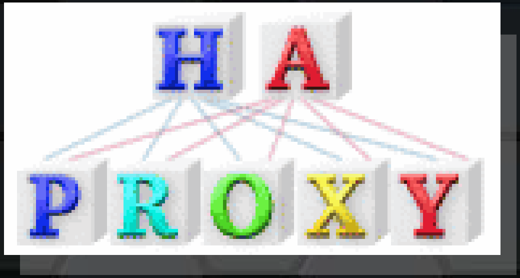
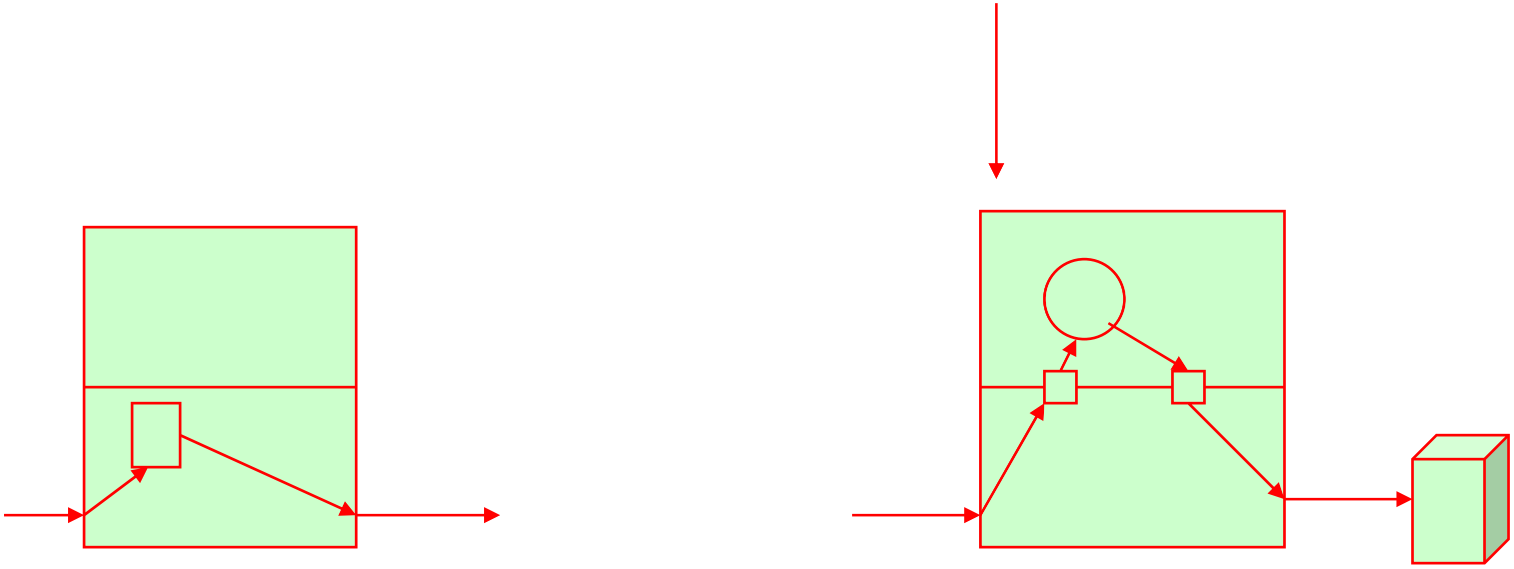
马哥教育

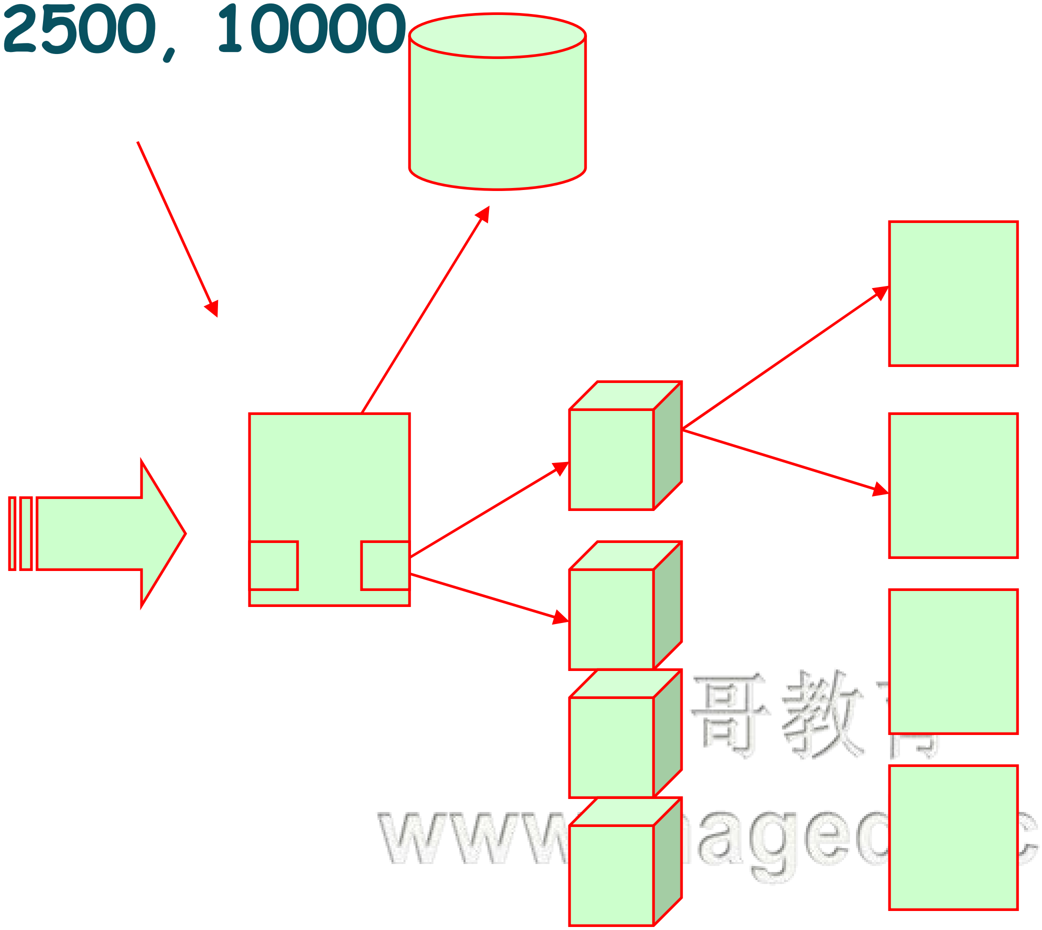


主讲：马永亮(马哥)
客服QQ：2813150558，1661815153
博客：http://mageedu.blog.51cto.com
http://www.magedu.com

