Apache httpd v2.4: Hello Cloud

Jim Jagielski
Introduction

Jim Jagielski

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

• Performance Related Enhancements
• Reverse Proxy Server Enhancements
Apache httpd 2.4

- Currently in beta release
- Expected GA: This May!

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
Apache httpd 2.4

- Bandwidth control now standard
- Finer control of timeouts, esp. during requests
- Controllable buffering of I/O
- Support for Lua
Reverse Proxy Improvements

- Supports FastCGI, SCGI
- Additional load balancing mechanisms
- Runtime changing of clusters w/o restarts
- Support for dynamic configuration
mod_proxy

- An Apache module
- Implements core proxy capability
- Both forward and reverse proxy
- In general, most people use it for reverse proxy (gateway) functionality
How did we get here?

- A stroll down mod_proxy lane
  - First available in Apache 1.1
- "Experimental Caching Proxy Server"
  - In Apache 1.2, pretty stable, but just HTTP/1.0
  - In Apache 1.3, much improved with added support for HTTP/1.1
  - In Apache 2.0, break out cache and proxy
  - In Apache 2.2, lay framework
Proxy Improvements

- Becoming a robust but generic proxy implementation
- Support various protocols
  - HTTP, HTTPS, CONNECT, FTP
  - AJP, FastCGI, SCGI, WSGI (soon)
  - Load balancing
- Clustering, failover
AJP? Really?

- Yep, Apache can now talk AJP with Tomcat directly
- `mod_proxy_ajp` is the magic mojo
- Other proxy improvements make this even more exciting
- `mod_jk` alternative
But I like mod_jk

- That’s fine, but...
  - Now the config is much easier and more consistent
    
    ProxyPass /servlets ajp://tc.example.com:8089
  
  - Easier when Apache needs to proxy both HTTP and AJP
  
  - Leverage improvements in proxy module
Features of Proxy Server

• Performance
• Monitoring
• Filtering
• Caching (with mod_cache)
Reverse Proxy

- Operated at the server end of the transaction
- Completely transparent to the Web Browser – thinks the Reverse Proxy Server is the real server

Browser ➔ Internet ➔ Reverse Proxy Server ➔ Cloud ➔ Firewall ➔ Transactional Servers
Features of Reverse Proxy

- **Security**
  - Uniform security policy can be administered
  - The real transactional servers are behind the firewall

- **Delegation, Specialization, Load Balancing**
Configuring Reverse Proxy

- Set ProxyRequests Off
- Apply ProxyPass, ProxyPassReverse and possibly RewriteRule directives
Reverse Proxy Directives:

- Allows remote server to be mapped into the space of the local (Reverse Proxy) server

- Example:
  - ProxyPass /secure/ http://secureserver/
  - Presumably “secureserver” is inaccessible directly from the internet
Reverse Proxy Directives:

- Used to specify that redirects issued by the remote server are to be translated to use the proxy before being returned to the client.

- Syntax is identical to `ProxyPass`; used in conjunction with it.

- Example:
  - `ProxyPass /secure/ http://secureserver/`
Simple Rev Proxy

- All requests for /images to a backend server
  - ProxyPass /images http://images.example.com/
  - ProxyPass <path> <scheme>://<full url>

- Useful, but limited

- What if:
  - images.example.com dies?
  - traffic for /images increases
Baby got back

- We need more backend servers
- And balance the load between them
- Before 2.2, mod_rewrite was your only option
- Some people would prefer spending an evening with an Life Insurance salesman rather than deal with mod_rewrite
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

- LB algorithms are implemented as providers
  - easy to add
  - no core code changes required
  - growing list of methods
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

Load Balancer Manager for localhost

Server Version: Apache/2.3.12-dev (Unix) DAV/2
Server Built: Mar 10 2011 11:45:09

LoadBalancer Status for balancer://foo

MaxMembers  StickySession  DisableFailover  Timeout  FailoverAttempts  Method
8  [3 Used]  (None)  Off  0  2  bytraffic

