HAProxy Load balancer and proxy

CONCEPTS

Always run haproxy -c before a reload: a bad config keeps the old process serving.

frontend bind, ACLs
↓ use_backend
backend balance, checks
↓ balance
server app instance

A request flows through frontend rules to a backend

Runtime changes via the socket are lost on reload; persist them in the config.

PROCESS CONTROL

haproxy -c -f *<file>*
Validate config and exit

haproxy -f *<file>*
Start with a config file

haproxy -D -f *<file>*
Fork into the background

haproxy -W -f *<file>*
Master-worker mode

haproxy -sf *<pid>*
Reload; old process stops gracefully

kill -USR2 *<master-pid>*
Reload in master-worker mode

RUNTIME API

stats socket /run/haproxy.sock mode 600 level admin
Enable the socket (global section)

echo "*<command>***" | socat /run/haproxy.sock stdio**
Run one command via socat

show info
Version, uptime, load, limits

show stat
CSV stats for all proxies

SERVERS AND LIMITS

disable server *<be>*/*<srv>*
Put server in maintenance

enable server *<be>*/*<srv>*
Bring server back

set server *<be>*/*<srv>* **state drain**
Stop new sessions, keep persisted

set server *<be>*/*<srv>* **weight** *<n>*[%]
Change load share

CONFIG BASICS

global
Process-wide settings section

defaults
Shared mode and timeouts

frontend *<name>*
Where clients connect

backend *<name>*
Pool of servers

bind :*<port>*
Accept connections on a port

mode http
Layer 7 proxying

server *<name>* *<ip>*:*<port>* **check**
Add a checked server

balance roundrobin
Rotate across servers

default_backend *<name>*
Fallback backend

use_backend *<name>* **if** *<acl>*
Route by condition

acl *<name>* *<criterion>* *<value>*
Define a named condition

http-request redirect scheme https unless { ssl_fc }
Force HTTPS

timeout client|server|connect *<t>*
Set timeouts