Enable Keepalive connections in Nginx Upstream proxy configurations

Mattias Geniar, Tuesday, October 27, 2015 - *last modified: Wednesday, November 4, 2015*

A very common setup to see nowadays is to have an Nginx SSL proxy in front of a Varnish configuration, that handles all the SSL configurations while Varnish still maintains the caching abilities.

It's one of my "2015 server stack predictions" that held up pretty accurately so far. Although, honestly, this wasn't a hard one to predict.

An Nginx configuration however, by default, only talks HTTP/1 to the upstream (in this exampple a Varnish). **Keepalive connections are only supported as of HTTP/1.1, an upgrade to the HTTP protocol.** We have to explicitly enable this setting in Nginx so it does keepalive connections to the upstream it's connecting to.

This can greatly reduce the number of new TCP connections in an Nginx SSL setup, as Nginx can now reuse its existing connections (keepalive) per upstream.

To enable Keepalive in Nginx upstream configurations, add the following to your configs. First, modify your upstream definition and add the keepalive parameter.

```
upstream your_upstream {
  # The keepalive parameter sets the maximum number of idle keepalive
connections
  # to upstream servers that are preserved in the cache of each worker process.
When
  # this number is exceeded, the least recently used connections are closed.
  keepalive 100;

  server 10.5.1.5:80;
  server 10.5.1.6:80;
  ...
}
```

Next, enable the HTTP/1.1 protocol in all upstream requests.

```
location / {
  ...
  proxy_pass             http://your_upstream;
  proxy_read_timeout     300;
  proxy_connect_timeout  300;
  ...

  # Default is HTTP/1, keepalive is only enabled in HTTP/1.1
  proxy_http_version 1.1;
```

```
  # Remove the Connection header if the client sends it,
  # it could be "close" to close a keepalive connection
  proxy_set_header Connection "";
}
```

If your upstream server supports keepalive in its config, Nginx will now reuse existing TCP connections without creating new ones. This can greatly reduce the number of `TIME_WAIT` TCP connections on a busy SSL server.

Hi! My name is Mattias Geniar. I'm a Support Manager at Nucleus Hosting in Belgium, a general web geek, public speaker and podcaster. If you're interested in keeping up with me, have a look at my podcast and weekly newsletter below. For more updates, follow me on Twitter as @mattiasgeniar.