<table>
<thead>
<tr>
<th>Worker URL</th>
<th>Route</th>
<th>RouteRedir</th>
<th>Factor Set</th>
<th>Status</th>
<th>Elected To From</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www2.jagunet.com">http://www2.jagunet.com</a></td>
<td>1</td>
<td>0</td>
<td></td>
<td>Init Ok</td>
<td>0 0 0</td>
</tr>
<tr>
<td><a href="http://www3.example.com">http://www3.example.com</a></td>
<td>1</td>
<td>0</td>
<td></td>
<td>Init Ok</td>
<td>0 0 0</td>
</tr>
<tr>
<td><a href="http://www4.example.com/snap/crackle/pcp/">http://www4.example.com/snap/crackle/pcp/</a></td>
<td>1</td>
<td>0</td>
<td></td>
<td>Init Stby Ok</td>
<td>0 0 0</td>
</tr>
</tbody>
</table>
Admin

Load Balancer Manager for localhost

Server Version: Apache/2.3.12-dev (Unix) DAV/2
Server Built: Mar 10 2011 11:45:09

LoadBalancer Status for balancer:/foo

<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</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 To</th>
<th>From</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www1.example.com">http://www1.example.com</a></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Init Ok</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www2.example.com">http://www2.example.com</a></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Init Ok</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www3.example.com/snap/crackle/pop/">http://www3.example.com/snap/crackle/pop/</a></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Init Sby Ok</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Load factor:</th>
<th>LB Set:</th>
<th>Route:</th>
<th>Route Redirect:</th>
<th>Ign</th>
<th>Drn</th>
<th>Dis</th>
<th>Stby</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td>On</td>
<td>Off</td>
<td>On</td>
<td>On</td>
</tr>
</tbody>
</table>

Submit
Admin

Load Balancer Manager for localhost

Server Version: Apache/2.3.12-dev (Unix) DAV/2
Server Built: Mar 10 2011 11:45:09

LoadBalancer Status for balancer://foo

<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>Off</td>
<td>0</td>
<td>bytraffic</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Worker URL</th>
<th>Route</th>
<th>RouteDir</th>
<th>Factor</th>
<th>Set</th>
<th>Status</th>
<th>Elected 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>InitOk</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www2.example.com">http://www2.example.com</a></td>
<td>1</td>
<td>0</td>
<td>InitOk</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</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://foo

- **LBmethod**: byrequests
- **Timeout**: 0
- **Failover Attempts**: 2
- **Disable Failover**: On
- **Sticky Session**: Off
- **Add New Worker**: http://www4.example.com/poss

This work is licensed under a Creative Commons Attribution 3.0 Unported License.
Load Balancer Manager for localhost

Server Version: Apache/2.3.12-dev (Unix) DAV/2
Server Built: Mar 10 2011 11:45:09

LoadBalancer Status for \texttt{balancer://foo}

\begin{tabular}{|c|c|c|c|}
\hline
MaxMembers & StickySession & DisableFailover & Timeout & FailoverAttempts & Method  \\
8 & 0 & Off & 0 & 2 & byrequests \\
\hline
\end{tabular}

\begin{tabular}{|l|c|c|c|c|c|c|}
\hline
Worker URL & Route & RouteRedir & Factor & Set & Status & Elected To From  \\
http://www1.example.com & 1 & 0 & Init Ok & 0 & 0 & 0  \\
http://www2.example.com & 1 & 0 & Init Ok & 0 & 0 & 0  \\
http://www3.example.com/snap/oracle/pop/ & 1 & 0 & InitSby Ok & 0 & 0 & 0  \\
http://www4.example.com/posscon/ & 1 & 0 & Init DIS & 0 & 0 & 0  \\
\hline
\end{tabular}

Edit balancer settings for balancer://foo

\begin{itemize}
\item LBmethod: byrequests
\item Timeout: 
\item Failover Attempts: 2
\item Disable Failover: On
\item Sticky Session: 
\end{itemize}

Add New Worker: Are you sure? 

