators** field, click **&&**.

15. In the **functions** field, click **log**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **log** and type "Last hot spare activated! Check servers!". Click **accept**. The **expression workbench** should now look like this:

16. Click the next icon (**>**)at top, and then **commit** to create the event. The Configuration tabs for the event open in the right frame.

Note that the above example is very basic and does not handle all possible event combinations. In particular:

- No action is taken when one of the hot spares becomes unavailable while the first server is passing traffic. These will be indicated as down in the Cluster and Server tabs.

- No action is taken if the third server is not available. This could be handled by a separate event.

- No action is taken if all three servers are not available. This is best handled by a Responder combined with a Match Rule.

Rebooting an Unresponsive Virtual Machine (VLB only)

In this example, we want to reboot a particular virtual machine running under VMware when the machine has been unresponsive for 15 minutes.

To do this, we create two events to power up and power down the machine, and a third event that acts as a timer for the power down and power up events.

Event 1: If the server is not running, power the server down; otherwise, do nothing.

Event 2: If the server is not powered on, power the server on; otherwise, do nothing.

Event 3: If the server is running, block Event 1 and Event 2 for 15 minutes; otherwise, do nothing.

While the server is running, Event 3 continually blocks the other two events from being evaluated. If the server goes down, Event 1 stops blocking after about 900 seconds. The first time that Events 1 and 2 are evaluated, Event 1 is triggered while Event 2 does nothing (Event 1 has only just triggered, so VMware is not reporting the server as down yet). At the next event cycle, Event 2 determines that the server is not powered on and will power it back on, completing the reboot.

To create Event 1:

1. Right-click on the cluster name in the left frame and select **Add Event** from the menu:

2. Type in an event name, such as **poweroff-sv00**, or accept the default. Click the next icon (**>**)at top to open the **Event Trigger** editor.

3. In the **operators** field, click on the NOT operator (**!**).

4. In the **functions** field, click **running**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **running**. Select **sv00** and click **accept**. The **expression workbench** should now look like this:

5. Click the next icon (**>**)at top to open the **Event Action** editor.

6. In the **functions** field, click **power_off**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **power_off**. Select **sv00** and click **accept**. The **expression workbench** should now look like this:

7. Click the next icon (**>**)at top, and then **commit** to create the event. The Configuration tabs for the event open in the right frame.

To create Event 2:

1. Right-click on the cluster name in the left frame and select **Add Event** from the menu:

2. Type in an event name, such as **poweron-sv00**, or accept the default. Click the next icon (**>**)at top to open the **Event Trigger** editor.

3. In the **operators** field, click on the NOT operator (**!**).

4. In the **functions** field, click **powered**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **powered**. Select **sv00** and click **accept**. The **expression workbench** should now look like this:

5. Click the next icon (**>**)at top to open the **Event Action** editor.

6. In the **functions** field, click **power_on**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **power_on**. Select **sv00** and click **accept**. The **expression workbench** should now look like this:

7. Click the next icon (**>**)at top, and then **commit** to create the event. The Configuration tabs for the event open in the right frame.

To create Event 3:

1. Create the timer event. Right-click on the cluster name in the left frame and select **Add Event** from the menu.

2. Type in an event name, such as **reboot-timer**, or accept the default. Click the next icon (**>**)at top to open the **Event Trigger** editor.

3. In the **functions** field, click **running**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **running**. Select **sv00** and click **accept**. The **expression workbench** should now look like this:

4. Click the next icon (**>**)at top to open the **Event Action** editor.

5. In the **functions** field, click **event_wait**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **event_wait**. Select **poweroff-sv00**, and then type "**900**" for the **wait seconds**. Click **accept**.

6. In the **operators** field, click **&&**.

7. In the **functions** field, click **event_wait**. In the **expression workbench** field at bottom, click on the drop-down arrow next to **event_wait**. Select **poweron-sv00**, and then type "**900**" for the **wait seconds**. Click **accept**. The **expression workbench** should now look like this:
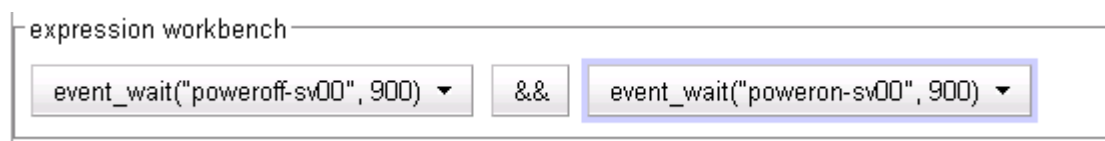
8. Click the next icon (**>**)at top, and then **commit** to create the event. The Configuration tabs for the event open in the right frame.

VLB Logging

Equalizer VLB writes a number of messages to the equalizer log (*Equalizer* **> Status > Event Log**). These messages are described below (timestamps normally displayed at the beginning of each line have been omitted):

```
Logged into VC or ESX successfully
Failed to connect to VC or ESX
Failed to log into VC or ESX
```

The messages above indicate that Equalizer attempted to log into the Virtual Center or ESX Server IP configured on the **VLB** tab. The status of the first login attempt after a reboot is recorded in the log; subsequent attempts are only logged if the login status changed since the last login. For example, the first successful login attempt is logged; subsequent successful attempts are not recorded. Likewise, the first failure is recorded; no further messages are logged during subsequent attempts until a login attempt succeeds.

```
VLB:  probe:  Server IP_address VLB state changed from old_value to new_value
```

A message in the above format indicates that the VLB agent return value for the virtual machine at *IP_address* has changed since the last probe of the Virtual Center (or ESX Server). Both the previous return value (*old_value*) and the latest return value (*new_value*) are logged in the message.

For example, the following series of messages was logged when a spike of CPU activity reduced availability for one virtual machine (server) in a VLB cluster:

```
VLB:  probe: Server 192.168.1.51 VLB state changed from 0 to 100
VLB:  probe: Server 192.168.1.51 VLB state changed from 100 to 20
VLB:  probe: Server 192.168.1.51 VLB state changed from 20 to 0
VLB:  probe: Server 192.168.1.51 VLB state changed from 0 to 1
VLB:  probe: Server 192.168.1.51 VLB state changed from 1 to 100
```

As the messages indicate, Equalizer continually adjusts to changing conditions on the server. Without VLB agents, Equalizer would not have known about the CPU utilization spike since the 'ping time' of the server IP did not change during this period.

VLB Plotting

The VLB agent return values can be plotted for any virtual machine in a VLB cluster.

1. Click on the server name in the left frame object tree. Select the **Reporting > Plots** tab in the right frame.

2. In the **display** multi-pick box, select **Server Agent**. Select other options as desired (click **Help > Context Help** for descriptions of each setting).

3. Select **plot** to display the graph.

Additional Operational Notes

1. **Failover**: All Equalizer VLB configuration settings are stored in the Equalizer configuration file, and so are transferred over to the failover peer when the configurations are synchronized.

   We recommend that both failover peers run Equalizer VLB. If Equalizer VLB is used in a failover configuration with an Equalizer that is not running Equalizer VLB, then the **dont transfer** flag must be enabled on both peers. To view or set this option, select the object at the top of the left frame and then open the **Parameters** tab.

2. **Envoy**: Equalizer VLB operation is transparent to Envoy. In other words, you can use a VLB cluster in a GeoCluster configuration just like any other cluster.