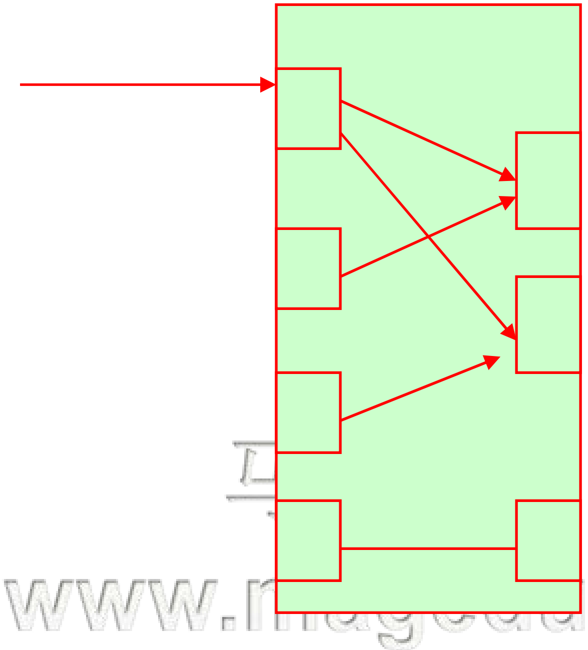
4000*8640



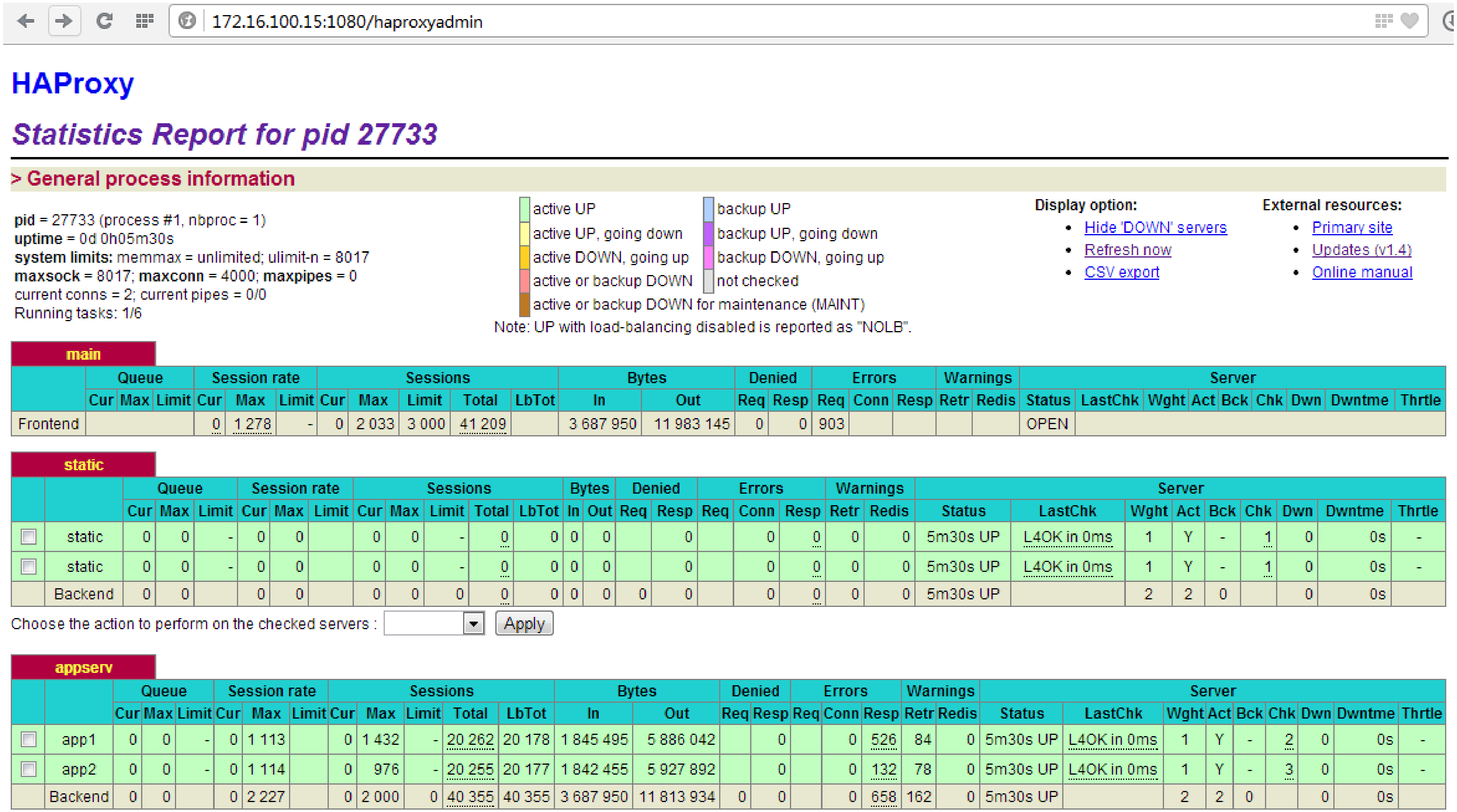
- 2500，10000



❖ listen = frontend + backend



haproxy stats page



HAProxy

Statistics Report for pid 27733

> General process information

pid = 27733 (process #1, nbproc = 1)
uptime = 0d 0h05m30s
system limits: memmax = unlimited; ulimit-n = 8017
maxsock = 8017; **maxconn** = 4000; **maxpipes** = 0
current conns = 2; current pipes = 0/0
Running tasks: 1/6

🟩	active UP	🟦	backup UP
🟨	active UP, going down	🟪	backup UP, going down
🟧	active DOWN, going up	🟪	backup DOWN, going up
🟥	active or backup DOWN	⬜	not checked
🟫	active or backup DOWN for maintenance (MAINT)		

Note: UP with load-balancing disabled is reported as "NOLB".

Display option:

- Hide 'DOWN' servers
- Refresh now
- CSV export

External resources:

- Primary site
- Updates (v1.4)
- Online manual

main

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				0	1 278	-	0	2 033	3 000	41 209		3 687 950	11 983 145	0	0	903					OPEN								

static

		Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
		Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
☐	static	0	0	-	0	0		0	0	-	0	0	0	0		0		0	0	0	0	5m30s UP	L4OK in 0ms	1	Y	-	1	0	0s	-
☐	static	0	0	-	0	0		0	0	-	0	0	0	0		0		0	0	0	0	5m30s UP	L4OK in 0ms	1	Y	-	1	0	0s	-
	Backend	0	0		0	0		0	0	0	0	0	0	0	0	0		0	0	0	0	5m30s UP		2	2	0		0	0s	

