HAProxy

Free, Fast High Availability and Load Balancing

Adam Thornton
10 September 2014

What?

- HAProxy is a proxy for Layer 4 (TCP) or Layer 7 (HTTP) traffic
- GPLv2
- http://www.haproxy.org

- Disclaimer: I don't work for them. I'm just a satisfied (non-paying) customer.

Free?

- GPLv2
- Just like the Linux kernel itself. Same restrictions: free to use, free to modify, if you distribute a modified version you are required to make your modifications freely available.
- https://www.gnu.org/licenses/gpl-2.0.html
- Beer and speech.

Fast? Scalable?

- “Haproxy on a typical Xeon E5 of 2014 can forward data up to about 40 Gbps. A fanless 1.6 GHz Atom CPU is slightly above 1 Gbps.”
  - (http://www.haproxy.org/#plat)
  - This is probably faster than you need. On one processor.
- If not: http://brokenhaze.com/blog/2014/03/25/how-stack-exchange-gets-the-most-out-of-haproxy/
  - Stack Exchange is probably busier than your site.

Why?

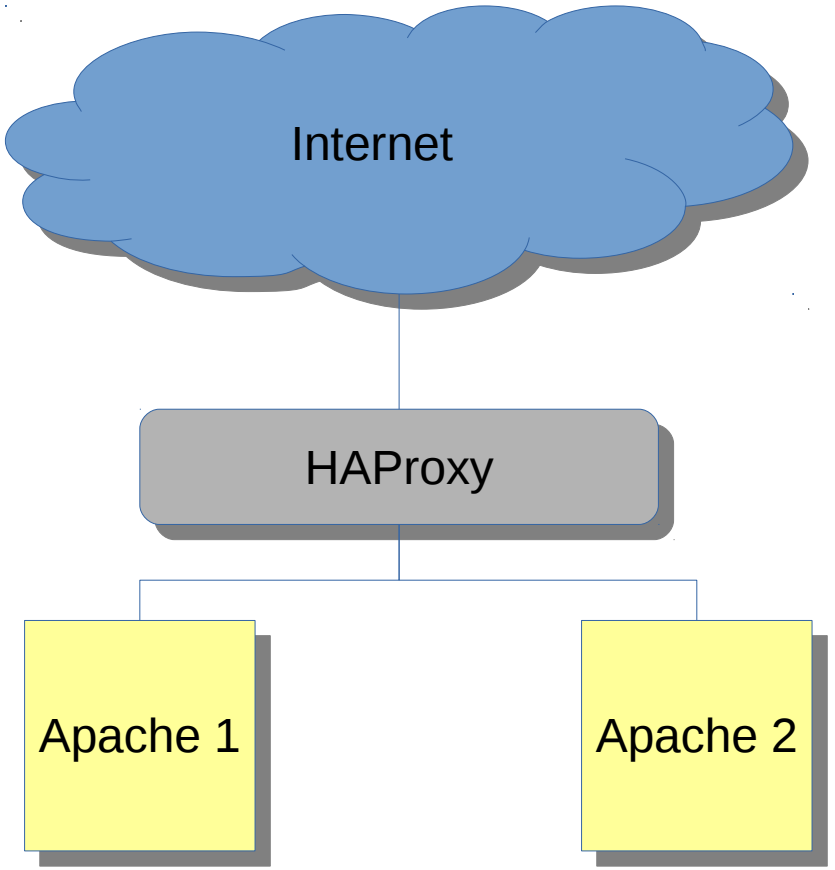
- Your “hardware load balancer” is probably something very much like F5 BigIP, right?
  - Costs a whole lot
  - Is under the control of the network group, which requires that all changes be scheduled two months in advance, or require an “expedite fee” of $5000.
  - Is really just a Linux box anyway
    - With a horrible tcl-derived user shell/control syntax

Whereas, HAProxy

- Is GPLv2.
  - Zero licensing fee, and you get the source.
  - But they do have a paid-support model if you need that to make it more legitimate for your enterprise.
- Is just a software layer
  - Doesn't require privilege if you don't need to bind to ports < 1024
    - All my examples assume you're running HAProxy and backends with no privilege. Low ports work just like you'd expect if you start the service as root (or grant capabilities with setcap).
- Configuration syntax is, if not pretty, at least well-documented.
  - https://cbonte.github.io/haproxy-dconv/configuration-1.5.html

