Apache httpd v2.4: Hello Cloud: Buy you a drink?

Jim Jagielski
About me

Jim Jagielski

- Longest still-active developer/contributor
- Co-founder of the ASF
- Member, Director and President
- Director: Outercurve and OSI
- Consulting Engineer with Red Hat
What we will cover

- Overview of Apache httpd 2.4
- General improvements
- Reverse proxy improvements
- How the Cloud is a game-changer for web
- Performance Related Enhancements
Apache httpd 2.4

Currently in final beta release (in vote)

Expected GA: This month!

Significant Improvements

- high-performance
- cloud suitability
Apache httpd 2.4

- Support for async I/O w/o dropping support for older systems
- Larger selection of usable MPMs: added Event, Simple, etc...
- Leverages higher-performant versions of APR
What’s New: Apache httpd 2.4

- Bandwidth control now standard
  - mod_ratelimit

- Finer control of timeouts, esp. during requests
  - mod_reqtimeout
What’s New: Apache httpd 2.4

- Finer control over logging
- per module/per directory
- new logging levels (TRACE[1-8])
- `<If>` supports per-request conditions
- slot-based shared memory capability
What’s New: Apache httpd 2.4

- Controllable buffering of I/O
- mod_buffer
- Support for Lua (??)
- Loadable MPMs
- Proxy improvements (‘natch)
Why Proxy Matters

Cloud puts big focus on horizontal scaling

Apache httpd still the most frequently used front-end

Proxy capabilities must be cloud friendly
Proxy Design Drivers

- Becoming a robust but generic proxy implementation
- Support various protocols
  - HTTP, HTTPS, CONNECT, FTP
  - AJP, FastCGI, SCGI, WSGI (soon)
- Load balancing
- Clustering, failover
- Performance
What's New: Apache httpd 2.4

Reverse Proxy Improvements

- Supports FastCGI, SCGI in balancer
- Additional load balancing mechanisms
- Runtime changing of clusters w/o restarts
- Support for dynamic configuration
- mod_proxy_express
- mod_proxy_html
- mod_fcgid
Load Balancer

mod_proxy_balancer.so

mod_proxy can do native load balancing

- weight by actual requests
- weight by traffic
- weight by busyness
- lbfactors
Load Balancer

Backend connection pooling

Available for named workers:

- eg: `ProxyPass /foo http://bar.example.com`

Reusable connection to origin

- For threaded MPMs, can adjust size of pool (min, max, smax)
- For prefork: singleton

Shared data held in shared memory
Pooling example

<Proxy balancer://foo>

  BalancerMember http://www1.example.com:80/ loadfactor=1
  BalancerMember http://www2.example.com:80/ loadfactor=1
  BalancerMember http://www3.example.com:80/ loadfactor=4
  status=+h

  ProxySet lbmethod=bytraffic

</Proxy>
Load Balancer

- Sticky session support
  - aka “session affinity”
- Cookie based
  - stickysession=PHPSESSID
  - stickysession=JSESSIONID
- Natively easy with Tomcat
- May require more setup for “simple” HTTP proxying
Load Balancer

- Cluster set with failover
- Group backend servers as numbered sets
  - balancer will try lower-valued sets first
  - If no workers are available, will try next set
- Hot standby
Example

<Proxy balancer://foo>

BalancerMember http://php1:8080/ loadfactor=1
BalancerMember http://php2:8080/ loadfactor=4
BalancerMember http://phpbkup:8080/ loadfactor=4 status=+h
BalancerMember http://offsite1:8080/ lbset=1
BalancerMember http://offsite2:8080/ lbset=1

ProxySet lbmethod=bytraffic

</Proxy>

ProxyPass /apps/ balancer://foo/
Embedded Admin

- Allows for real-time
- Monitoring of stats for each worker
- Adjustment of worker params
  - lbset
  - load factor
  - route
  - enabled / disabled
  - ...