Choose the action to perform on the checked servers : [▾] Apply

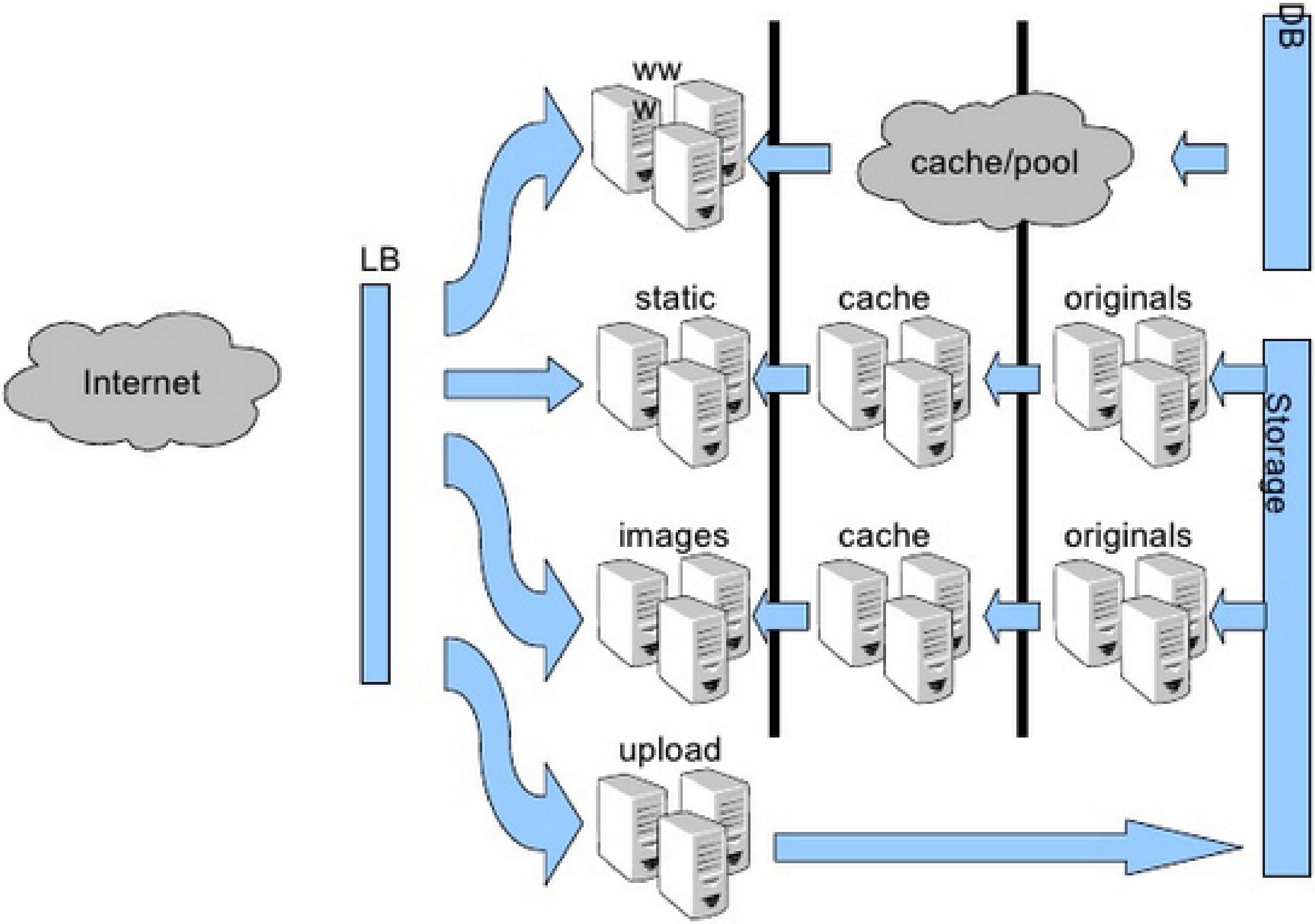
appserv

		Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
		Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
☐	app1	0	0	-	0	1 113		0	1 432	-	20 262	20 178	1 845 495	5 886 042		0		0	526	84	0	5m30s UP	L4OK in 0ms	1	Y	-	2	0	0s	-
☐	app2	0	0	-	0	1 114		0	976	-	20 255	20 177	1 842 455	5 927 892		0		0	132	78	0	5m30s UP	L4OK in 0ms	1	Y	-	3	0	0s	-
	Backend	0	0		0	2 227		0	2 000	0	40 355	40 355	3 687 950	11 813 934	0	0		0	658	162	0	5m30s UP		2	2	0		0	0s	

Agenda

- Load Balancer
- HAProxy Overview
- Installing HAProxy
- Configuring HAProxy
  - acl

Load Balancer



Load Balancer

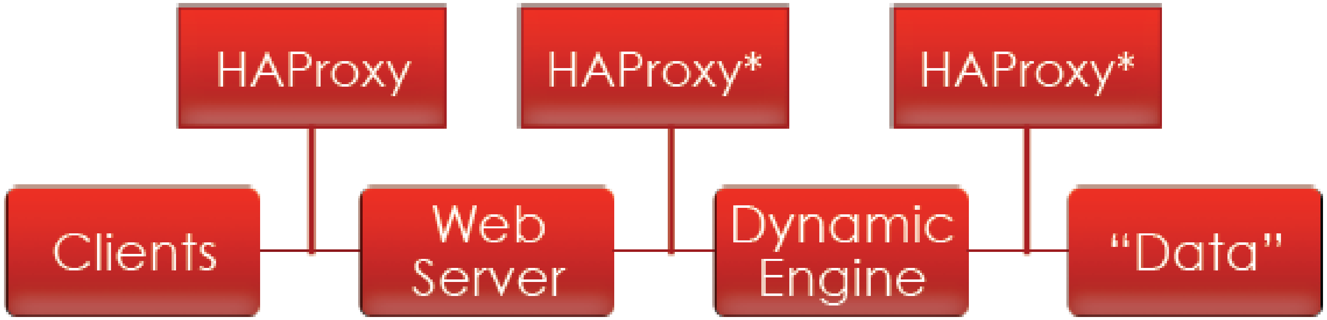
- Layer 4
  - lvs
- Layer 7
  - ats
  - Nginx
  - HAProxy
  - Apache
  - Pound
  - Perlbal (LiveJournal)

HAProxy

- High Availability Proxy
- TCP load balancing proxy with awesome health checking built in
- Fast
- Scalable
- Makes non-HA services HA

HAProxy Uses

- Not really a service unto itself
- Fits into the gaps between layers well
- Issue: Becomes a single point of failure itself



Eliminating SPOFs

- Two types of HAProxy SPOFs:
  - Service Outage
    - Hardware failure or HAProxy service failure
  - HAProxy Limit Outage / Upstream Outage
    - Hit some arbitrary limit we defined somewhere or ran out of some slots somewhere

Service Outage

- HAProxy service crashes or dies for some reason
- Hardware / Network Failure
- Solution
  - Corosync & Pacemaker
  - keepalived

HAProxy Limit Outage/Upstream Outage

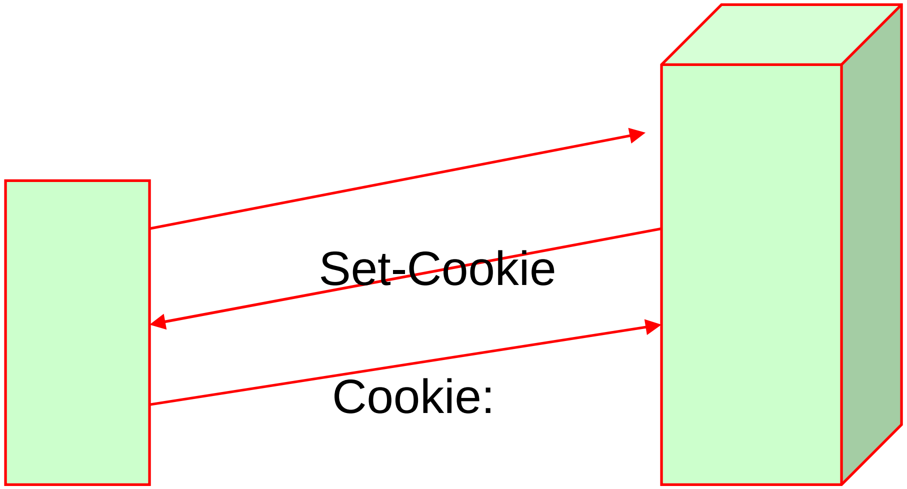
- Usually because of an outage further upstream at the Dynamic or “Data” layer
- Completely Hypothetical Situation: Mongo slows down, causing PHP processes to back up, causing the connection limit to go through the roof, causing total outage

马哥教育

HAProxy proxies

主讲：马永亮(马哥)
客服QQ：2813150558，1661815153
博客：http://mageedu.blog.51cto.com
http://www.magedu.com

- PHPSESSIONID
- JSESSION

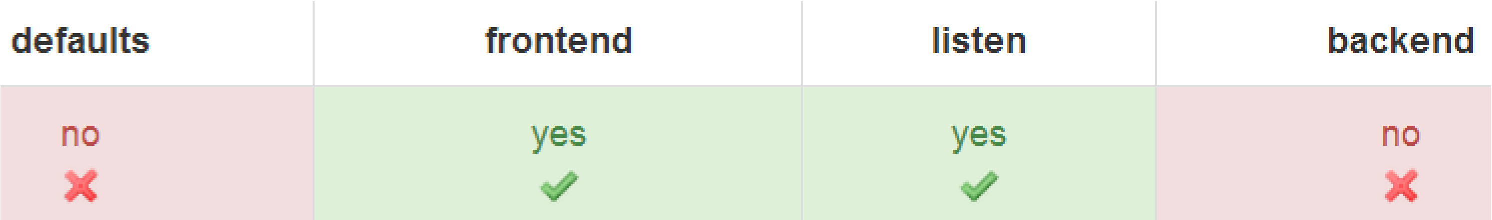


Proxies

- Proxy configuration can be located in a set of sections
  - defaults <name>
    - Sets default parameters for all other sections following its declaration
  - frontend <name>
    - Describes a set of listening sockets accepting client connections
  - backend <name>
    - Describes a set of servers to which the proxy will connect to forward incoming connections
  - listen <name>
    - Defines a complete proxy with its frontend and backend parts combined in one section
    - It is generally useful for TCP-only traffic

keyword: bind

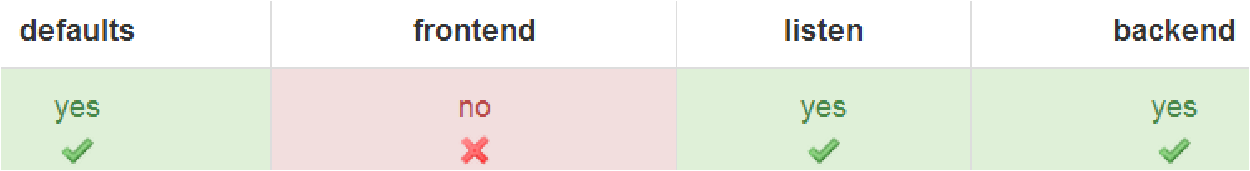
- bind [<address>]:<port_range> [, ...]
  - Define one or several listening addresses and/or ports in a frontend
  - e.g.
    - listen http_proxy
    - bind :80,:443
    - bind 10.0.0.1:10080,10.0.0.1:10443