Troubleshooting

You usually can diagnose Equalizer installation and configuration problems using standard network troubleshooting techniques. This section identifies some common problems, the most likely causes, and the best solutions.

For additional Troubleshooting information, as well as the most up to date documentation, supplements, and technical articles, please visit the Coyote Point Support website:

http://www.coyotepoint.com/support.php

Equalizer Doesn't Boot for First Time

Terminal or terminal emulator not connected to Equalizer

Check the serial cable connection and the communication settings of the terminal or terminal emulator. Required settings are 9600 bps, 8 data bits, no parity, one stop bit, and VT100 terminal emulation. If you are using terminal emulation software on a Windows or Unix system, make sure the terminal emulation software is connecting to the port to which the serial cable is connected.

Newer Equalizer models also have a USB keyboard connector and VGA display adapter at the back of the unit. You can connect a USB cable and VGA display and use these as a console instead of the serial port.

Clients Time Out Trying to Contact a Virtual Cluster

Equalizer is not gatewaying reply packets from the server

Log on to the server(s) and check the routing tables. Perform a `traceroute` from the server to the client. Adjust the routing until Equalizer's address shows up in the `traceroute` output.

> ⚠ **All packets sent from the server back to clients must pass through Equalizer unless the spoof cluster option is disabled.**

Test client is on the same network as the servers

If the test client is on the same network as the servers, the servers will probably try to send data packets directly to the client, bypassing Equalizer. You can correct this by adding *host routes* on the servers so that the servers send their reply packets via Equalizer.

No active servers in the virtual cluster

Possible solutions:

- Check the Equalizer Summary page. Are there any servers in that virtual cluster? Are all the servers marked DOWN?

- Log onto the server an run the `netstat` command (Unix servers). If the `netstat` output shows connections in the SYN-RCVD state, the server is not forwarding its reply packets to Equalizer.

Equalizer is not active

Is Equalizer functioning? Try to `ping` the administration address. If you do not get a response, "Equalizer Doesn't Respond to Pings to the Admin Address" provides additional troubleshooting information.

Primary and Backup Equalizer Are in a Conflict Over Primary

Certain switches (often those from Cisco and Dell) have Spanning Tree enabled by default. This can cause a delay in the times that the network is accessible and cause the backup Equalizer to enter into failover mode. If you cannot disable Spanning Tree, enable FastPort for all ports connected to the Equalizers.

Backup Equalizer Continues to Boot

Primary and Backup Equalizer Are in a Conflict over Primary

Certain Dell and Cisco switches have Spanning Tree enabled by default. This can cause a delay in the times that the network is accessible and cause the backup Equalizer to enter into failover mode. If you cannot disable Spanning Tree, enable PortFast for all ports connected to the Equalizers.

Can't View Equalizer Administration Pages

Equalizer is not active

Is Equalizer functioning? Try to `ping` the administration address. If you do not get a response, see "Equalizer doesn't respond to pings to the admin address" below, which provides additional troubleshooting information.

Equalizer Administration Interface Unresponsive

Clear your browser cache; or, close your browser and open it again to establish a new connection.

Equalizer Administration Page Takes a Long Time to Display

DNS server configured on Equalizer is not responding

Possible solutions:

- Check that Equalizer has IP connectivity to the name server configured using the serial configuration utility.

- If you want to disable DNS lookups on Equalizer, specify a name server IP address of 0.0.0.0 in Equalizer's serial configuration utility.

Equalizer Doesn't Respond to Pings to the Admin Address

Equalizer is not powered on

Check that power switch is on and the front panel LED is lit. Connect the keyboard and monitor, cycle the power, and watch the startup diagnostic messages.

Equalizer isn't connected to your network

Check the network wiring.

Administration address not configured on the external interface

This applies to dual network configurations. Use the Equalizer Configuration Utility to set the IP address and netmask for external interface. Be sure to commit your changes.

Browser Hangs When Trying to Connect Via FTP to an FTP Cluster

FTP server returns its private IP address in response to a "PASV" command

This behavior is likely to cause problems if you're using reserved internal addresses for the server. Enabling PASV mode FTP translation on the **Networking** tab of the Equalizer Administration Interface substitutes the cluster IP for the server IP in PASV responses. For more information, see "**passive FTP translation**" on page 51.

Return Packets from the Server Aren't Routing Correctly

IP spoofing is enabled

This problem normally occurs in a single network setup. When you enable IP spoofing, clustered servers see the client's IP address. If the server tries to reply directly to the client, the client will reject the reply (it had sent its request to a different address).

Run a `traceroute` to ensure that routes from a server to a client go through Equalizer and not directly back to the client. If Equalizer does not appear, modify the route to include Equalizer. Alternatively, you can disable IP spoofing.

Web Server Cannot Tell Whether Incoming Requests Originate Externally or Internally

IP Spoofing is not enabled

Check the cluster's configuration and enable IP spoofing. This causes Equalizer to pass the client's IP address. Make sure that responses from the server go through the Equalizer.

Why aren't my clusters working if the server status is "up"?

There are several reasons this could be happening. Make sure that Equalizer is being used as the default gateway on all your servers, and that the server service or daemon is running. Sometimes additional host or network routes will need to be added to the clustered servers in single network. The **traceroute** (Unix) and **tracert** (Windows) commands area useful diagnostic tools. Trace from the clustered server back to any client that is not able to resolve the cluster address. If Equalizer is not showing up as the first hop, routing is the cause of the problem.

Context Help Does Not Appear

Turn off the Pop-up Blocker for your browser. In FireFox, select **Tools > Options > Content** and disable the **Block popup windows** check box. In Internet Explorer, select **Tools > Internet Options > Privacy** and disable the **Turn on Pop-up Blocker** check box.

Restoring IP Access to the Administrative Interface

The browser-based Administrative Interface can be accessed via the internal IP, the external IP, or the failover IP, using the `http://` or `https://` protocols. Settings for interface access appear in the configuration file. While the Administration Interface prevents you from disabling all access to the interface, all access can be disabled if the access settings are removed from the configuration file manually or if the configuration file becomes corrupted.

If access to the Administrative Interface is disabled on all available IP addresses and protocols, do the following to enable access again:

1. Log into Equalizer using the serial line or SSH as *root*.

2. Enter the following command exactly as shown to enable access via all IP addresses and protocols:

   `parse_config -a -H 1 -i /var/eq/eq.conf -E -I -F -p -s`

   - `-a:`   Update the Apache server configuration.
   - `-H 1:`   Restart Apache after one second (seconds must be greater than 0).
   - `-i /var/eq/eq.conf:`   Location of Equalizer's configuration file.
   - `-E -I -F:`   Start Apache on the External IP, Internal IP, and Failover IP; respectively.
   - `-p -s:`   Enable the HTTP and HTTPS protocols; respectively.

3. Running the above command *does not* update the configuration file, so access may be lost the next time the Apache server is restarted. To restore interface access in the current configuration, see the section "Managing Equalizer Access" on page 34.

