# Advanced Reverse Proxy Load Balancing in Apache HTTP Server 2.2

Jim Jagielski

http://www.jimjag.com/

jim@jaguNET.com

ApacheCon

Leading the Wave of Open Source

# Whew

- That’s a mouthful

# About me

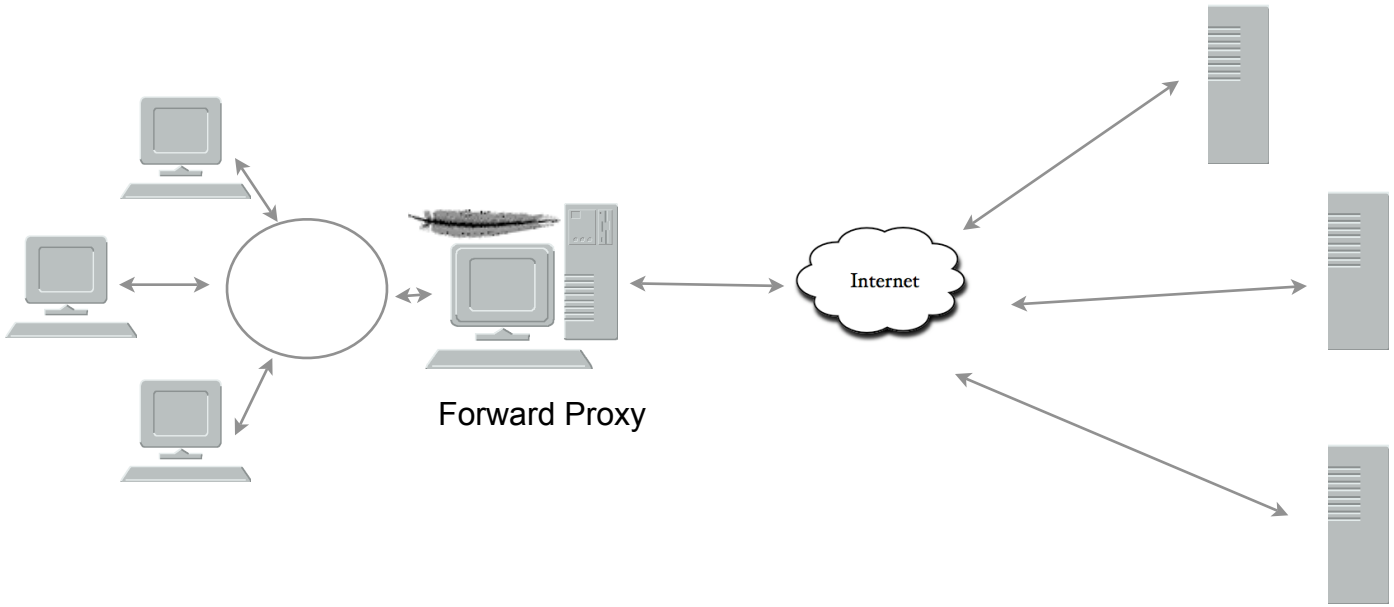
- Longtime active contributor (July/Aug 1995)
- Been giving mod_proxy much TLC
- ASF Co-founder
- Other ASF titles as well
- Chief Architect at Springsource
- Husband, father, all around nice guy