Basic Features

Let's look at a trivial HA solution:



HAProxy Config (boilerplate)

```
global
    daemon
    maxconn 256
defaults
    mode http
    timeout connect 2000ms
    timeout client 10000ms
    timeout server 10000ms
```

HAProxy Config (the good stuff)

```
frontend http-in
    bind *:10080
    default_backend web_servers

backend web_servers
    balance roundrobin
    server apache1 10.11.11.10:1080
    server apache2 10.11.11.11:1080
```

But I need session persistence!

```
backend app_servers
    balance roundrobin
    cookie SERVERID insert indirect nocache
    server tomcat1 10.11.11.20:1080 \
     check cookie tomcat1
    server tomcat2 10.11.11.21:1080 \
     check cookie tomcat2
```

But we already use an app cookie!

```
backend app_servers
    balance roundrobin
    cookie JSESSIONID insert indirect nocache
    server tomcat1 10.11.11.20:1080 \
     check cookie tomcat1
    server tomcat2 10.11.11.21:1080 \
     check cookie tomcat2
```

- HAProxy prepends/strips cookie value on its way through. So if your server says:

  `Set-Cookie: JSESSIONID=foo`

- HAProxy turns that into tomcat1~foo for the return to the user's browser
- And then Cookie: `JSESSIONID=tomcat1~foo` gets turned into `JSESSIONID=foo` when HAProxy sends it back to the server.

I want to provide access to SMTP servers, but have session affinity

- So...that's not HTTP, so there aren't cookies *per se*...but we can make the same source IP get the same backend all the time.

```
frontend smtp-in
     mode tcp
    bind *:10025
    default_backend smtp_servers

backend smtp_servers
    balance source
    server exim1 10.11.11.10:1025
    server exim2 10.11.11.11:1025
```

I need to terminate SSL at HAProxy!

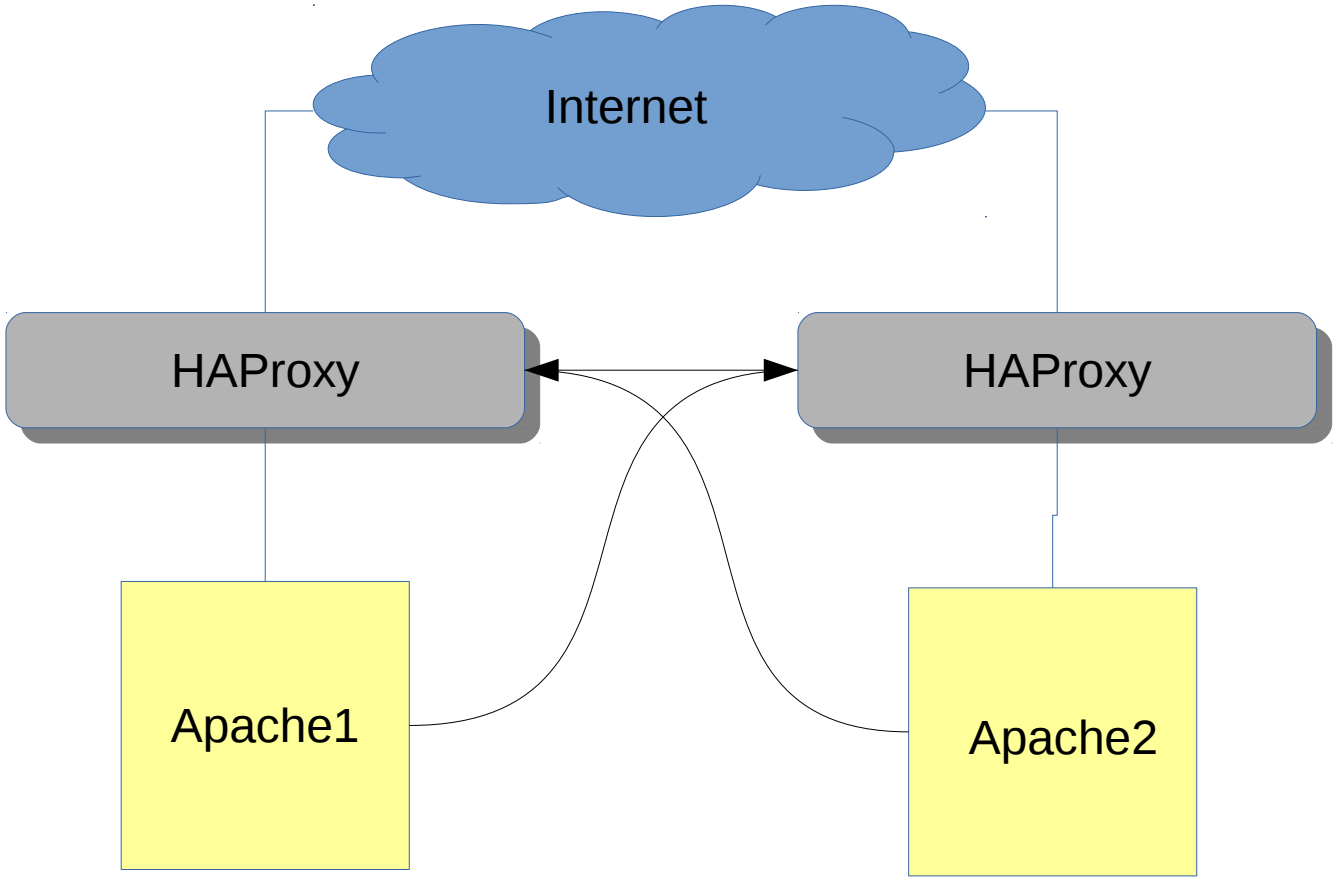
```
frontend https-in
    bind *:10443 ssl crt /etc/haproxy/site.pem
    default_backend web_servers
```