defaults	frontend	listen	backend
no ✗	yes ✓	yes ✓	no ✗

keyword: balance

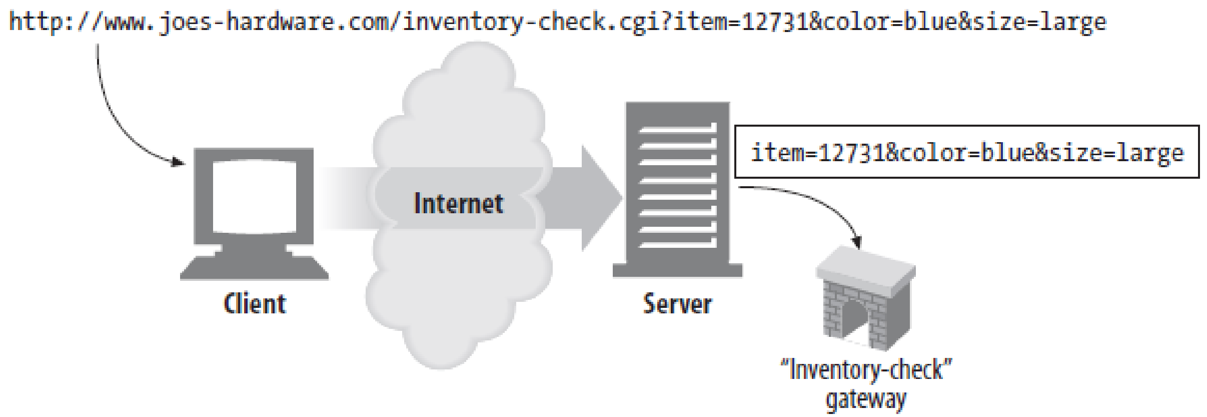
- balance <algorithm> [<arguments>]
- balance url_param <param> [check_post [<max_wait>]]
  - Define the load balancing algorithm to be used in a backend
    - roundrobin
    - static-rr
    - leastconn
    - source
    - uri
    - url_param
    - hdr(<name>)
    - rdp-cookie
    - rdp-cookie(name)



defaults	frontend	listen	backend
yes ✓	no ✗	yes ✓	yes ✓

keyword: balance(2)

- URL Syntax:
  - <scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>
    - ;<params>
      - ftp://downloads.magedu.com/pub/gnu;type=d
      - http://www.magedu.com/hammers;sale=false/index.html;graphics=true
    - ?<query>
      - Some resources, such as database services, can be asked questions or queries to narrow down the type of resource being requested
      - http://www.joes-hardware.com/inventory-check.cgi?item=12731


http://www.joes-hardware.com/inventory-check.cgi?item=12731&color=blue&size=large
Client
Internet
Server
item=12731&color=blue&size=large
"Inventory-check" gateway

keyword: balance(3)

- <algorithm>
  - roundrobin
    - Each server is used in turns, according to their weights
    - This algorithm is dynamic, which means that server weights may be adjusted on the fly for slow starts for instance
    - It is limited by design to 4095 active servers per backend
  - static-rr
    - Each server is used in turns, according to their weights
    - This algorithm is static, which means that changing a server's weight on the fly will have no effect
    - it has no design limitation on the number of servers, and when a server goes up, it is always immediately reintroduced into the farm, once the full map is recomputed

keyword: balance(4)

- <algorithm>
  - leastconn
    - The server with the lowest number of connections receives the connection
    - Use of this algorithm is recommended where very long sessions are expected such as LDAP, SQL, TSE, etc... but is not very well suited for protocols using short sessions such as HTTP
    - This algorithm is dynamic
  - source
    - The source IP address is hashed and divided by the total weight of the running servers to designate which server will receive the request
    - This algorithm is generally used in TCP mode where no cookie may be inserted
    - This algorithm is static by default, but this can be changed using "hash-type"

keyword: balance(5)

- <algorithm>
  - uri
    - This algorithm hashes either the left part of the URI (before the question mark) or the whole URI (if the "whole" parameter is present) and divides the hash value by the total weight of the running servers
    - This ensures that the same URI will always be directed to the same server as long as no server goes up or down
    - This is used with proxy caches and anti-virus proxies in order to maximize the cache hit rate
    - Note that this algorithm may only be used in an HTTP backend
    - This algorithm is static by default, but this can be changed using "hash-type"
    - This algorithm supports two optional parameters "len" and "depth", both followed by a positive integer number

keyword: balance(6)

- <algorithm>
  - url_param
    - The URL parameter specified in argument will be looked up in the query string of each HTTP GET request
    - If the parameter is found followed by an equal sign ('=') and a value, then the value is hashed and divided by the total weight of the running servers
    - This is used to track user identifiers in requests and ensure that a same user ID will always be sent to the same server as long as no server goes up or down
    - If no value is found or if the parameter is not found, then a round robin algorithm is applied
    - Note that this algorithm may only be used in an HTTP backend
    - This algorithm is static by default, but this can be changed using "hash-type“

keyword: balance(7)

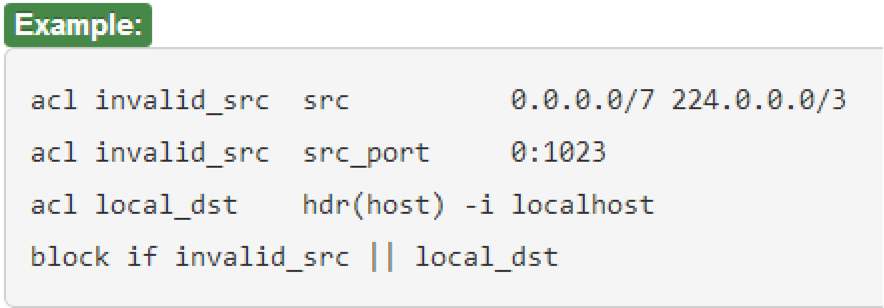
- <algorithm>
  - hdr(<name>)
    - The HTTP header <name> will be looked up in each HTTP request
      - the header name in parenthesis is not case sensitive
    - If the header is absent or if it does not contain any value, the roundrobin algorithm is applied instead
    - An optional 'use_domain_only' parameter is available, for reducing the hash algorithm to the main domain part with some specific headers such as 'Host'
    - This algorithm is static by default, but this can be changed using "hash-type“

keyword: balance(8)

- <algorithm>
  - rdp-cookie
  - rdp-cookie(name)
    - The RDP cookie <name> (or "mstshash" if omitted) will be looked up and hashed for each incoming TCP request
      - Remote Desktop Protocol (RDP) is a multichannel-capable protocol that allows for separate virtual channels for carrying presentation data, serial device communication, licensing information, highly encrypted data (keyboard and mouse activity), and so on
    - This algorithm is static by default, but this can be changed using "hash-type“

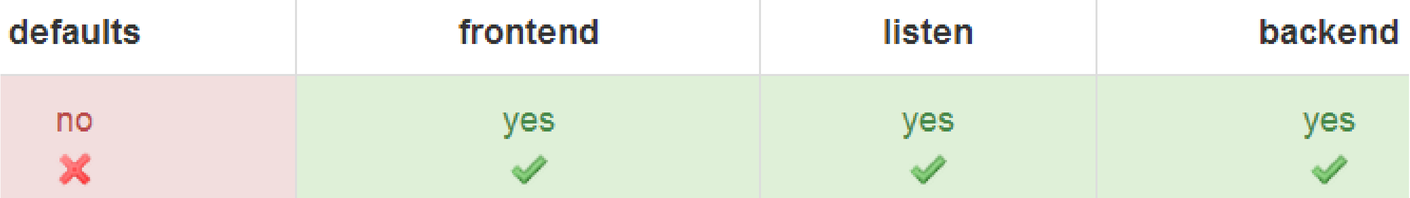
keyword: block

- block { if | unless } <condition>
  - Block a layer 7 request if/unless a condition is matched
  - The HTTP request will be blocked very early in the layer 7 processing if/unless <condition> is matched
  - A 403 error will be returned if the request is blocked
  - The condition has to reference ACLs