This work is licensed under a Creative Commons Attribution 3.0 Unported License.
Embedded Admin

Allows for real-time

- Addition of new workers/nodes
- Change of LB methods
- Can be persistent
- More RESTful
- Can be CLI-driven
Easy setup

<Location /balancer-manager>
  SetHandler balancer-manager
  Order Deny,Allow
  Deny from all
  Allow from 192.168.2.22
</Location>
Admin

Balancer Manager for localhost

LoadBalancer Status for balancer://acna11

MaxMembers StickySession DisableFailover Timeout FailoverAttempts Method
8 [3 Used] (None) Off 2 bytraffic

Worker URL Route RecalcFactor Set Status Elected Busy Load To From
http://www1.example.com 1 0 Init Ok 5 0 0 2.1K 110
http://www2.example.com 1 0 Init Ok 5 0 0 2.1K 110
http://www3.example.com/snap/crackle/pop/ 1 0 Init Stby Ok 0 0 0 0 0

Edit worker settings for http://www3.example.com/snap/crackle/pop/

Load factor: 1
LB Set: 0
Route: 
Route Redirect: Ign Drn Dis Stby
Status: On On On On Off Off Off

Submit 

Click here
## Load Balancer Manager for localhost

**Server Version:** Apache/2.3.15-dev (Unix) DAV/2  
**Server Built:** Nov 1 2011 06:19:34

### LoadBalancer Status for `balancer://acna11`

<table>
<thead>
<tr>
<th>MaxMembers</th>
<th>StickySession</th>
<th>DisableFailover</th>
<th>Timeout</th>
<th>FailoverAttempts</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 [3 Used]</td>
<td>(None)</td>
<td>Off</td>
<td>0</td>
<td>2</td>
<td>bytraffic</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Worker URL</th>
<th>Route</th>
<th>RouteRedir</th>
<th>Factor</th>
<th>Set</th>
<th>Status</th>
<th>Elected Busy Load To</th>
<th>From</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www1.example.com">http://www1.example.com</a></td>
<td>1</td>
<td>0</td>
<td>Init Ok</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2.1K</td>
</tr>
<tr>
<td><a href="http://www2.example.com">http://www2.example.com</a></td>
<td>1</td>
<td>0</td>
<td>Init Ok</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2.1K</td>
</tr>
<tr>
<td><a href="http://www3.example.com/snap/crackle/pop/">http://www3.example.com/snap/crackle/pop/</a></td>
<td>1</td>
<td>0</td>
<td>Init Stby Ok</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Edit balancer settings for `balancer://acna11`

- **LBmethod:** bytraffic
- **Timeout:** 0
- **Failover Attempts:** 2
- **Disable Failover:** On
- **Sticky Session:** Off
- **Add New Worker:** (Use '-' to delete)

[Submit]
Admin

Changing the LB method

Adding new worker
Admin

Load Balancer Manager for localhost

Server Version: Apache/2.3.15-dev (Unix) DAV/2
Server Built: Nov 1 2011 06:19:34

LoadBalancer Status for balancer://acna11

MaxMembers StickySession DisableFailover Timeout FailoverAttempts Method
8 [4 Used] (None) Off 0 2 byrequests

<table>
<thead>
<tr>
<th>Worker URL</th>
<th>Route</th>
<th>RouteRedir Factor Set</th>
<th>Status</th>
<th>Elected Busy Load To</th>
<th>From</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www1.example.com">http://www1.example.com</a></td>
<td>1</td>
<td>0</td>
<td>Init Ok</td>
<td>5 0 0 2.1K 110</td>
<td></td>
</tr>
<tr>
<td><a href="http://www2.example.com">http://www2.example.com</a></td>
<td>1</td>
<td>0</td>
<td>Init Ok</td>
<td>5 0 0 2.1K 110</td>
<td></td>
</tr>
<tr>
<td><a href="http://www3.example.com/snap/crackle/top/">http://www3.example.com/snap/crackle/top/</a></td>
<td>1</td>
<td>0</td>
<td>Init Sby Ok</td>
<td>0 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td><a href="http://www4.example.com/acna">http://www4.example.com/acna</a></td>
<td>1</td>
<td>0</td>
<td>Init Dis</td>
<td>0 0 0 0 0</td>
<td></td>
</tr>
</tbody>
</table>