- ...and do SSL to the backends, but they all use self-signed certs!

```
backend web_servers
    balance roundrobin
    server apache1 10.11.11.10:1443 \
     ssl verify none
    server apache2 10.11.11.11:1443 \
     ssl verify none
```

OK, OK, but what if HAProxy goes down?

- So you want something like this?



Add the following to both configs

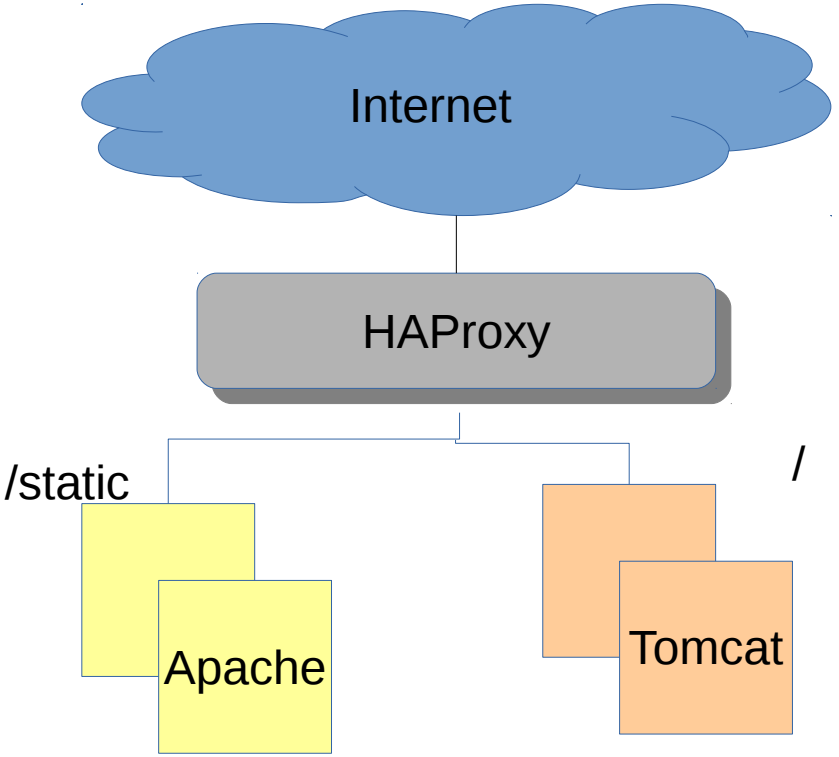
```
peers proxies
    peer haproxy1 10.11.10.5:9925
    peer haproxy2 10.11.10.6:9925
```

- And then add `stick-table` entries to the backends.

```
backend web_servers
    balance roundrobin
    stick-table type ip size 20k peers proxies
    server apache1 10.11.11.10:1080
    server apache2 10.11.11.11:1080
```

I need a multi-component app!