Restoring Login Access to the Administrative Interface

The Administrative Interface prevents you from deleting the login that you are currently using. For example, you cannot log in as **touch** and delete the **touch** login; to delete **touch**, you must log in using a different user name that has the **add/del** permission on users. This also prevents you from deleting all logins via the interface. However, it is possible that all user logins could be deleted by manually editing the configuration file, or in the unlikely event the configuration file becomes corrupted. If this occurs, do the following:

1. Log into Equalizer using the serial line or SSH as *eqadmin* or *root*.

2. Enter:

   eqadmin

3. Select **4 Manage users** and press **Enter**.

4. Select **1 Full Access** to create an Administrator login; select **2 Read Only** to create a read-only login. Press **Enter**.

5. A series of prompts appears at the bottom of the screen. Type in the information described below at each prompt and press **Enter**:

   ```
   Enter username: <Login>
   Enter full name: <Description or Name>
   Enter password: <password>
   Enter password again: <repeat password>
   ```

6. After you press **Enter** at the final prompt above, the system should respond with:

   ```
   User <Login> created successfully.
   ```

   The **eqadmin** utility then returns to the main menu.

Note that any logins you create via **eqadmin** are auto ma ti ally added to */var/eq/eq.conf*, and will appear in the Administration Interface's **Users** table (***Equalizer* > Permissions > Users**) when you log in again.

Appendix H:

License and Warranty

SOFTWARE LICENSE

PLEASE READ THESE TERMS AND CONDITIONS CAREFULLY BEFORE USING THE SOFTWARE. BY USING THIS SOFTWARE YOU AGREE TO BE BOUND BY THE TERMS AND CONDITIONS OF THIS LICENSE. IF YOU DO NOT AGREE WITH THE TERMS OF THIS LICENSE, PROMPTLY RETURN THE UNUSED SOFTWARE, MANUAL, AND RELATED EQUIPMENT AND HARDWARE (WITH PROOF OF PAYMENT) TO THE PLACE OF PURCHASE FOR A FULL REFUND.

Coyote Point Systems, Inc. ("Coyote Point Systems") and its suppliers grant to Customer ("Customer") a nonexclusive and nontransferable license to use the Coyote Point Systems software ("Software") in object code form solely on a single central processing unit owned or leased by Customer or otherwise embedded in equipment provided by Coyote Point Systems. Customer may make one (1) archival copy of the software provided Customer affixes to such copy all copyright, confidentiality, and proprietary notices that appear on the original. EXCEPT AS EXPRESSLY AUTHORIZED ABOVE, CUSTOMER SHALL NOT COPY, IN WHOLE OR IN PART, SOFTWARE OR DOCUMENTATION; MODIFY THE SOFTWARE; REVERSE COMPILE OR REVERSE ASSEMBLE ALL OR ANY PORTION OF THE SOFTWARE; OR RENT, LEASE, DISTRIBUTE, SELL, OR CREATE DERIVATIVE WORKS OF THE SOFTWARE.

Customer agrees that aspects of the licensed materials, including the specific design and structure of individual programs, constitute trade secrets and/or copyrighted material of Coyote Point Systems. Customer agrees not to disclose, provide, or otherwise make available such trade secrets or copyrighted material in any form to any third party without the prior written consent of Coyote Point Systems. Customer agrees to implement reasonable security measures to protect such trade secrets and copyrighted material. Title to Software and documentation shall remain solely with Coyote Point Systems.

This License is effective until terminated. Customer may terminate this License at any time by destroying all copies of Software including any documentation. This License will terminate immediately without notice from Coyote Point Systems if Customer fails to comply with any provision of this License. Upon termination, Customer must destroy all copies of Software. Software, including technical data, is subject to U.S. export control laws, including the U.S. Export Administration Act and its associated regulations, and may be subject to export or import regulations in other countries. Customer agrees to comply strictly with all such regulations and acknowledges that it has the responsibility to obtain licenses to export, reexport, or import Software.

This License shall be governed by and construed in accordance with the laws of the State of New York, United States of America, as if performed wholly within the state and without giving effect to the principles of conflict of law. If any portion hereof is found to be void or unenforceable, the remaining provisions of this License shall remain in full force and effect. This License constitutes the entire License between the parties with respect to the use of the Software.

Restricted Rights - Coyote Point Systems' software is provided to non-DOD agencies with RESTRICTED RIGHTS and its supporting documentation is provided with LIMITED RIGHTS. Use, duplication, or disclosure by the U.S. Government is subject to the restrictions as set forth in subparagraph "C" of the Commercial Computer Software - Restricted Rights clause at FAR 52.227-19. In the event the sale is to a DOD agency, the U.S. Government's rights in software, supporting documentation, and technical data are governed by the restrictions in the Technical Data Commercial Items clause at DFARS 252.227-7015 and DFARS 227.7202.

LIMITED WARRANTY

This document includes Limited Warranty information for Coyote Point Systems products. For products purchased in the European Union, please refer to the European Union Amendment.

General Terms.

EXCEPT AS EXPRESSLY SET FORTH IN THIS LIMITED WARRANTY, COYOTE POINT SYSTEMS MAKES NO OTHER WARRANTIES OR CONDITIONS, EXPRESS OR IMPLIED, INCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE. COYOTE POINT SYSTEMS EXPRESSLY DISCLAIMS ALL WARRANTIES AND CONDITIONS NOT STATED IN THIS LIMITED WARRANTY. ANY IMPLIED WARRANTIES THAT MAY BE IMPOSED BY LAW ARE LIMITED IN DURATION TO THE LIMITED WARRANTY PERIOD. SOME STATES OR COUNTRIES DO NOT ALLOW A LIMITATION ON HOW LONG AN IMPLIED WARRANTY LASTS OR THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES FOR CONSUMER PRODUCTS. IN SUCH STATES OR COUNTRIES, SOME EXCLUSIONS OR LIMITATIONS OF THIS LIMITED WARRANTY MAY NOT APPLY TO YOU.

This Limited Warranty applies to the Coyote Point Systems software and hardware products sold by Coyote Point Systems, Inc., its subsidiaries, affiliates, authorized resellers, or country distributors (collectively referred to in this Limited Warranty as "Coyote Point Systems") with this Limited Warranty.

Software.

Coyote Point Systems warrants that: (i) the media on which the Software is furnished will be free of defects in materials and workmanship under normal use; and (ii) the Software substantially conforms to its published specifications during the Limited Warranty Period. Except for the foregoing, the Software is provided AS IS.

Hardware.

Coyote Point Systems warrants that the Hardware product will be free from defects in materials and workmanship under normal use during the Limited Warranty Period.

The Limited Warranty Period is for one year from the date of shipment. Your dated delivery receipt, showing the date of shipment of the product, is your proof of the shipment date. You may be required to provide proof of purchase as a condition of receiving warranty service. You are entitled to warranty service according to the terms and conditions of this document if a repair to your Coyote Point Systems software or hardware is required within the Limited Warranty Period. This Limited Warranty extends only to the original purchaser of this Coyote Point Systems product and is not transferable to anyone who obtains ownership of the Coyote Point Systems product from the original purchaser.

Coyote Point Systems products are manufactured using new materials or new and used materials equivalent to new in performance and reliability. Replacement products are guaranteed to have functionality at least equal to our published specifications. Replacement parts are warranted to be free from defects in material or workmanship for the remainder of the Limited Warranty Period. Repair or replacement of a part will not extend the Limited Warranty.