Edit balancer settings for balancer://acna11

LBmethod: byrequests
Timeout: 0
Failover Attempts: 2
Disable Failover: Off
Sticky Session: (Use 'v' to delete)
Add New Worker: Are you sure?
Submit
Putting it all together

<Proxy balancer://foo>
  BalancerMember http://php1:8080/   loadfactor=1
  BalancerMember http://php2:8080/   loadfactor=4
  BalancerMember http://phpbkup:8080/ loadfactor=4 status=+h
  BalancerMember http://phpexp:8080/  lbset=1
  ProxySet lbmethod=bytraffic
</Proxy>

<Proxy balancer://javaapps>
  BalancerMember ajp://tc1:8089/   loadfactor=1
  BalancerMember ajp://tc2:8089/   loadfactor=4
  ProxySet lbmethod=byrequests
</Proxy>

ProxyPass /apps/ balancer://foo/
ProxyPassReverse /apps/ balancer://foo/
ProxyPass /serv/ balancer://javaapps/
ProxyPass /images/ http://images:8080/
What’s on the horizon?

- Improving AJP
- Adding additional protocols
- More dynamic configuration
- Adding balancers!
Cloud and Performance

The Cloud is a game changer for web servers

- Horizontal scalability is no longer as painful
- Concurrency is somewhat minimized
- What’s important now? Transaction Time!
- Low latency
- Fast req/resp turnover
- Does density still matter? Of course!
Apache httpd vs nginx

Benchmark: local and reverse proxy transaction times

Apache httpd 2.3.15-dev, nginx 1.1.6
Fedora 14, Xeon 2.28GHz
1GB memory
localhost loopback and external (no firewall)
Double check results: OSX, Ubuntu 10.04
Setup

Setup 1:

Setup 2:  

Setup 3:

Setup 3:
Considerations

Multiple benchmarking systems:

- **flood** (50/150/5/2, 50/100/5/2, 50/5/5/2)
- **httpperf** (num-conns=100->1000, numcalls=3)

Full URL requests (**www.example.com/index.html**)

Static local requests

Static reverse proxy requests

All Apache httpd MPMs

No significant “tuning” efforts (out of the box configs)
nginx vs Event (typical)

The figure compares the performance of nginx and Apache in the Event MPM mode. The graph shows the number of operations (Open, Write, Read, Close) over time, with nginx on the left and Apache on the right. The color coding indicates the type of operation:

- Open: Blue
- Write: Green
- Read: Yellow
- Close: Red

The y-axis represents the number of operations, and the x-axis represents time. The graph illustrates the typical performance differences between the two web servers in an event-driven environment.
nginx vs Worker (typical)
nginx vs Prefork *(typical)*

- **Open**
- **Write**
- **Read**
- **Close**

This work is licensed under a Creative Commons Attribution 3.0 Unported License.
Focus on open()
Focus on write()
Focus on read()
Total req/resp time

Comparison - total transaction (close)
Benchmark Conclusions

- Events, polling and fork/spawn creates overhead: good for “more bang for buck” system, bad for performance for *that* request
- For concurrency, Event&Worker on par with nginx
- For transaction speed, prefork shines
- Let’s reboot “Simple” mpm

Main Caveats:

- Apache is never resource starved
- More work can (and should) be done
Performance of Apache httpd 2.4 still in the big leagues

For cloud environs, the performance and dynamic control of Apache httpd 2.4 in reverse proxies is just what the Dr. ordered

Architecture of Apache httpd 2.4 allows a lot of room for growth and additional functionality (both for the cloud and not)