Example:

```
acl invalid_src  src          0.0.0.0/7 224.0.0.0/3
acl invalid_src  src_port     0:1023
acl local_dst    hdr(host) -i localhost
block if invalid_src || local_dst
```



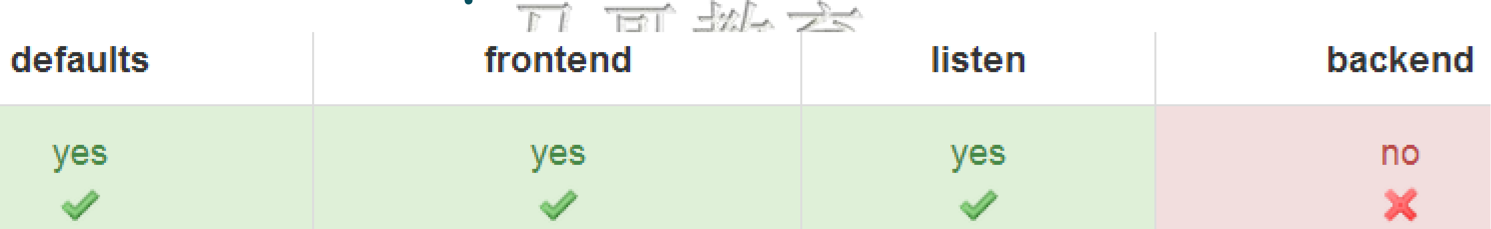
defaults	frontend	listen	backend
no ✗	yes ✓	yes ✓	yes ✓

keyword: cookie

- cookie <name> [rewrite | insert | prefix] [indirect] [nocache] [postonly] [preserve] [httponly] [secure] [domain <domain>]* [maxidle <idle>] [maxlife <life>]
  - Enable cookie-based persistence in a backend
  - There can be only one persistence cookie per HTTP backend, and it can be declared in a defaults section
  - The value of the cookie will be the value indicated after the "cookie" keyword in a "server" statement
  - If no cookie is declared for a given server, the cookie is not set

keyword: default_backend

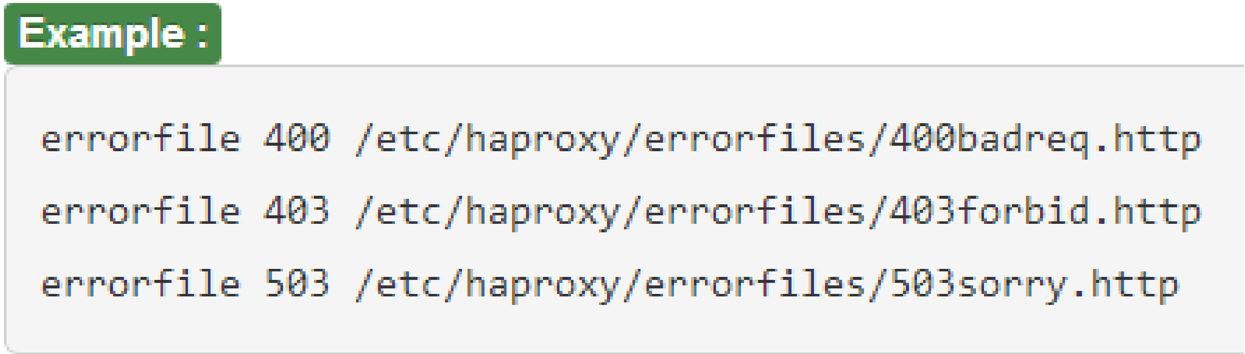
- default_backend <backend>
  - Specify the backend to use when no "use_backend" rule has been matched
  - When doing content-switching between frontend and backends using the "use_backend" keyword, it is often useful to indicate which backend will be used when no rule has matched
  - It generally is the dynamic backend which will catch all undetermined requests



defaults	frontend	listen	backend
yes ✓	yes ✓	yes ✓	no ✗

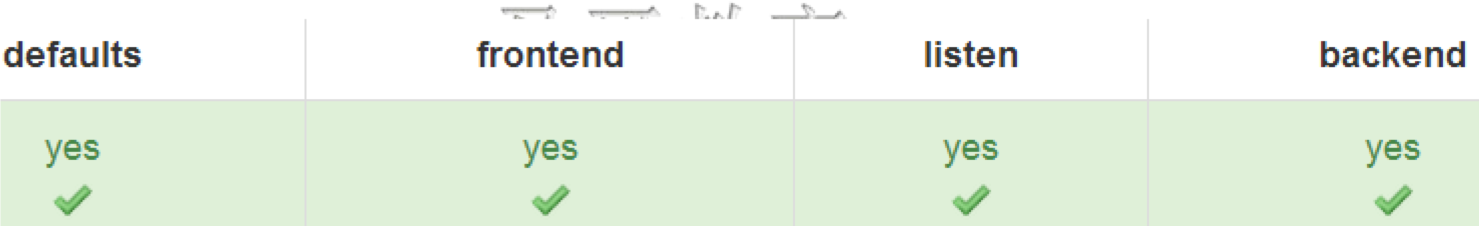
keyword: errorfile

- errorfile <code> <file>
  - Return a file contents instead of errors generated by HAProxy



Example :

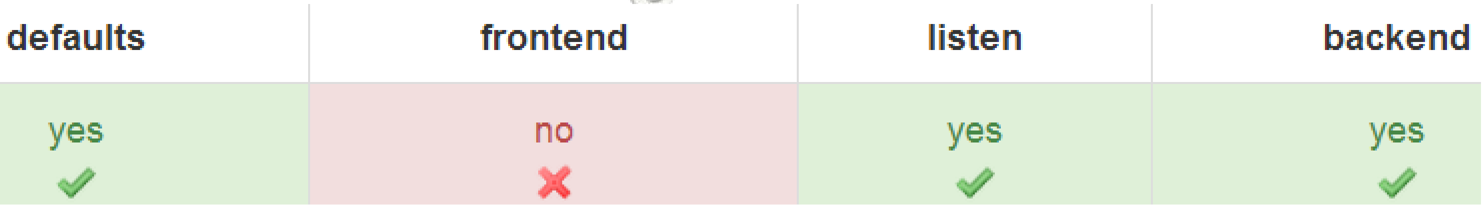
```
errorfile 400 /etc/haproxy/errorfiles/400badreq.http
errorfile 403 /etc/haproxy/errorfiles/403forbid.http
errorfile 503 /etc/haproxy/errorfiles/503sorry.http
```



defaults	frontend	listen	backend
yes ✓	yes ✓	yes ✓	yes ✓

keyword: hash-type

- hash-type <method>
  - Specify a method to use for mapping hashes to servers
    - map-based
      - The hash table is a static array containing all alive servers
      - The hashes will be very smooth, will consider weights, but will be static in that weight changes while a server is up will be ignored
    - consistent
      - The hash table is a tree filled with many occurrences of each server
      - The hash key is looked up in the tree and the closest server is chosen
      - This hash is dynamic, it supports changing weights while the servers are up, so it is compatible with the slow start feature
    - The default hash type is "map-based" and is recommended for most usages



defaults	frontend	listen	backend
yes ✓	no ✗	yes ✓	yes ✓

keyword: http-request

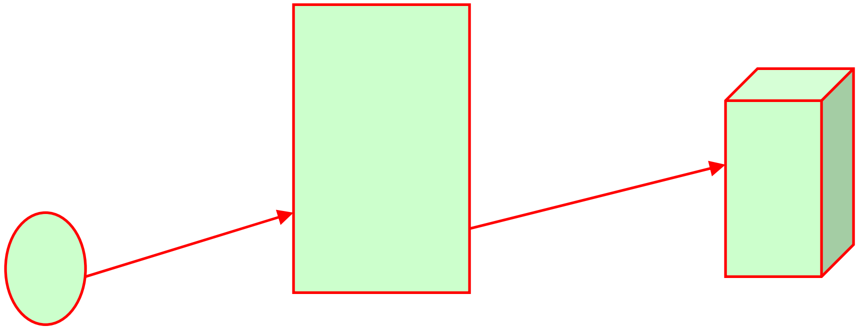
- http-request { allow | deny | auth [realm <realm>] } [{ if | unless } <condition>]
  - Access control for Layer 7 requests
  - e.g.

```
acl nagios src 192.168.129.3
acl local_net src 192.168.0.0/16
acl auth_ok http_auth(L1)
http-request allow if nagios
http-request allow if local_net auth_ok
http-request auth realm Gimme if local_net auth_ok
http-request deny
```

option: forwardfor

- option forwardfor [except <network>] [header <name>] [if-none]
  - Enable insertion of the X-Forwarded-For header to requests sent to servers
  - <network> is an optional argument used to disable this option for sources matching <network>
  - <name> an optional argument to specify a different "X-Forwarded-For" header name
  - Note:
    - only the first request will have the header appended
    - In order to fix this, ensure that any of the "httpclose", "forceclose" or "http-server-close" options is set when using this option