During the Limited Warranty Period, Coyote Point Systems will repair or replace the defective component parts or the hardware product. All component parts or hardware products removed under this Limited Warranty become the property of Coyote Point Systems. Coyote Point Systems, at its discretion, may elect to provide you with a replacement unit of Coyote Point Systems' choosing that is at least equivalent to your Coyote Point Systems product in hardware performance. Coyote Point Systems reserves the right to elect, at its sole discretion, to give you a refund of your purchase price instead of a replacement. This is your exclusive remedy for defective products.

To request Limited Warranty service, you must contact Coyote Point Systems Technical Support, which can be reached at (888) 891-8150 or via E-mail at support@coyotepoint.com. Coyote Point Systems Technical Support will determine the nature of the problem, and if a return is necessary, issue a Return Materials Authorization (RMA). No returned product will be accepted without an RMA number obtained in advance and clearly marked on the outside of the shipping container. All products to be returned must be in the original manufacturer's undamaged packaging

along with all accessories shipped with the original product including cables, handles and manuals. If you did not retain the original packaging materials, there may be a charge for replacement packaging.

If a defective product is returned, the cost of incoming freight and insurance is the responsibility of the customer. The cost of return freight is the responsibility of Coyote Point Systems, if shipped within the United States. Shipments to other locations will be freight collect. You are responsible for missing or physically damaged parts on the returned defective product, if they are not covered under the product Limited Warranty. You are responsible for all customs fees, taxes or VAT that may be due (excluding income taxes). A product returned for repair but found to be in good working order will be charged a $75 "No Trouble Fee".

Coyote Point Systems does not warrant that the operation of this product will be uninterrupted or error-free. Coyote Point Systems is not responsible for damage that occurs as a result of your failure to follow the instructions that came with the Coyote Point Systems product.

Restrictions.

This Limited Warranty does not extend to software errors that can not be reproduced, or for any product from which the serial number has been removed, or that has been damaged or rendered defective (a) as a result of accident, misuse, abuse, or other external causes; (b) by operation outside the usage parameters stated in the user documentation that shipped with the product; (c) by the use of parts not manufactured or sold by Coyote Point Systems; or (d) by modification or service by anyone other than (i) Coyote Point Systems, (ii) a Coyote Point Systems authorized service provider, or (iii) your own installation of end-user replaceable Coyote Point Systems or Coyote Point Systems approved parts.

COYOTE POINT SYSTEMS WILL NOT HAVE ANY LIABILITY FOR ANY DAMAGES ARISING FROM THE USE OF THE PRODUCTS IN ANY HIGH-RISK ACTIVITY, INCLUDING, BUT NOT LIMITED TO, THE OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION OR COMMUNICATION SYSTEMS, AIR TRAFFIC CONTROL, MEDICAL SYSTEMS, LIFE SUPPORT, OR WEAPONS SYSTEMS.

These terms and conditions constitute the complete and exclusive warranty agreement between you and Coyote Point Systems regarding the Coyote Point Systems product you have purchased. These terms and conditions supersede any prior agreements or representations including representations made in Coyote Point Systems sales literature or advice given to you by Coyote Point Systems or an agent or employee of Coyote Point Systems that may have been made in connection with your purchase of the Coyote Point Systems product. No change to the conditions of this Limited Warranty is valid unless it is made in writing and signed by an authorized representative of Coyote Point Systems.

Limitation of Liability.

IF YOUR COYOTE POINT SYSTEMS SOFTWARE OR HARDWARE PRODUCT FAILS TO WORK AS WARRANTED ABOVE, YOUR SOLE AND EXCLUSIVE REMEDY SHALL BE REPAIR OR REPLACEMENT (INCLUDING REFUND). COYOTE POINT SYSTEMS' MAXIMUM LIABILITY UNDER THIS LIMITED WARRANTY IS EXPRESSLY LIMITED TO THE LESSER OF THE PRICE YOU HAVE PAID FOR THE PRODUCT OR THE COST OF REPAIR OR REPLACEMENT OF ANY SOFTWARE OR HARDWARE COMPONENTS THAT MALFUNCTION IN CONDITIONS OF NORMAL USE. COYOTE POINT SYSTEMS IS NOT LIABLE FOR ANY DAMAGES CAUSED BY THE PRODUCT OR THE FAILURE OF THE PRODUCT TO PERFORM, INCLUDING ANY LOST PROFITS OR SAVINGS OR DATA, OR SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, OR PUNITIVE DAMAGES. COYOTE POINT SYSTEMS IS NOT LIABLE FOR ANY CLAIM MADE BY A THIRD PARTY OR MADE BY YOU FOR A THIRD PARTY.

THIS LIMITATION OF LIABILITY APPLIES WHETHER DAMAGES ARE SOUGHT, OR A CLAIM MADE, UNDER THIS LIMITED WARRANTY OR AS A TORT CLAIM (INCLUDING NEGLIGENCE AND STRICT PRODUCT LIABILITY), A CONTRACT CLAIM, OR ANY OTHER CLAIM. THIS LIMITATION OF LIABILITY CANNOT BE WAIVED OR AMENDED BY ANY PERSON. THIS LIMITATION OF LIABILITY WILL BE EFFECTIVE EVEN IF YOU HAVE ADVISED COYOTE POINT SYSTEMS OR AN AUTHORIZED REPRESENTATIVE OF COYOTE POINT SYSTEMS OF THE POSSIBILITY OF ANY SUCH DAMAGES. THIS LIMITATION OF LIABILITY, HOWEVER, WILL NOT APPLY TO CLAIMS FOR PERSONAL INJURY. THE FOREGOING LIMITATIONS SHALL APPLY EVEN IF THE ABOVE-STATED WARRANTY FAILS OF ITS ESSENTIAL PURPOSE.

THIS LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS. YOU MAY ALSO HAVE OTHER RIGHTS THAT MAY VARY FROM STATE TO STATE OR FROM COUNTRY TO COUNTRY. YOU ARE ADVISED TO CONSULT APPLICABLE STATE OR COUNTRY LAWS FOR A FULL DETERMINATION OF YOUR RIGHTS.

IN THE EVENT OF INCONSISTENCY BETWEEN ANY TERMS OF THIS DISCLAIMER OF WARRANTIES AND LIMITED WARRANTY AND ANY TRANSLATION THEREOF INTO ANOTHER LANGUAGE, THE ENGLISH LANGUAGE VERSION SHALL PREVAIL.

THIS DISCLAIMER OF WARRANTIES AND LIMITED WARRANTY ARE GOVERNED BY THE LAWS OF THE STATE OF NEW YORK, UNITED STATES OF AMERICA, WITHOUT REGARD TO THE CONFLICT OF LAWS PROVISIONS THEREOF.   THE UNITED NATIONS CONVENTION ON CONTRACTS FOR THE INTERNATIONAL SALE OF GOODS SHALL NOT APPLY TO THESE TERMS IN ANY RESPECT.

THIS DISCLAIMER OF WARRANTIES AND LIMITED WARRANTY ARE SUBJECT TO THE TERMS OF SALE OF THE COYOTE POINT SYSTEMS' PRODUCT.

Additional Requirements and Specifications

Short-Circuit Protection

Warning   This product relies on the building's installation for short-circuit (overcurrent) protection. Ensure that a fuse or circuit breaker no larger than 120 VAC, 15A U.S. (240 VAC, 10A international) is used on the phase conductors (all current-carrying conductors).