Submit
Some tuning params

• For workers:
  – loadfactor
    • normalized load for worker [1]
  – lbset
    • worker cluster number [0]
  – retry
    • retry timeout, in seconds, for failed workers [60]
Some tuning params

- For workers - connection pool:
  - min
    - Initial number of connections [0]
  - max
    - Hard maximum number of connections [1 | TPC]
  - smax:
    - soft max - keep this number available [max]
    - time to live for connections above smax
Some tuning params

• For workers - connection pool:

  – disable reuse:
    • bypass the connection pool

  – ttl
    • time to live for connections above smax
Some tuning params

For workers (cont):

- connectiontimeout/timout
  - Connection timeouts on backend [ProxyTimeout]

- flushpackets *
  - Does proxy need to flush data with each chunk of data?
    - on: Yes  |  off: No  |  auto: wait and see

- flushwait *
  - ms to wait for data before flushing
For workers (cont):

- status (+/-)
  - D : disabled
  - S : Stopped
  - I : Ignore errors
  - H : Hot standby
  - E : Error
Some tuning params

For balancers:

- **lbmethod**
  - load balancing algo to use [byrequests]

- **stickysession**
  - sticky session name (eg: PHPSESSIONID)

- **maxattempts**
  - failover tries before we bail

- **Nofailover**
  - Back-ends don't support failover so don't send session when failing over
Recent improvements

- **ProxyPassMatch**
  - ProxyPass can now take regex’s instead of just “paths”
    - `ProxyPassMatch ^(/.*\.gif)$ http://backend.example.com$1`

- JkMount migration

- Or

  - `ProxyPass ~ ^(/.*\.gif)$ http://backend.example.com$1`

- **mod_rewrite** is balancer aware
Recent improvements

- **ProxyPassReverse** is NOW balancer aware!
- The below will work:

```xml
<Proxy balancer://foo>
  BalancerMember http://php1:8080/ loadfactor=1
  BalancerMember http://php2:8080/ loadfactor=4
</Proxy>

ProxyPass /apps/ balancer://foo/

ProxyPassReverse /apps balancer://foo/
```
Useful Envars

• **BALANCER_SESSION_STICKY**
  - This is assigned the `stickysession` value used in the current request. It is the cookie or parameter name used for sticky sessions.

• **BALANCER_SESSION_ROUTE**
  - This is assigned the `route` parsed from the current request.

• **BALANCER_NAME**
  - This is assigned the name of the balancer used for the current request. The value is something like `balancer://foo`. 
Useful Envars

- **BALANCER_WORKER_NAME**
  - This is assigned the name of the worker used for the current request. The value is something like `http://hostA:1234`.

- **BALANCER_WORKER_ROUTE**
  - This is assigned the route of the worker that will be used for the current request.

- **BALANCER_ROUTE_CHANGED**
  - This is set to 1 if the session route does not match the worker route (BALANCER_SESSION_ROUTE != BALANCER_WORKER_ROUTE) or the session does not yet have an established route. This can be used to determine when/if the client needs to be sent an updated route when sticky sessions are used.
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>
Putting it all together

ProxyPass /apps/ balancer://foo/

ProxyPass /serv/ balancer://javaapps/

ProxyPass /images/ http://images:8080/
Manipulating HTTP Headers:

- Modify HTTP request and response headers
  - Can be used in Main server, Vhost, Directory, Location, Files sections
  - Headers can be merged, replaced or removed
  - Pass on client-specific data to the backend server
- IP Address, Request scheme (HTTP, HTTPS), UserAgent, SSL connection info, etc.
Manipulating HTTP Headers:

- Shield backend server’s info from the clients
  - Strip out Server name
  - Server IP address
  - etc.
Header examples

• Copy all request headers that begin with “TS” to response headers
  – Header echo ^TS

• Say hello to Joe
  – Header add JoeHeader “Hello Joe!”