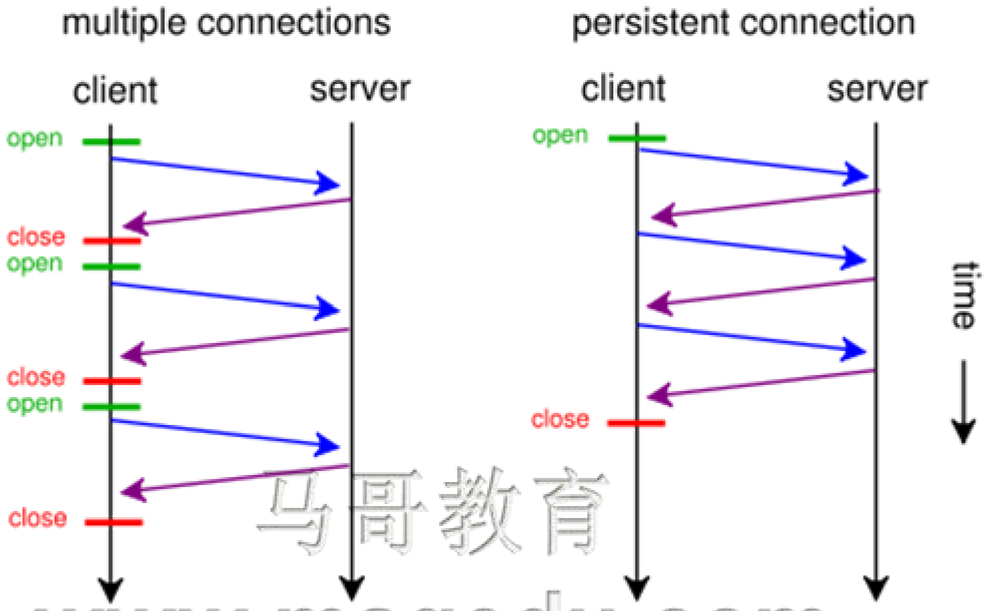
X-Forward-FOR: Client-IP



option: http-server-close

- option http-server-close
- no option http-server-close
  - Enable or disable HTTP connection closing on the server side
  - This provides the lowest latency on the client side (slow network) and the fastest session reuse on the server side to save server resources, similarly to "option forceclose“
  - It also permits non-keepalive capable servers to be served in keep-alive mode to the clients if they conform to the requirements of RFC2616
  - Some servers do not always conform to those requirements when they see "Connection: close" in the request
    - The effect will be that keep-alive will never be used
    - A workaround consists in enabling "option http-pretend-keepalive"

❖ Keep-Alive


multiple connections
client
server
open
close
open
close
open
close
persistent connection
client
server
open
close
time

option http-pretend-keepalive

- option http-pretend-keepalive
- no option http-pretend-keepalive
  - Define whether haproxy will announce keepalive to the server or not
  - When running with "option http-server-close" or "option forceclose", haproxy adds a "Connection: close" header to the request forwarded to the server
  - Unfortunately, when some servers see this header, they automatically refrain from using the chunked encoding for responses of unknown length, while this is totally unrelated
  - By setting "option http-pretend-keepalive", haproxy will make the server believe it will keep the connection alive
  - It is recommended not to enable this option by default, because most servers will more efficiently close the connection themselves after the last packet, and release its buffers slightly earlier

option: httpclose

- option httpclose
- no option httpclose
  - Enable or disable passive HTTP connection closing
    - By default, when a client communicates with a server, HAProxy will only analyze, log, and process the first request of each connection
    - If "option httpclose" is set, it will check if a "Connection: close" header is already set in each direction, and will add one if missing
    - Each end should react to this by actively closing the TCP connection after each transfer, thus resulting in a switch to the HTTP close mode

option: redispatch

- option redispatch
- no option redispatch
  - Enable or disable session redistribution in case of connection failure
    - In HTTP mode, if a server designated by a cookie is down, clients may definitely stick to it because they cannot flush the cookie, so they will not be able to access the service anymore
    - Specifying "option redispatch" will allow the proxy to break their persistence and redistribute them to a working server

keyword: redirect

- redirect location <to> [code <code>] <option> [{if | unless} <condition>]
- redirect prefix <to> [code <code>] <option> [{if | unless} <condition>]
  - Return an HTTP redirection if/unless a condition is matched

Redirect Example

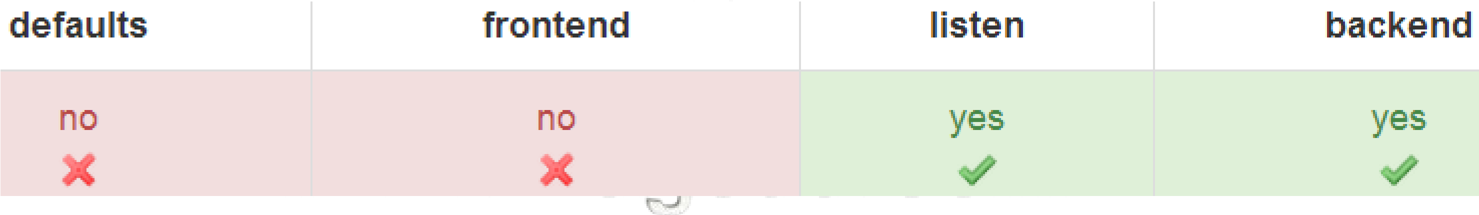
- acl clear dst_port 80
- acl secure dst_port 8080
- acl login_page url_beg /login
- acl logout url_beg /logout
- acl uid_given url_reg /login?userid=[^&]+
- acl cookie_set hdr_sub(cookie) SEEN=1
- redirect prefix https://magedu.com set-cookie SEEN=1 if !cookie_set
- redirect prefix https://magedu.com if login_page !secure
- redirect prefix http://magedu.com drop-query if login_page !uid_given
- redirect location http://www.magedu.com/ if !login_page secure
- redirect location / clear-cookie USERID= if logout

keyword: reqadd

- reqadd <string> [{if | unless} <cond>]
  - Add a header at the end of the HTTP request
  - <string> is the complete line to be added
    - Any space or known delimiter must be escaped using a backslash ('\')
  - <cond> is an optional matching condition built from ACLs
  - e.g.
    - acl is-ssl dst_port 81
    - reqadd X-Proto:\ SSL if is-ssl
- rspadd <string> [{if | unless} <cond>]
  - Add a header at the end of the HTTP response

keyword: server

- server <name> <address>[:port] [param*]
  - Declare a server in a backend
  - <name> is the internal name assigned to this server, and will appear in logs and alerts
  - <address> is the IPv4 address of the server
  - <ports> is an optional port specification
    - If set, all connections will be sent to this port
    - If unset, the same port the client connected to will be used



defaults	frontend	listen	backend
no ✗	no ✗	yes ✓	yes ✓

Server and default-server options

- server <name> <address>[:port] [settings ...]
- default-server [settings ...]

