Proto Balance – Enterprise Load Balancer

http://www.protonet.co.za/

White Paper

Proto Balance - Enterprise Load Balancer

March 13, 2010

Contents

1 Overview

Proto Balance provides a method of spreading the load of a large number of clients across many servers. Like hardware load-balancing, it accepts many connections on a single IP address and redirects each client connection to a node in a cluster of services. However, where hardware load balancers usually work on the packet level and are expensive appliances, Proto Balance provides the same functionality on the TCP socket level, allowing Proto Balance to run as a light-weight software package on any existing hardware/OS combination.

2 Connection Forwarding

Proto Balance is meant to operate as the front-end server to a number of instances of a service within a cluster. Clients connect directly to Proto Balance. Their connections are transparently forwarded to an instance chosen by Proto Balance in a way that evenly distributes load across all instances.

Proto Balance does not change the contents of the traffic it is forwarding, but transparently forwards traffic between the client and the service instance so that both are unaware of the intermediary. In this way, Proto Balance is able to handle any type of service.

Some of the services Proto Balance is able to handle are:

* HTTP
* SMTP
* IMAP
* POP3
* IIOP
* T3
* Passive FTP

3 Client Affinity

Proto Balance can be configured to forward the same client to the same instance every time that client reconnects. It does this by recording the clients' IP addresses. This is useful when client's state information does not migrate easily between instances of your cluster. Proto Balance will faithfully ensure that a client connects to the same instance even if the client has been disconnected for a long period of time.

4 Load Balancing

When deciding what instance (i.e. node) to choose in the cluster, several load balancing algorithms are available. Proto Balance supports the following load balancing algorithms:

1. Round robin
2. Pure random
3. Connection-rate based
4. Traffic-rate based

5. By minimum load, using a custom load rating function

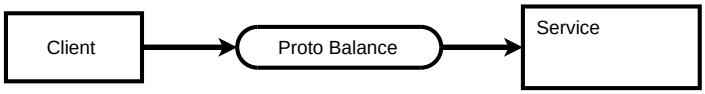
6. Random, based on a cryptographic hash of the clients IP address

The last algorithm ensures that a particular client will always connect to the same instance even if the Client Affinity feature is disabled. This algorithm is useful when you would like to ensure that multiple deployments of Proto Balance have parallel behavior with respect to clients.

5 Deployment Scenarios

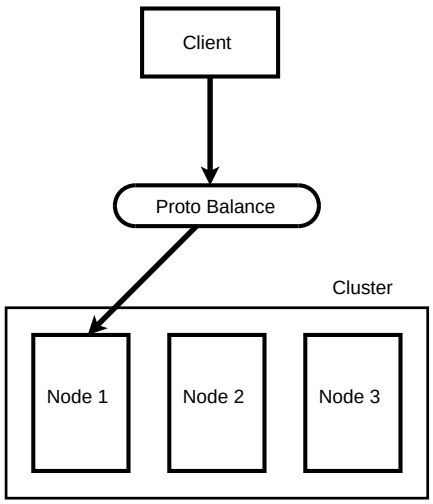
There are many ways of using Proto Balance. In the simplest configuration Proto Balance can forward from a single TCP port to another port on a remote machine. This provides no load balancing or fall-over safety but is useful where you would like to forward a TCP connection through an unroutable network, across a firewall, or just point a client to a different IP address that it would otherwise not be able to easily connect to.

Most importantly, this is useful where you would like to reconfigure the connection on the fly. This means: for any client where you would like to avoid service interruption, Proto Balance allows a dynamic centrally located point of redirection that is independent of your client or server.



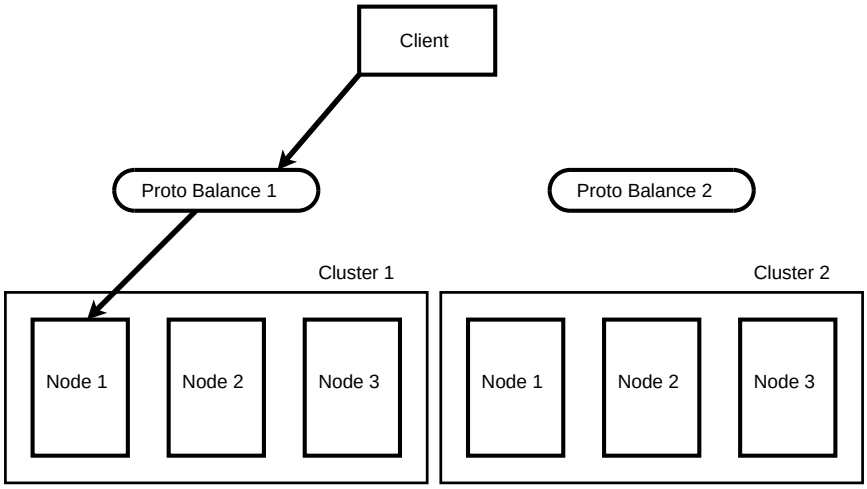
5.1 Single cluster deployment

The simplest cluster environment consists of a single cluster of several nodes. The client connects through Proto Balance to the service. The total load is distributed across all nodes in the cluster according to the chosen load-balancing algorithm. Such a deployment also provides fall-over safety should individual nodes fail. However, this environment does not provide fall-over safety against failure of the hardware on which Proto Balance itself runs.