Attention   Pour ce qui est de la protection contre les courts-circuits (surtension), ce produit dépend de l'installation électrique du local. Vérifier qu'un fusible ou qu'un disjoncteur de 120 V alt., 15 A U.S. maximum (240 V alt., 10 A international) est utilisé sur les conducteurs de phase (conducteurs de charge).

Warnung   Dieses Produkt ist darauf angewiesen, daß im Gebäude ein Kurzschluß- bzw. Überstromschutz installiert ist. Stellen Sie sicher, daß eine Sicherung oder ein Unterbrecher von nicht mehr als 240 V Wechselstrom, 10 A (bzw. in den USA 120 V Wechselstrom, 15 A) an den Phasenleitern (allen stromführenden Leitern) verwendet wird.

Power Supply Cord

CAUTION: THE POWER SUPPLY CORD IS USED AS THE MAIN DISCONNECT DEVICE, ENSURE THAT THE SOCKET-OUTLET IS LOCATED/INSTALLED NEAR THE EQUIPMENT AND IS EASILY ACCESSIBLE.

ATTENTION: LE CORDON D'ALIMENTATION EST UTILISÉ COMME INTERRUPTEUR GÉNÉRAL. LA PRISE DE COURANT DOIT ÊTRE SITUÉE OU INSTALLÉE À PROXIMITÉ DU MATÉRIEL ET ÊTRE FACILE D'ACCÉS.

Warnung: Das Netzkabel dient als Netzschalter. Stellen Sie sicher, das die Steckdose einfach zugänglich ist.

Installation into an Equipment Rack

When operating the unit in an equipment Rack, take the following precaution:

- Make sure the ambient temperature around the unit (which may be higher than the room temperature) is within the limit specified for the unit.

- Make sure there is sufficient airflow around the unit.

- Make sure electrical circuits are not overloaded - consider the nameplate rating of all the connected equipment, and make sure you have over current protection.

- Make sure the equipment is properly grounded.

- Make sure no objects are placed on top of the unit.

Chassis Warning—Rack-Mounting and Servicing

Warning   To prevent bodily injury when mounting or servicing this unit in a rack, you must take special precautions to ensure that the system remains stable. The following guidelines are provided to ensure your safety:

- This unit should be mounted at the bottom of the rack if it is the only unit in the rack.
- When mounting this unit in a partially filled rack, load the rack from the bottom to the top with the heaviest component at the bottom of the rack.
- If the rack is provided with stabilizing devices, install the stabilizers before mounting or servicing the unit in the rack.

Attention   Pour éviter toute blessure corporelle pendant les opérations de montage ou de réparation de cette unité en casier, il convient de prendre des précautions spéciales afin de maintenir la stabilité du système. Les directives ci-dessous sont destinées à assurer la protection du personnel :

- Si cette unité constitue la seule unité montée en casier, elle doit être placée dans le bas.
- Si cette unité est montée dans un casier partiellement rempli, charger le casier de bas en haut en plaçant l'élément le plus lourd dans le bas.
- Si le casier est équipé de dispositifs stabilisateurs, installer les stabilisateurs avant de monter ou de réparer l'unité en casier.

Warnung   Zur Vermeidung von Körperverletzung beim Anbringen oder Warten dieser Einheit in einem Gestell müssen Sie besondere Vorkehrungen treffen, um sicherzustellen, daß das System stabil bleibt. Die folgenden Richtlinien sollen zur Gewährleistung Ihrer Sicherheit dienen:

- Wenn diese Einheit die einzige im Gestell ist, sollte sie unten im Gestell angebracht werden.
- Bei Anbringung dieser Einheit in einem zum Teil gefüllten Gestell ist das Gestell von unten nach oben zu laden, wobei das schwerste Bauteil unten im Gestell anzubringen ist.
- Wird das Gestell mit Stabilisierungszubehör geliefert, sind zuerst die Stabilisatoren zu installieren, bevor Sie die Einheit im Gestell anbringen oder sie warten.

Battery

A lithium battery is included in this unit. Do not puncture, mutilate, or dispose of the battery in a fire. There is a danger of explosion if the battery is incorrectly replaced. Replace only with the same or equivalent type, as recommended by the manufacturer. Dispose of a used battery according to the manufacturer's instructions and in accordance with your local regulations.

Specifications

Power Requirements

The unit's power supply is rated at **100-240 VAC** auto selecting **60/50 Hz @ 4.0A**.

Power Consumption

Use the following power consumption information to determine how many units can be connected to available power circuits without overload. The information shown in the tables below was captured during the following operational stages of the product:

- **Rush-in** current -- when the product is powered ON

- **No Load** -- when the product is booted from OS but no resource-hungry process is running

- **100% CPU** -- when 100% processor load is emulated on the product

The following data is captured during the test, at both 110V and 220V:

- **Watts** -- total power consumed by product

- **PF/VA** -- Power Factor in Volt-Amps (a ratio of the real power and apparent power consumed by the product)

- **V/KHz** -- Voltage in kilohertz

- **Amp** -- total current consumed by product

110V Test Results

Model	110V/60Hz	Watts	PF/VA	Volts	Amps
E650GX					
	Rush-in	112.5	1.000	118.9	0.954
	No Load	112.2	1.000	118.7	0.943
	100% CPU	145.0	1.000	118.2	1.222
E450GX					
	Rush-in	112.5	1.000	118.9	0.954
	No Load	112.2	1.000	118.7	0.943
	100% CPU	145.0	1.000	118.2	1.222
E350GX					
	Rush-in	76.7	0.993	119.9	0.644
	No Load	70.9	0.992	119.7	0.597
	100% CPU	99.5	1.000	119.6	0.831
E250si					
	Rush-in	52.3	0.933	120	1.1
	No Load	41.7	0.901	120	0.39
	100% CPU	46.8	0.924	120	0.43

220V Test Results

Model	220V/50Hz	Watts	PF/VA	Volts	Amps
E650GX					
	Rush-in	109.1	0.645	224	0.752
	No Load	109.9	0.925	222	0.536
	100% CPU	140.5	0.943	222	0.671
E450GX					
	Rush-in	109.1	0.645	224	0.752
	No Load	109.9	0.925	222	0.536
	100% CPU	140.5	0.943	222	0.671
E350GX					
	Rush-in	74.6	0.877	445	0.378
	No Load	68.5	0.862	225	0.354
	100% CPU	96.3	0.923	224	0.466
E250si					
	Rush-in	45.9	0.823	220	0.258
	No Load	39.1	0.781	220	0.226
	100% CPU	43.9	0.795	220	0.249

Operating Environment

- **Temperature**: 40 - 105 °F, 5 - 40 °C.
- **Humidity**: 5 - 90%, non-condensing.

Physical Dimensions

Model	Weight	Height	Width	Depth
E250si	10.9 lbs.	1.75 in.	19 in.	11.75 in.
E350GX / E450GX / E650GX	15 lbs.	1.75 in.	19 in.	15.75 in.

Regulatory Certification

Please see the product data sheets on the Coyote Point Website (`www.coyotepoint.com`) for product certification details.