- Like this?



Layer 7 routing

```
frontend http-in
    bind *:10080
    default_backend app_servers
    acl static path_beg /static
    use_backend web_servers if static
```

- Your ACLs can also be regex-based
- And of course you can rewrite based on regex (e.g. to strip /static from the front before passing the request to the web server tier).
- This lets you do arbitrarily complex URI-based routing to HTTP(S)-accessible components

…and then...

```
backend web_servers
    balance roundrobin
    server apache1 10.11.11.10:1080
    server apache2 10.11.11.11:1080
backend app_servers
    balance roundrobin
    cookie SERVERID insert indirect nocache
    server tomcat1 10.11.11.20:1080 \
     check cookie tomcat1
    server tomcat2 10.11.11.21:1080 \
     check cookie tomcat2
```

Now I want statistics

OK:

```
frontend http-in
    bind *:10080
    stats enable
    stats uri /stats
    stats refresh 30s
    default_backend app_servers
    acl static path_beg /static
    use_backend web_servers if static
```

No, without the GUI

- Sure—append a “;csv” to the stats URI
- Or use the socket interface

```
stats socket /tmp/haproxy.socket
```

- And then send it commands

```
echo “show sess” | \
  socat unix-connect:/tmp/haproxy.socket stdio
```

Backend checking

- Does layer 4 checks by default
- You can also set up HTTP checks
  - So if your apps have a health page, you can actually do pretty decent monitoring just with HAProxy
  - Can't do TCP half-open monitoring
    - I guess you **could** with a custom kernel module and some elbow grease, but it'd be some work.
    - F5s only do this because they run as root and have access to the kernel bits of the TCP stack.
    - Fix your app to not log healthchecks, or filter your logging.

Transparent proxy?

- Yes, if HAProxy is running as root and if the kernel supports TPROXY and HAProxy is compiled with USE_LINUX_TPROXY
- Is it worth that much hassle?
- X-Forwarded-For header insertion lets you log where people were coming from when they hit the backends; you probably don't really need to do IP spoofing.

IPv6 support?

- Yes.

Unix sockets?

- Yes. You can:
  - Direct network traffic to a Unix socket on the HAProxy box using the proxy features
  - Enable a control socket in HAProxy to dynamically tune it via the socket
    - You cannot create new backends this way...but you could create a pool of backends and enable/replumb them dynamically, which would look almost the same to the user
    - https://github.com/flores/haproxyctl is interesting

No file I/O after startup

- If you have a DNS record as a backend target, you get the IP you look up at startup time, and never again.
- HAProxy cannot log to a file; it logs to syslog, and you can specify the collectors in the config file.
  - You could set up a Unix domain socket as a log target, and have a coprocess to read from that socket and write to a file.
  - But it's probably just easier to configure syslog, since that's pretty much exactly what it does already.

Rolling restart already built in

- You can pass the *new* HAProxy process the PID of the *old* one, and it will stop accepting connections on the old one and once the old one has terminated all extant connections, terminate the process.
- This makes updating the config really easy:
  - Upload new config file
  - Call for a rolling restart
  - No one gets booted
  - New connections get new config
- Perfect for incremental bleed-in / bleed-out

Non-commercial Support

- No bug tracker.
- Mailing list and very active development community.
  - Currently a little kerfluffle going on about the amount of spam and the unwillingness of the HAProxy team to take on more mailing list administration duties
  - Haproxy at formilux dot oh arr gee
    - Yes, they allow unauthenticated posts :-(
  - Subscribe by sending an empty email to haproxy+subscribe at the same domain.
- Development stream:
  - https://github.com/haproxy/haproxy

You said commercial support?

- HAProxy Enterprise Edition (HAPEE) – currently 1.4, but 1.5 should be along Real Soon Now.
  - https://www.haproxy.com/products/haproxy-enterprise-edition/
- Or you can buy a 1U appliance:
  - https://www.haproxy.com/products/main-features-of-aloha-load-balancers/aloha-load-balancer-appliance-rack-1u/
- Or a virtual appliance. We don't judge.
  - https://www.haproxy.com/products/main-features-of-aloha-load-balancers/aloha-load-balancer-virtual-appliance/

Live demo?

- http://demo.haproxy.org/

Questions?

athornton at gee mail dot com