  - backup
  - check
  - cookie <value>
  - disabled
  - fall <count>
  - inter <delay>
  - fastinter <delay>
  - downinter <delay>
  - maxconn <maxconn>
  - maxqueue <maxqueue>
  - minconn <minconn>
  - observe <mode>
  - redir <prefix>
  - rise <count>
  - slowstart <start_time_in_ms>
  - weight <weight>

Server and default-server options(2)

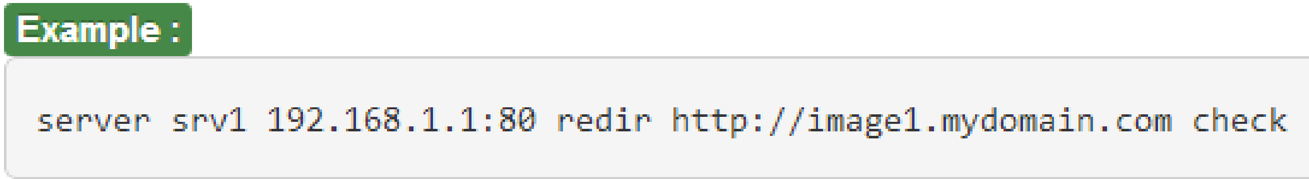
- backup
  - The server is only used in load balancing when all other non-backup servers are unavailable
- check
  - Enables health checks on the server
    - By default, a server is always considered available
  - The interval and timers using the "inter", "rise" and "fall" parameters
  - The request method is define in the backend using the "httpchk", "smtpchk", "mysql-check" and "ssl-hello-chk" options
- disabled
  - starts the server in the "disabled" state
  - That means that it is marked down in maintenance mode

- fall <count>
  - The server will be considered as dead after <count> consecutive unsuccessful health checks
  - This value defaults to 3 if unspecified
- inter <delay>
- fastinter <delay>
- downinter <delay>
  - The "inter" parameter sets the interval between two consecutive health checks to <delay> milliseconds
  - The delay defaults to 2000 ms
  - It is also possible to use "fastinter" and "downinter" to optimize delays between checks depending on the server state

- maxconn <maxconn>
  - Specifies the maximal number of concurrent connections that will be sent to this server
  - If the number of incoming concurrent requests goes higher than this value, they will be queued, waiting for a connection to be released
  - If a "minconn“ parameter is specified, the limit becomes dynamic
  - The default value is "0" which means unlimited
- maxqueue <maxqueue>
  - The "maxqueue" parameter specifies the maximal number of connections which will wait in the queue for this server
  - If this limit is reached, next requests will be redispatched to other servers instead of indefinitely waiting to be served
  - The default value is "0" which means the queue is unlimited

- minconn <minconn>
  - When the "minconn" parameter is set, the maxconn limit becomes a dynamic limit following the backend's load
  - The server will always accept at least <minconn> connections, never more than <maxconn>, and the limit will be on the ramp between both values when the backend has less than <fullconn> concurrent connections
- observe <mode>
  - Enables health adjusting based on observing communication with the server
  - There are two supported modes: "layer4" and "layer7"

- redir <prefix>
  - Enables the redirection mode for all GET and HEAD requests addressing this server
    - This means that instead of having HAProxy forward the request to the server, it will send an "HTTP 302" response with the "Location" header composed of this prefix immediately followed by the requested URI beginning at the leading '/' of the path component
    - All invalid requests will be rejected, and all non-GET or HEAD requests will be normally served by the server
    - Note: never use a relative location here, it would cause a loop between the client and HAProxy



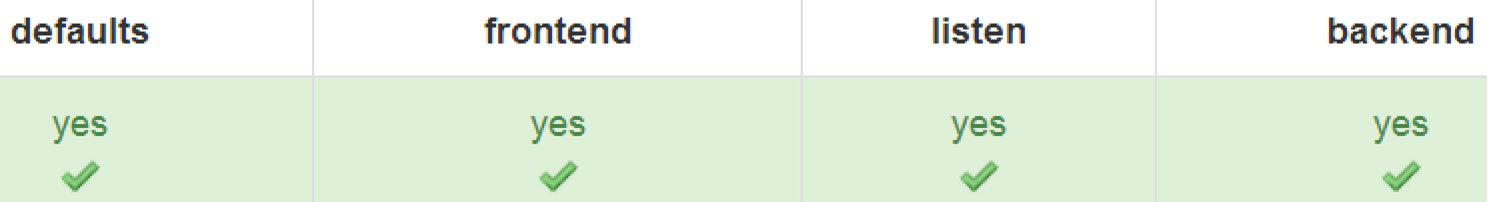
Example :

```
server srv1 192.168.1.1:80 redir http://image1.mydomain.com check
```

- rise <count>
  - States that a server will be considered as operational after <count> consecutive successful health checks
- slowstart <start_time_in_ms>
  - Indicates after how long a server which has just come back up will run at full speed
  - The slowstart never applies when haproxy starts, otherwise it would cause trouble to running servers
    - It only applies when a server has been previously seen as failed
- weight <weight>
  - The default weight is 1, and the maximal value is 256
  - A value of 0 means the server will not participate in load-balancing but will still accept persistent connections

keyword: timeout http-request

- timeout http-request <timeout>
  - Set the maximum allowed time to wait for a complete HTTP request
    - Milliseconds by default, but can be in any other unit if the number is suffixed by the unit
  - This helps protecting against established connections on which nothing is sent



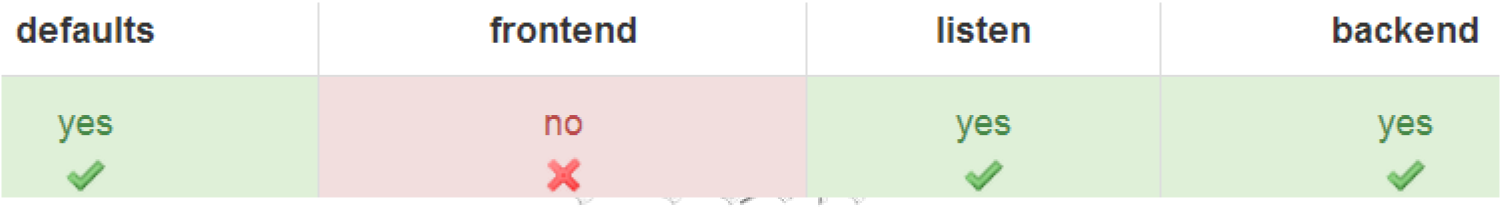
defaults	frontend	listen	backend
yes ✓	yes ✓	yes ✓	yes ✓

option: timeout queue

- timeout queue <timeout>
  - Set the maximum time to wait in the queue for a connection slot to be free
  - When a server's maxconn is reached, connections are left pending in a queue which may be server-specific or global to the backend
    - In order not to wait indefinitely, a timeout is applied to requests pending in the queue
    - If the timeout is reached, it is considered that the request will almost never be served, so it is dropped and a 503 error is returned to the client

keyword: timeout connect

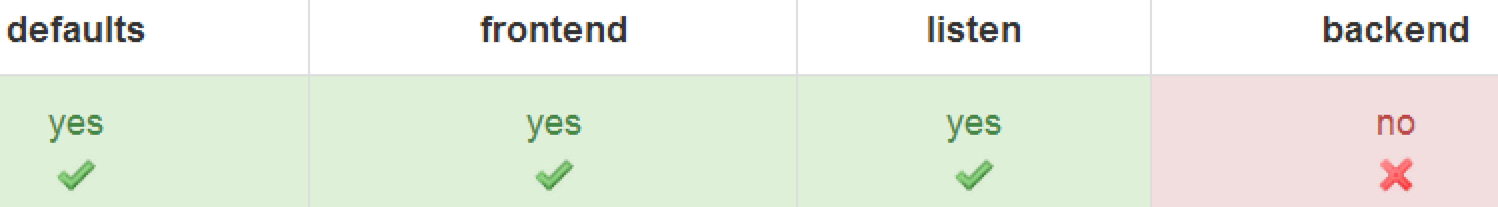
- timeout connect <timeout>
  - Set the maximum time to wait for a connection attempt to a server to succeed
  - If the server is located on the same LAN as haproxy, the connection should be immediate (less than a few milliseconds)