Chapter J:

Glossary

This glossary defines some of the key terms used in this document. Some of the glossary definitions are based on RFC1208, "A Glossary of Networking Terms."[1]

active content verification (ACV)	Active Content Verification; an Equalizer mechanism for checking the validity of a server. ACV does not support UDP-based services.
administration address	The IP address assigned to Equalizer on the internal network. *See* internal network and IP address.
administration interface	The browser-based interface for setting up and managing the operation of Equalizer.
address translation	The modification of external addresses to standardized network addresses and of standardized network addresses to external addresses.
agent	An application that gathers or processes information for a larger application. *See* server agent.
aggregation	A summary of all the data that is computed from detailed information. *See* sticky network aggregation.
alias	A nickname that replaces a long name or one that is difficult to remember or spell. *See* IP alias.
aliased IP address	A nickname for an IP address. *See* IP alias.
algorithm	Instructions, procedures, or formulas used to solve a problem.
application layer	Layer 7 (L7); the highest layer of standards in the Open Systems Interconnection (OSI) model (according to The MIcrosoft Press *Computer Dictionary*), which helps a user perform work such as transferring files, formatting e-mail messages, and accessing remote computers.
atom	The smallest part of a regular expression in Equalizer. *See* branch, piece, and regular expression.
authoritative name server	A name server that maintains the complete information for a particular part of the domain name space. *See* name server.
back-end server	A physical server on the internal network that receives connection requests from Equalizer.
backup Equalizer	The backup unit, which replaces the primary Equalizer if that Equalizer fails. *See* hot backup and primary Equalizer.
bound	A character that represents the limit of part of a regular expression.
bracket expression	In a regular expression, a list of characters enclosed in brackets ([...]).
branch	In an Equalizer regular expression, a complete piece of a regular expression. You can concatenate and/or match branches. *See* atom, piece, and regular expression.
cache	An area in which information is temporarily stored.

Class A	An ISO/IEC 11801 standard for twisted pair cabling rated to 100 KHz; similar to Category 1 cabling. Use the Class A standard for voice and low frequency applications. According to the Microsoft Press *Computer Dictionary,* you can use Class A networks "for sites with few networks but numerous hosts." *See* ISO/IEC.
Class B	An ISO/IEC 11801 standard for twisted pair cabling rated to 1 MHz; similar to Category 2 cabling. Use the Class B standard for medium bit rate applications. *See* ISO/IEC.
Class C	An ISO/IEC 11801 standard for twisted pair cabling rated to 16 MHz; similar to TIA/EIA Category 3 cabling. Use the Class C standard for high bit rate applications, in which the network allocates 24 bits for the IP address network-address field. A Class C network allocates 24 bits for the IP address network-address field and 8 bits for the host field. *See* ISO/IEC.
cluster	A set of networked computer systems that work together as one system. *See* server cluster and virtual cluster.
cluster address	The IP address assigned to a particular cluster configured on Equalizer.
computed load	A measure of the performance of a server relative to the overall performance of the cluster of which the server is a part.
cookie	Data that a Web server stores on a client on behalf of a Web site. When a user returns to the Web site, the server reads the cookie data on the client, providing the Web site all the saved information about the user.
daemon	An application that runs in the background and performs one or more actions when events trigger those actions.
DNS	Domain Name System or Domain Name Service; used to map domain names to Internet servers in order to link to IP addresses or map IP addresses to domain names. *See* IP address.
DNS TTL	The amount of time, in seconds, that a name server is allowed to cache the domain information. *See* DNS and TTL.
domain	The highest level in an IP address and the last part of the address in the URL. The domain identifies the category under which the Web site operates. For example, in `www.coyotepoint.com`, `com` is the domain, where `com` represents a *commercial* site. *See* domain name, IP address, and subdomain. *See also* DNS.
domain name	The owner of an IP address. The next highest level in an IP address and the next-to-last part of the address. For example, in `www.coyotepoint.com`, `coyotepoint` is the domain name. *See* domain, IP address, and subdomain. *See also* DNS.
dynamic weight	The weight that Equalizer assigns to a particular server during operation. *See* server weight, initial weight, and weight.
echo	The transmittal of data that has been sent successfully back to the originating computer. *See* ping. *See also* CMP echo request.
edit mode	One of two modes in which Equalizer can be administered. In edit mode, you can view and modify parameters. *See* view mode.
EIA	Electronic Industries Association; a trade association that sets standards for electrical and electronic components.
endpoint	An IP address-port pair that identifies the start or end of an address; a value that ends a process.

---

1. O. Jacobsen and D. Lynch, Interop, Inc. March 1991.

Envoy	Equalizer add-in; software that supports geographic clustering and load balancing. *See* geographic cluster, geographic load balancing, and load balancing. *See also* intelligent load balancing.
Equalizer Administration Interface	An Equalizer window with which you can monitor Equalizer's operation; view statistics; add, modify, or clusters; add, modify, and delete servers; and shut down a server or Equalizer through a Javascript-enabled browser.
Equalizer Configuration Utility	An Equalizer feature that enables you to configure Equalizer, set parameters, and shut down and upgrade Equalizer.
external address	The IP address assigned to Equalizer on the external network.
external interface	A network interface used to connect Equalizer to the external network. *See* interface, internal interface, and network interface.
external network	The subnet to which the client machines and possibly the Internet or an intranet are connected.
failover	The act of transferring operations from a failing component to a backup component without interrupting processing.
firewall	A set of security programs, which is located at a network gateway server and which protect the network from any user on an external network. *See* gateway.
FQDN	*See* Fully Qualified Domain Name (FQDN).
FTP	File Transfer Protocol; rules for transferring files from one computer to another.
FTP cluster	A virtual cluster providing service on the FTP control port (port 21). *See* cluster and virtual cluster.
Fully Qualified Domain Name (FQDN)	The complete, registered domain name of an Internet host, which is written relative to the root domain and unambiguously specifies a host's location in the DNS hierarchy. For example, `east` is a hostname and `east.coyotepoint.com` is its fully qualified domain name. *See also* domain name.
gateway	A network route that typically translates information between two different protocols.
geographic cluster	A collection of servers (such as Web sites) that provide a common service over different physical locations. *See* cluster.
geographic load balancing	Distributing requests as equally as possible across servers in different physical locations. *See* load balancing. *See also* intelligent load balancing.
geographic probe	A query sent to a site in a geographic cluster to gather information so Equalizer can determine the site that is best able to process a pending request. *See* geographic cluster.
header	One or more lines of data that identify the beginning of a block of information or a file.
hot backup	Configuring a second Equalizer as a backup unit that will take over in case of failure. Also known as a hot spare. *See* backup Equalizer. *See also* primary Equalizer. A server can also be used as a hot backup, or hot spare, within a cluster. If all the other servers in the cluster fail, the hot spare will begin processing requests for the cluster.
HTTP	HyperText Transfer Protocol; the protocol with which a computer or user access information on the World Wide Web.

