## Installing HAProxy CentOS 7

Download the source code with the command below. You can check if there is a newer version available at the HAProxy [download page](http://www.haproxy.org/#down).

Wget http://www.haproxy.org/download/1.8/src/haproxy-1.8.7.tar.gz -O ~/haproxy.tar.gz

Once the download is complete, extract the files using the command below.

tar xzvf ~/haproxy.tar.gz -C ~/

Change into the extracted source directory.

cd ~/haproxy-1.8.7

Then compile the program for your system.

make TARGET=linux2628

And finally, install HAProxy itself.

sudo make install

With that done, HAProxy is now installed but requires some additional steps to get it operational. Continue below with setting up the software and services.

## Setting up HAProxy for your server

Next, add the following directories and the statistics file for HAProxy records.

sudo mkdir -p /etc/haproxy

sudo mkdir -p /var/lib/haproxy

sudo touch /var/lib/haproxy/stats

Create a symbolic link for the binary to allow you to run HAProxy commands as a normal user.

sudo ln -s /usr/local/sbin/haproxy /usr/sbin/haproxy

If you want to add the proxy as a service to the system, copy the haproxy.init file from the examples to your /etc/init.d directory. Change the file permissions to make the script executable and then reload the systemd daemon.

sudo cp ~/haproxy-1.7.8/examples/haproxy.init /etc/init.d/haproxy

sudo chmod 755 /etc/init.d/haproxy

sudo systemctl daemon-reload

You will also need to enable the service to allow it to restart automatically at system boot up.

sudo chkconfig haproxy on

For general usage, it is also recommended to add a new user for HAProxy to be run under.

sudo useradd -r haproxy

Afterwards, you can double check the installed version number with the following command.

haproxy -v

HA-Proxy version 1.7.8 2017/07/07

Copyright 2000-2017 Willy Tarreau <willy@haproxy.org>

In this case, the version should be 1.7.8 like shown in the example output above.

Lastly, the firewall on CentOS 7 is quite restrictive for this project by default. Use the following commands to allow the required services and reload the firewall.

sudo firewall-cmd --permanent --zone=public --add-service=http

sudo firewall-cmd --permanent --zone=public --add-port=8181/tcp

sudo firewall-cmd --reload

Configuring the load balancer

Setting up HAProxy for load balancing is a quite straight forward process. Basically, all you need to do is tell HAProxy what kind of connections it should be listening for and where the connections should be relayed to.

This is done by creating a configuration file /etc/haproxy/haproxy.cfg with the defining settings. You can read about the configuration options at HAProxy [documentation page](https://cbonte.github.io/haproxy-dconv/1.7/configuration.html) if you wish to find out more.

Load balancing on layer 4

Once installed HAProxy should already have a template for configuring the load balancer. Open the configuration file, for example, using nano with the command underneath.

sudo nano /etc/haproxy/haproxy.cfg

Add the following sections to the end of the file. Replace the <server name> with what ever you want to call you servers on the statistics page and the <private IP> with the private IPs for the servers you wish to direct the web traffic to. You can check the private IPs at your [UpCloud Control Panel](https://my.upcloud.com/network" \t "_blank) and *Private network* tab under *Network* menu.

global

log /dev/log local0

log /dev/log local1 notice

chroot /var/lib/haproxy

stats timeout 30s

user haproxy

group haproxy

daemon

defaults

log global

mode http

option httplog

option dontlognull

timeout connect 5000

timeout client 50000

timeout server 50000

frontend http\_front

bind \*:80

stats uri /haproxy?stats

default\_backend http\_back

backend http\_back

balance roundrobin

server <server name> <private IP>:80 check

server <server name> <private IP>:80 check

This defines a layer 4 load balancer with a front-end name http\_front listening to the port number 80, which then directs the traffic to the default backend named http\_back. The additional stats URI /haproxy?stats enables the statistics page at that specified address.

Different load balancing algorithms

Configuring the servers in the backend section allows HAProxy to use these servers for load balancing according to the roundrobin algorithm whenever available.

The balancing algorithms are used to decide which server at the backend each connection is transferred to. Some of the useful options include the following:

* Roundrobin: Each server is used in turns according to their weights. This is the smoothest and fairest algorithm when the servers’ processing time remains equally distributed. This algorithm is dynamic, which allows server weights to be adjusted on the fly.
* Leastconn: The server with the lowest number of connections is chosen. Round-robin is performed between servers with the same load. Using this algorithm is recommended with long sessions, such as LDAP, SQL, TSE, etc, but it is not very well suited for short sessions such as HTTP.
* First: The first server with available connection slots receives the connection. The servers are chosen from the lowest numeric identifier to the highest, which defaults to the server’s position on the farm. Once a server reaches its maxconn value, the next server is used.
* 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 way the same client IP address will always reach the same server while the servers stay the same.