defaults	frontend	listen	backend
yes ✓	no ✗	yes ✓	yes ✓

option: timeout client

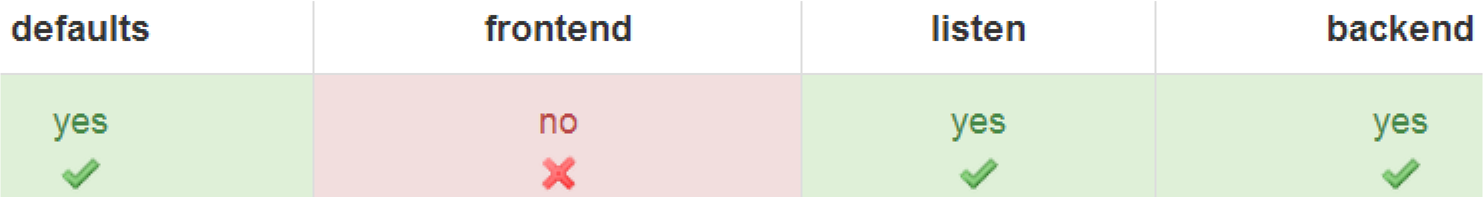
- timeout client <timeout>
  - Set the maximum inactivity time on the client side
  - The inactivity timeout applies when the client is expected to acknowledge or send data



defaults	frontend	listen	backend
yes ✓	yes ✓	yes ✓	no ✗

option: timeout server

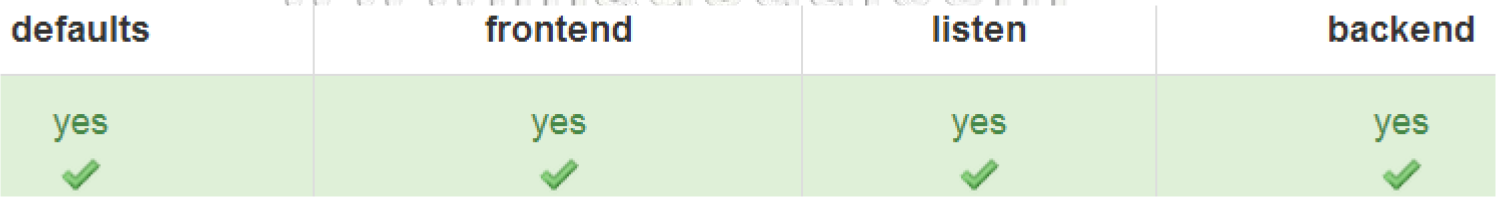
- timeout server <timeout>
  - Set the maximum inactivity time on the server side
  - The inactivity timeout applies when the server is expected to acknowledge or send data



defaults	frontend	listen	backend
yes ✓	no ✗	yes ✓	yes ✓

keyword: timeout http-keep-alive

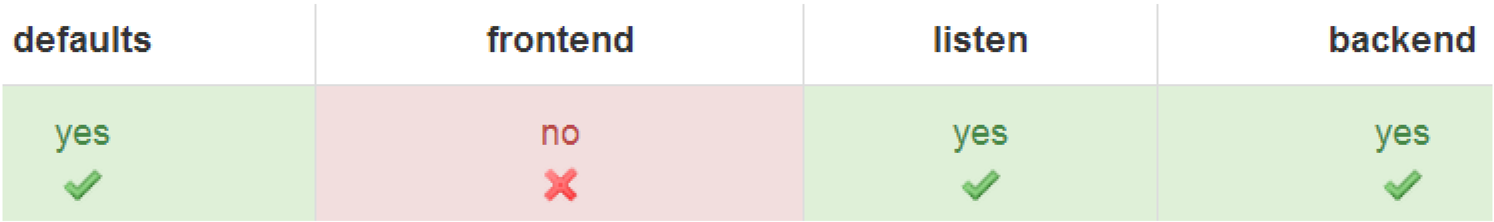
- timeout http-keep-alive <timeout>
  - Set the maximum allowed time to wait for a new HTTP request to appear
    - By default, the time to wait for a new request in case of keep-alive is set by "timeout http-request“
    - However this is not always convenient
  - It will define how long to wait for a new HTTP request to start coming after a response was sent
  - If this parameter is not set, the "http-request" timeout applies, and if both are not set, "timeout client" still applies at the lower level



defaults	frontend	listen	backend
yes ✓	yes ✓	yes ✓	yes ✓

keyword: timeout check

- timeout check <timeout>
  - Set additional check timeout, but only after a connection has been already established



defaults	frontend	listen	backend
yes ✓	no ✗	yes ✓	yes ✓

马哥教育

HAProxy ACL

主讲：马永亮(马哥)

客服QQ：2813150558，1661815153

博客：http://mageedu.blog.51cto.com

http://www.magedu.com

ACL

- ACL provides a flexible solution to perform content switching and generally to take decisions based on content extracted from the request, the response or any environmental status
- The principle is simple
  - define test criteria with sets of values
  - perform actions only if a set of tests is valid
  - The actions generally consist in blocking the request, or selecting a backend
  - e.g.

```
acl static_graph url_reg .*\.(jpg|gif|png|js|css|ico|swf)$
use_backend varnish if static_graph
```

ACL Syntax

- acl <aclname> <criterion> [flags] [operator] <value> ...
  - This creates a new ACL **<aclname>** or completes an existing one with new tests
    - ACL names must be formed from upper and lower case letters, digits, '-' (dash), '_' (underscore) , '.' (dot) and ':' (colon)
    - ACL names are case-sensitive
  - Those tests apply to the portion of request/response specified in **<criterion>** and may be adjusted with optional flags **[flags]**
    - Some criteria also support an operator which may be specified before the set of values
    - The values are of the type supported by the criterion, and are separated by spaces

ACL flags

- -i
  - ignore case during matching of all subsequent patterns
- -f
  - load patterns from a file
- - -
  - force end of flags
  - Useful when a string looks like one of the flags

Supported types of values

- integers or integer ranges
  - ranges, e.g. 1024:65535
  - operators: eq, ge, gt, le, lt
- strings
  - -i
- regular expressions
- IP addresses and networks

Matching at Layer 4 and below

- dst <ip_address>
- dst_port <integer>
- src <ip_address>
- src_port <integer>

```
acl goodguys src 10.0.0.0/24
tcp-request content accept if goodguys
tcp-request content reject
```

Matching at Layer 7

- hdr(header) <string>
  - Returns true if any of the headers matching the criteria match any of the strings
  - e.g. hdr(Connection) -i close
- hdr_reg(header) <regex>
  - Returns true when one of the headers matches of the regular expressions
- http_first_req
  - Returns true when the request being processed is the first one of the connection
- method <string>
  - Applies to the method in the HTTP request, eg: "GET"

- path <string>
  - Returns true when the path part of the request, which starts at the first slash and ends before the question mark, equals one of the strings
  - It may be used to match known files, such as /favicon.ico
- path_beg <string>
  - Returns true when the path begins with one of the strings
  - This can be used to send certain directory names to alternative backends
- path_end <string>
  - Returns true when the path ends with one of the strings
  - This may be used to control file name extension

- path_reg <regex>
  - Returns true when the path matches one of the regular expressions
- url <string>
  - Applies to the whole URL passed in the request
- url_beg <string>
  - Returns true when the URL begins with one of the strings
  - This can be used to check whether a URL begins with a slash or with a protocol scheme

Using ACLs to form conditions

- Some actions are only performed upon a valid condition
- A condition is a combination of ACLs with operators
- 3 operators are supported
  - AND (implicit)
  - OR (explicit with the "or" keyword or the "||" operator)
  - Negation with the exclamation mark ("!")

An Example

- acl url_static path_beg /static /images /img /css
- acl url_static path_end .gif .png .jpg .css .js
- acl host_www hdr_beg(host) -i www
- acl host_static hdr_beg(host) -i img. video. download. ftp.
- use_backend static if host_static or url_static
- use_backend www if host_www

关于马哥教育

- 博客：http://mageedu.blog.51cto.com
- 主页：http://www.magedu.com
- 客服QQ：2813150558，1661815153
- QQ群：203585050，279599283

马哥教育

Thank You!