HTTPS	HyperText Transfer Protocol (Secure); a server application programmed to run under the Windows NT operating system.
hub	A device that joins all the components attached to a network.
ICMP	See Internet Control Message Protocol.
ICMP echo request	The act of repeating a stream of characters (for example, echoing on the computer screen characters as a user types those characters). *See* ping. *See also* echo.
ICMP triangulation	Routing client requests to the closest site geographically based on triangulation, a method of calculating the location of a site using the known locations of two or more other sites.
initial weight	The weight that an administrator assigns to a particular server. During operation, Equalizer dynamically adjusts the server weights (that is, dynamic weight), so a server's weight at a particular time might be different from the initial weight originally set by the administrator. *See* dynamic weight, server weight, and weight.
intelligent load balancing	A request for load balancing using Equalizer-based algorithms that assess the configuration options set for cluster and servers, real-time server status information, and information in the request itself. *See* algorithm and load balancing. *See also* geographic load balancing.
interface	The place at which two or more systems connect and communicate with each other. *See* external interface, internal interface, and network interface.
internal address	The IP address assigned to Equalizer on the internal network.
internal network	The subnet to which the back-end server machines are connected.
Internet Control Message Protocol (ICMP)	The ISO/OSI Layer 3, Network, protocol that controls transport routes, message handling, and message transfers during IP packet processing. *See* ICMP triangulation and ISO/OSI model.
IP	Internet protocol; the TCP/IP protocol that controls breaking up data messages into packets, sending the packets, and reforming the packets into their original data messages. *See* Internet protocol stack, IP address, packet, and TCP/IP.
IP address	A 32-bit address assigned to a host using TCP/IP. IP addresses are written in dotted decimal format, for example, `192.22.33.1`.
ISO/IEC	International Organization for Standardization/International Electrotechnical Commission; international standards organizations.

ISO/OSI model	International Organization for Standardization/Open Systems Interconnection model, a standard that consists of seven layers that control how computers communicate with other computers over a network.
	• Layer 1, Physical, which sets the rules for physical connections via hardware, is the lowest layer.
	• Layer 2, Data-link, uses Layer 1 and its own rules to control coding, addressing, and transmitting information.
	• Layer 3, Network, uses the prior two layers rules as well as its own rules to control transport routes, message handling, and message transfers.
	• Layer 4, Transport, uses its rules and those of the previous layers to control accuracy of message delivery and service.
	• Layer 5, Session, uses its rules and those of the previous layers to establish, maintain, and coordinate communication.
	• Layer 6, Presentation, uses its rules and those of the previous layers to control text formatting and appearance as well as conversion of code.
	• Layer 7, Application, uses its rules and those of the other layers to control transmission of information from one application to another. Layer 7 is the highest layer.
	See Layer 4, Layer 7, and transport layer.
L4	*See* Layer 4.
L7	*See* Layer 7.
latency	The time over which a signal travels over a network, from the starting point to the endpoint. *See* ping. *See also* CMP echo request and echo.
Layer 4 (L4)	The transport layer; Layer 4 uses its rules and those of the previous three layers to control accuracy of message delivery and service.which controls accuracy of message delivery and service. *See* ISO/OSI model and Layer 7.
Layer 7 (L7)	The application layer; Layer 7 uses its rules and those of the other layers to control transmission of information from one application to another. Layer 7 is the highest layer in the ISO/OSI model. *See* ISO/OSI model and Layer 4.
load	A job that can be processed or transported once. *See* load balancing. *See also* geographic load balancing and intelligent load balancing.
load balancing	Moving a load from a highly-used resource to a resource that is used less often so that operations are efficient. Equalizer balances loads over a wide physical area or by using algorithms that assess options and real-time information. *See* geographic load balancing and intelligent load balancing.
MX exchanger	Mail exchanger; a fully qualified domain name to be returned if a server receives a mail exchanger request.
name server	A server that stores information about the domain name space.
NAT	Network Address Translation; an Internet standard that defines the process of converting IP addresses on a local-area network to Internet IP addresses. *See* NAT subsystem.
NAT subsystem	The Equalizer subsystem responsible for transferring connections to and from the back-end servers.

netmask	Address mask; a bit mask used to select bits from an Internet address for subnet addressing. The mask is 32 bits long and selects the network portion of the Internet address and one or more bits of the local portion.
Network Address Translation (NAT)	*See* NAT.
network interface	The place at which two or more networks connect and communicate with each other. *See* interface. *See* external interface, interface, and internal interface.
network route	*See* gateway.
OSI network	A network that uses the International Organization for Standardization/Open Systems Interconnection model. *See* ISO/OSI model, Layer 4, Layer 7, and transport layer.
packet	A group of data that is transmitted as a single entity.
passive FTP connection	An Equalizer option that rewrites outgoing FTP PASV control messages from the servers so that they contain the IP address of the virtual cluster rather than that of the server. *See* FTP and PASV.
PASV	Passive mode FTP; a mode with which you can establish FTP connections for clients that are behind firewalls. *See* firewall, FTP, and passive FTP connections.
pattern match	A pattern of ASCII or hexadecimal data that filters data.
payload	The set of data to be transmitted. A payload contains user information, user overhead information, and other information that a user requests. A payload *does not* include system overhead information. Also known as the mission bit stream.
persistence	The act of storing or retaining data for use at a later time, especially data that shows the state of the network before processing resumes. *See* cookie and IP-address-based persistence.
physical server	A machine located on the internal network that provides services on specific IP addresses and ports. *See* server and virtual web server. *See also* authoritative name server, back-end server, name server, and proxy server.
piece	An atom followed by a single *, +, or ?, or by a bound. *See* atom, branch, and regular expression.
ping	A program used to test reachability of destinations by sending them an ICMP echo request and waiting for a reply. *See* echo and probe. *See also* CMP echo request
port	The abstraction used by Internet transport protocols to distinguish among multiple simultaneous connections to a single destination host.
port number	The number used to identify a service contact port, such as HTTP port 80.
primary Equalizer	The primary unit that handles requests. If the primary Equalizer fails, the backup unit replaces it. *See also* backup Equalizer and hot backup.
probe	An action that obtains status information about a computer, network, or device. *See* geographic probe and ping.
protocol	A set of rules that govern adherence to a set of standards. *See* protocol stack.
protocol stack	A layer of protocols that process network actions cooperatively and in tandem. *See* protocol.

proxy server	A utility, which is part of a firewall, that helps the regular tasks of managing data transmittal from a network to the Internet and from the Internet to the network. *See also* firewall.
quiesce	To become quiet or more quiet than previously.
RADIUS	Remote Authentication Dial-In User Service; a protocol that authorizes and authenticates a user trying to link to a network or the Internet.
redirection	The process of receiving input from or sending output to a different resource than usual.
regular expression (RE)	One or more non-empty branches, separated by pipe symbols (|). An expression matches anything that matches one of the branches. *See* atom, branch and piece.
request packet	A packet that contains information that requests a response. *See* packet and response packet.
reserved network	A network consisting of "phony" IP addresses, which are not registered and cannot be made visible outside of the internal network.
resolution	The process of interpreting all the messages between an IP address and a domain name address.
response packet	A packet that contains information that responds to a request. *See* packet and request packet.
round robin	The default load balancing policy which distributes requests equally among all servers in a virtual cluster, without regard to initial weights or adaptive load balancing criteria. The first request received is routed to the first server in the list, the second request to the second server, and so on. When the last server is reached, the cycle starts again with the first server.
router	A network device that facilitates the transmission (that is, *routing*) of messages.
routing table	A database, which is static or dynamic, that contains a set of route addresses and routing information familiar to the router. A human being enters and updates the information in a static routing table; routers operate and constantly update a dynamic routing table.
RST	The reset command, which instructs a device to end a connection.
Secure Sockets Layer (SSL)	A protocol, which uses public-key encryption, that enables secure communications between a client and Web server, typically for guarding financial transactions.
server	A computer or application that controls access to a network and its associated devices and applications. A server communicates with one or more clients as well as other servers. *See* authoritative name server, back-end server, name server, physical server, proxy server, and virtual web server.
server address	The IP address of a server on the internal interface. Multiple IP addresses can be aliased to a single physical server. *See* server.
server agent	An agent that provides Equalizer with real-time performance statistics for a specified server. *See* server.
server cluster	A group of servers that are components in a network and joined through hardware or software. *See* cluster. *See also* FTP cluster, geographic cluster, and virtual cluster. *See* server.
server endpoint	An IP address-port pair that identifies a physical or virtual server on the internal network to which Equalizer can route connection requests. *See* server.