• If header “MyRequestHeader: value” is present, response will contain “MyHeader” header:
  – SetEnvIf MyRequestHeader value HAVE_MyRequestHeader
  – Header add MyHeader “%D %t mytext” env=HAVE_MyRequestHeader
Header examples

• Remember, sequence is important! Following will result in “MHeader” to be stripped from the response:
  – RequestHeader append MyHeader “value1”
  – RequestHeader append MyHeader “value2”
  – RequestHeader unset MyHeader
Example:

- **Pass additional info about Client Browsers to the App Server:**
  
  ```
  ProxyPass / http://backend.covalent.net
  ProxyPassReverse / http://backend.covalent.net
  RequestHeader set X-Forwarded-IP %{REMOTE_ADDR}e
  RequestHeader set X-Request-Scheme %{REQUEST_SCHEME}e
  ```

- **App Server receives the following HTTP headers:**
  
  - X-Forwarded-IP: 10.0.0.3
  - X-Request-Scheme: https
# mod_proxy lb example using request parameter

RewriteEngine On

# Use mod_rewrite to insert a node name into the url

RewriteCond %{QUERY_STRING} accountId=.*([0-2])\b
RewriteRule ^/sampleApp/(.*) balancer://tc1/$1 [P]

RewriteCond %{QUERY_STRING} accountId=.*([3-6])\b
RewriteRule ^/sampleApp/(.*) balancer://tc2/$1 [P]

RewriteCond %{QUERY_STRING} accountId=.*([7-9])\b
RewriteRule ^/sampleApp/(.*) balancer://tc3/$1 [P]

# No ID - round robin to all nodes

ProxyPass /sampleApp/ balancer://all/
<Proxy balancer://tcl>

# Default worker for this balancer
BalancerMember http://linux6401.dev.local:8080/sampleApp lbset=1

# Backup balancers for node failure - used in round robin
# no stickyness
BalancerMember http://linux6402.dev.local:8081/sampleApp lbset=1 status=H
BalancerMember http://linux6403.dev.local:8081/sampleApp lbset=1 status=H

# Maintenance balancer used to re-route traffic for upgrades etc
BalancerMember http://linux6404.dev.local:8080/sampleApp status=D

</Proxy>
Using mod-rewrite example

<Proxy balancer://tc2>

 BalancerMember http://linux6402.dev.local:8080/sampleApp lbset=1

# Backup balancers for node failure - used in round robin
# no stickyness
BalancerMember http://linux6401.dev.local:8081/sampleApp lbset=1 status=H
BalancerMember http://linux6403.dev.local:8081/sampleApp lbset=1 status=H

# Maintenance balancer used to re-route traffic for upgrades etc
BalancerMember http://linux6404.dev.local:8080/sampleApp status=D

</Proxy>
Using mod-rewrite example

<Proxy balancer://tc3>

BalancerMember http://linux6403.dev.local:8080/sampleApp lbset=1

# Backup balancers for node failure - used in round robin
# no stickyness
BalancerMember http://linux6401.dev.local:8081/sampleApp lbset=1 status=H
BalancerMember http://linux6402.dev.local:8081/sampleApp lbset=1 status=H

# Maintenance balancer used to re-route traffic for upgrades etc
BalancerMember http://linux6404.dev.local:8080/sampleApp status=D

</Proxy>
Using mod-rewrite example

<Proxy balancer://all>
  BalancerMember http://linux6401:8080/sampleApp
  BalancerMember http://linux6402:8080/sampleApp
  BalancerMember http://linux6403:8080/sampleApp
</Proxy>

<Location /balancer-manager>
  SetHandler balancer-manager
  Order deny,allow
  Deny from all
  Allow from .dev.local
</Location>
What’s on the horizon?

- Improving AJP
- Adding additional protocols
- mass_vhost like clusters/proxies
- More dynamic configuration