ALOHA Load-Balancer - Application Note

HTTP compression

Document version: v1.1

Last update: 3rd June 2014

EMEA Headquarters
3, rue du petit robinson
ZAC des Metz
78350 Jouy-en-Josas
France
http://www.haproxy.com/

Purpose

HTTP compression is a capability which allows reducing response size by compressing it. It has multiple benefits:

- faster delivery: less bytes to transfer
- reduce bandwidth cost
- smaller footprint in caches

Compression is usually widely deployed on web servers like Apache, nginx and IIS.

Limitation

In order to work properly, the client must announce the compression algorithms it supports and the server must be compatible with at least one of them.

Complexity

Versions concerned

- **ALOHA** 5.5 and above

Changelog

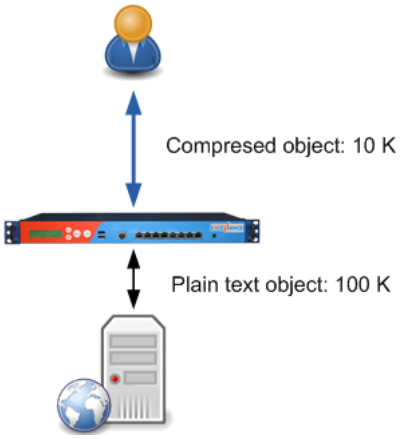
Version	Description
1.1	2014-06-02 – HAProxy Tech. theme update – minor changes
1.0	2013-01-03 Initial release

Synopsys

The **ALOHA** Load-Balancer can perform compression on behalf of servers or can compress on the fly responses that should have been compressed by servers (but obviously wasn't).
The **ALOHA** dynamically updates its compression rate based on its current load.

Diagram

The diagram below shows how compression works when performed on the **ALOHA** load-balancer:



HAProxy and compression

Compression is allowed based on the **Accept-Encoding** HTTP request header: if no header, no compression.
If backend servers support HTTP compression, then **HAProxy** will see a compressed response and will let it pass as is. If backend servers do not support HTTP compression and there is an **Accept-Encoding** header in the request, **HAProxy** will compress the response on the fly.
When offloading compression is turned on on the **ALOHA** Load-Balancer, **HAProxy** removes the **Accept-Encoding** header from the requests before forwarding it to the backend server in order to prevent backend servers from compressing responses.

HTTP Compression is disabled in **HAProxy** when:

- the request does not advertise a supported compression algorithm in the **Accept-Encoding** header
- the response message is not HTTP/1.1.
- HTTP status code is not 200
- response does not contain "**Transfer-Encoding**: chunked" or **Content-Length**
- **Content-Type** response header is "multipart"
- Request contains "Cache-control: no-transform"
- Request **User-Agent** matches "Mozilla/4" except MSIE 6 with XP SP2, or MSIE 7 and later
- Response is already compressed

The compression does not rewrite **Etag** header

Currently **HAProxy** supports gzip compression. Deflate is also supported but should be used in production in any case: its implementation depends on clients and may be broken.

Configuration

Standard usage

The configuration below applies when you want let the server compress the responses. The **ALOHA** watches the traffic and will compress anything that could have been not compressed.
The directive below can be added either in the **default**, **frontend** or **backend** section.

```
compression algo gzip
compression type text/html text/plain text/css text/javascript
```

Compression offload

The configuration below applies when you want to offload compression from the servers. The **ALOHA** will remove the **Accept-Encoding** http request header and will compress responses.
The directive below can be added either in the **default** section or in a **frontend** or a **backend** section.

```
compression algo gzip
compression offload
compression type text/html text/plain text/css text/javascript
```

Such type of configuration can be useful to prevent servers with a weird compression implementation to corrupt responses.