server weight	A value that indicates the relative proportion of connection requests that a particular server will receive. *See* dynamic weight, server, initial weight, and weight.
session	A logical connection between a server and a client that spans a series of individual client requests and server responses. The persistence of session data is maintained through the exchange of cookies in Layer 7, or through the sticky connections feature in Layer 4.
site	A cluster of servers under Equalizer control that is part of a geographic cluster.
spoofing	Using the client's IP address for the source IP address in client requests. This fools (or spoofs) the server into regarding the client as the source of the request. For spoofing to work, the default gateway for the server must be set to Equalizer's internal IP address.
SSL	*See* Secure Sockets Layer (SSL).
stack	An area of reserved memory in which applications place status data and other data. *See* protocol stack.
stale connection	A partially open or closed connection.
state	Status; the current condition of a network, computer, or peripherals.
stateless	A condition in which a server processes each request from a site independently and cannot store information about prior requests from that site. Each request stands on its own. *See also* DNS and RADIUS.
sticky connection	A connection in which a particular client remains connected to same server to handle subsequent requests within a set period of time.
sticky timer	A countdown timer that tracks periods of inactivity between a particular client and server.
subdomain	A section, which is formally named, that is under a domain name; analogous to the relationship between a subfolder and folder. For example, in `www.coyotepoint.com`, `www` is the subdomain. *See* domain, domain name, and IP address. *See also* DNS.
subnet	Part of a network that has the same address as the network plus a unique subnet mask.
switch	A device, which is attached to a network and which controls the route over which data is sent.
SYN/ACK	Synchronize and acknowledge; a message that synchronizes a sequence of data information and acknowledges the reception of that information.
syslog	A system log file, in which information, warning, and error messages are stored in a file, sent to a system, or printed.
TCP	Transmission Control Protocol; the rules for the conversion of data messages into packets. *See* ISO/OSI model, Layer 4, packet, transport layer.
TCP/IP	Transmission Control Protocol/Internet Protocol; the rules for transmitting data over networks and the Internet.
Telnet	Part of TCP/IP, a protocol that enables a user to log onto a remote computer connected to the Internet. *See* TCP/IP.
traceroute	A utility that shows the route over which a packet travels to reach its destination.

Transmission Control Protocol (TCP)	*See* TCP.
Transmission Control Protocol/Internet Protocol (TCP/IP)	*See* TCP/IP.
transport layer	*See* Layer 4. *See also* ISO/OSI model.
TTL	Time-to-live, the length of time, in seconds, that a client's DNS server should cache a resolved IP address.
User Datagram Protocol (UDP)	Within TCP/IP, a protocol that is similar to Layer 4 (the transport layer). UDP converts data into packets to be sent from one server to another but does not verify the validity of the data. *See* ISO/OSI, TCP/IP, and transport layer.
view mode	One of two modes in which Equalizer can be administered: edit and view. In view mode, you can view—but not edit—parameters. *See* edit mode.
virtual cluster	An endpoint that acts as the network-visible port for a set of hidden back-end servers. *See* cluster, endpoint, FTP cluster, geographic cluster, and server cluster.
virtual server address	An IP address that is aliased to a physical server that has its own, separate IP address. *See* virtual web server.
virtual web server	Software that imitates HTTP server hardware. A virtual web server has its own domain name and IP address. *See* domain name, HTTP, IP address, server, and virtual server address. *See also* authoritative name server, back-end server, name server, physical server, and proxy server.
WAP	*See* Wireless Application Protocol.
weight	The relative proportion of a single item in a population of similar items. *See* dynamic weight, server weight, and initial weight.
Wireless Application Protocol (WAP)	A set of rules that govern access to the Internet through wireless devices such as cellular telephones, pagers, and two-way communication devices.

Index

protocol 101
request 160
HTTPS 6, 95, 108, 270
custom headers 95
header insertion 95
request 160
hub 53, 270
HyperText Transfer Protocol (Secure). See HTTPS.
HyperText Transfer Protocol. See HTTP.

I
ICMP
drop redirects 51
ICMP ECHO request 149
ICMP echo request 189, 194, 270
ICMP echo request packet 29
ICMP echo response packet 29
ICMP probe 48
ICMP triangulation 190, 198, 270
ICMP Triangulation checkbox 198
idle timeout 50
ignore case 51
initial configuration 23
initial weight 3
installing
Envoy 192
latest Equalizer software 26
intelligent load balancing 270
inter-cluster stickiness 89
interface 270
administration 267
Equalizer Administration 31, 33, 54, 57, 269
external 10, 269
network 272
single-network 12
interface ports 68
interfaces
character-based 23
Interfaces option 24
internal
address 28, 29, 270
network 11, 12, 270
internal interface parameters 24
internal-network test machine 30
Internet 11
Internet Control Message Protocol (ICMP) 270
Internet Information Services (IIS)
certificates 236
intranet 11
IP 270
IP address 11, 24, 30, 270
reserved 15
IP Address field 24
IP address, aliased 267
IP headers 4
IP spoofing 100
ISO/IEC 270

ISO/IEC 11801 standard 268
ISO/OSI model 271

J
Javascript-enabled web browser 31

K
kernel, Equalizer 15

L
L4. See Layer 4 (L4).
L7. See Layer 7 (L7).
latency 7, 271
layer
Secure Sockets 273
Layer 4 (L4) 108, 271
cluster 88, 95, 108
Layer 4 load balancing 50
Layer 7 (L7) 2, 6, 108, 271
cluster 95, 108
load balancing 160
rules 160
Layers 1, 2, 3, 5, and 6 271
license 44, 259
licensing 44
load 271
computed 268
load balancing 161, 271
adaptive 86, 87, 198
aggressive 87
geographic 2, 7, 29, 269
geographic cluster 196
intelligent 270
Layer 4 50
Layer 7 (L7) 160
methods 86
options 86
policy 79, 86, 106
response 197
responsiveness 79
round robin 86, 106
round trip 198
site load 198
site weight 198
static weight 86, 106
WAP gateways 3
load balancing algorithms 7
Load Balancing Response option 197
load distribution, geographic 7
local name server 8
logging into
Equalizer console 23
logical AND 163
logical NOT operator 163
logical OR 163
login
Equalizer Administration interface 31

adjusting server 105
dynamic 87, 106
initial 3
oscillations 87
server 106, 274
spread coefficient 87
static 270
weight site parameter 202
Weight Spread Coefficient option 87
weighted round robin load balancing 86
window
Configure Network Interfaces 24
Equalizer Configuration Menu 24, 25
Network Configuration 24
Wireless Application Protocol (WAP) 275
wireless application protocol (WAP) 3
workload
relative 149
writing
custom agents 206

X
XCEL card 137
XCEL SSL accelerator card 238

Z
zone file 192

Equalizer Installation and Administration Guide