Configuring load balancing for layer 7

Another possibility is to configure the load balancer to work on layer 7, which is useful when parts of your web application are located on different hosts. This can be accomplished by conditioning the connection transfer for example by the URL.

Open the HAProxy configuration file with a text editor.

sudo nano /etc/haproxy/haproxy.cfg

Then set the front and backend segments according to the example below.

frontend http\_front

bind \*:80

stats uri /haproxy?stats

acl url\_blog path\_beg /blog

use\_backend blog\_back if url\_blog

default\_backend http\_back

backend http\_back

balance roundrobin

server <server name> <private IP>:80 check

server <server name> <private IP>:80 check

backend blog\_back

server <server name> <private IP>:80 check

The front end declares an ACL rule named url\_blog that applies to all connections with paths that begin with /blog. Use\_backend defines that connections matching the url\_blogcondition should be served by the backend named blog\_back, while all other requests are handled by the default backend.

At the backend side, the configuration sets up two server groups, http\_back like before and the new one called blog\_back that servers specifically connections to example.com/blog.

After making the configurations, save the file and restart HAProxy with the next command.

sudo systemctl restart haproxy

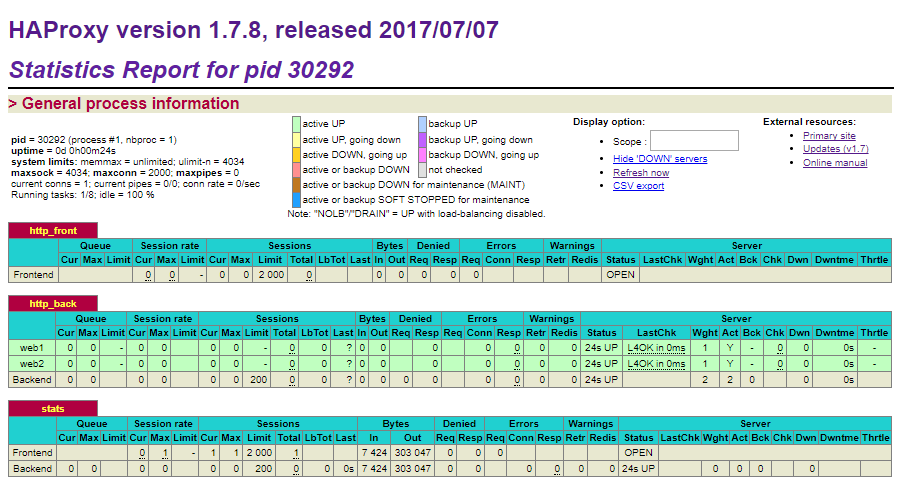
If you get any errors or warnings at start up, check the configuration for any mistypes and then try restarting again.

Testing the setup

With the HAProxy configured and running, open your load balancer server’s public IP in a web browser and check that you get connected to your backend correctly. The parameter stats uri in the configuration enables the statistics page at the defined address.

http://<load balancer public IP>/haproxy?stats

When you load the statistics page and all of your servers are listed in green your configuration was successful!



The statistics page contains some helpful information to keep track of your web hosts including up and down times and session counts. If a server is listed in red, check that the server is powered on and that you can ping it from the load balancer machine.

In case your load balancer does not reply, check that HTTP connections are not getting blocked by a firewall. Also, confirm that HAProxy is running with the command below.

sudo systemctl status haproxy

Password protecting the statistics page

Having the statistics page simply listed at the front end, however, is publicly open for anyone to view, which might not be such a good idea. Instead, you can set it up to its own port number by adding the example below to the end of your haproxy.cfg file. Replace the username and password with something secure.

listen stats

bind \*:8181

stats enable

stats uri /

stats realm Haproxy\ Statistics

stats auth username:password

After adding the new listen group, remove the old reference to the stats uri from the frontend group. When done, save the file and restart HAProxy again.

sudo systemctl restart haproxy

Then open the load balancer again with the new port number, and log in with the username and password you set in the configuration file.

http://<load balancer public IP>:8181

Check that your servers are still reporting all green and then open just the load balancer IP without any port numbers on your web browser.

http://<load balancer public IP>/

If your backend servers have at least slightly different landing pages you will notice that each time you reload the page you get the reply from a different host.

PORT FORWARDING:

Trying to change default port of 80 to some other port in haproxy config file to make the application secured.To do this we need to disabled selinux if it is in enforecement mode.go to path of selinux /etc/sysconfig/selinux after changing we need to reboot the system.If we are in the production this process is not allowed so to make this we use other approach which is shown below.