5.2 Twin cluster deployment

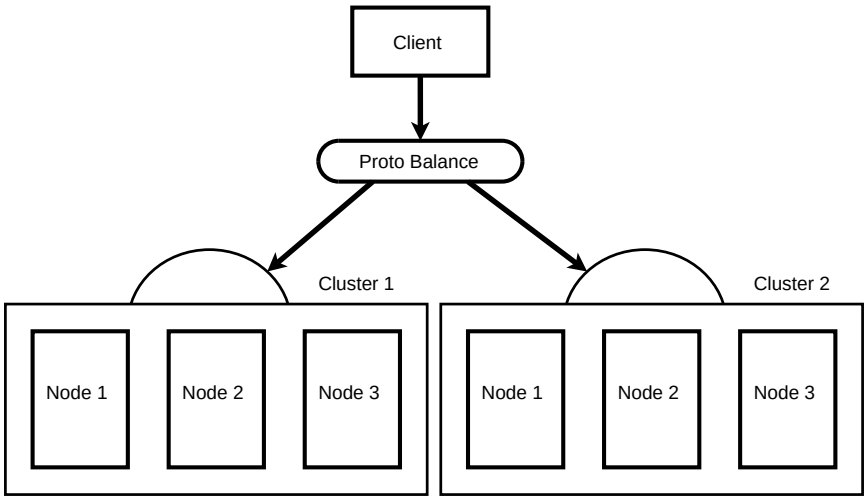
Multiple instances of Proto Balance can be deployed in parallel to provide fall-over safety, even in the case of a Proto Balance instance failing. In such an environment, the clients must be aware of both Proto Balance instances. This can be done by a round robin DNS server or via some application specific connection logic.



5.3 Controller deployment

Often you would like to be able to redirect your clients to a different deployment without changing your network or client configuration. Proto Balance allows a central, on-the-fly point of configuration of all your clients. The Controller Deployment would be used if you already have load balancing software deployed as a cluster, and would like to redeploy your entire cluster without interrupting service.

The Controller Deployment allows you to shutdown any one cluster without interrupting service. This is done by first using Proto Balance to redirect clients away from the cluster earmarked for shutdown. The configuration can be reverted once the cluster has been restored.



6 Uneven Load Balancing for More Powerful Nodes

Proto Balance allows you to specify the Load Factor of a node. The Load Factor means the proportion of the load handed to a node. The makes it easy to give a pro-rata share of the load to nodes that have more RAM or CPU, and likewise to be nicer to low-power nodes.

7 Transparent Fall-over

If Proto Balance cannot connect to a node, and if a fall-over node has been configured, then Proto Balance will transparently connect to the fall-over node in preference to the unavailable node.

A node is considered to be unavailable if the TCP connection fails for any reason before any data has been sent to the node. The logic of this comes from the fact that a node should not kill the TCP connection before having received any data - if it does, it is considered to be presently unavailable and the fall-over node is used instead.

Proto Balance also has a configurable timeout option indicating the maximum time for a connection to be established. After this time is exceeded, the node is considered to be unavailable.

Aside from this delay, clients will not be aware that they have been redirected to a fall-over node - i.e. the fall-over is seamless.

Unavailable nodes are retried periodically. The time period between retries is configurable.

Multiple fall-over nodes can be configured in a chain.

8 Transparent Shutdown

A node can manually be set to the "Preparing for shutdown" state if you are intending to take the node offline at a later stage. Clients will transparently be redirected to the fall-over node or another node in the cluster.

A node can be manually set to one of three states:

1. Available (Green). Available nodes accept new connections normally.
2. Preparing for shutdown (Yellow). Yellow nodes do not accept new connections - connections revert to other nodes in the cluster (preferring the fall-over node before others). Yellow nodes continue to service any current connections for as long as both the client and server chooses to keep the connection open.
3. Offline (Red). Switching a node to "Offline" immediately kills all client connections and accepts no new connections.

9 Proto Balance Resource Consumption

Proto Balance makes careful use of operating system resources to provide the best possible performance using the minimum of operating system threads, file descriptors, RAM, and CPU. Proto Balance will never launch an excess of operating system processes and will also effectively use as many CPU's as are available.

Some software load balancers launch a new thread or process for each client connection. Others handle all clients from a single thread or process. Proto Balance employs neither extreme enabling it to effectively

handle tens of thousands of concurrent client connections without causing undue load on your operating system environment.

10 Proto Balance Footprint and Startup-time

A typical Proto Balance memory footprint at startup is in the order of a few megabytes. Proto Balance uses in the order of tens of kilobytes of RAM for each concurrent client connection. For most configurations Proto Balance starts up and initializes in well under a second. Proto Balance properly releases unused resources and should operate indefinitely without a restart.