# mod_proxy? Wazzat?

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

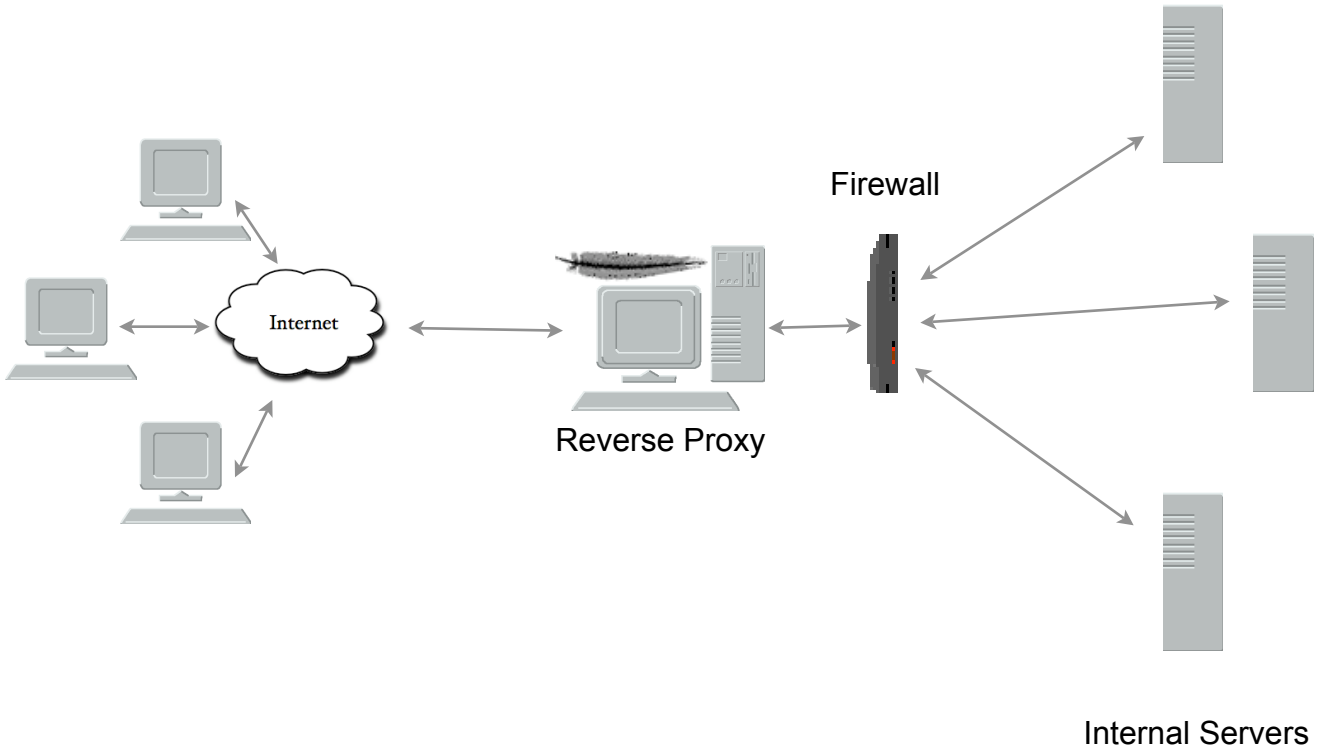
# Forward Proxy

- Intent is to “protect” internal clients



# Reverse Proxy

- Intent is to “protect” internal Servers



# How did we get here?

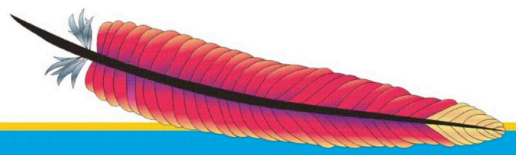
- A stroll down mod_proxy lane
  - First available in Apache 1.1
    - "Experimental Caching Proxy Server"
  - In Apache 1.2, pretty stable, but just HTTP/1.0
  - In Apache 1.3, much improved with added support for HTTP/1.1
  - In Apache 2.0, break out cache and proxy

# What's new/improved in 2.2

- Large file support
- Graceful stop
- mod_dbd
- mod_filter
- Better Debugging and info
- Caching
- Event MPM
- Authn/Authz
- ***Proxy***

# Goal for mod_proxy in 2.2

- Suck less ass



# Proxy Improvements

- Becoming a robust but generic proxy implementation
- Support various protocols
  - HTTP, HTTPS, CONNECT, FTP
  - AJP, FastCGI (coming “soonish”)
- 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

# mod_proxy Directives

- ProxyPass
- ProxyPassReverse
- <Proxy ... >
- ProxySet
- But NOT ProxyRequests

# Huh??

- Yep, you do not set ProxyRequests to On
  - This is just for forward proxies
  - You don't need this for Reverse Proxy functionality
  - Setting it to On will make you very, very sad

# Simple Rev Proxy

- All requests for /images to a backend server
  - ProxyPass /images http://images.example.com/
- 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
- LB algo's are impl as providers
  - easy to add
  - no core code changes required

# Providers? Wazzat?

- New feature of Apache 2.x
- Originally used mostly in mod_dav
- Then in caching
- Now in other places too
  - authn / authz
  - mod_proxy

# Providers... so what

- Think of providers as providing services
- modules implement providers and register them
- Other modules can then use those providers to implement that “service”

# Why cool for mod_proxy?

- We mentioned that right now, we balance by traffic, requests and busyness
- But what if you want some other method (eg: ByPhaseOfTheMoon)
- You can add that capability with no core code changes to Apache.
- Very flexible

# 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 scoreboard

# Pooling example

```
<Proxy balancer://foo>
  BalancerMember http://www1.example.com:80/  loadfactor=1
  BalancerMember http://www3.example.com:80/  loadfactor=1
  BalancerMember http://www2.example.com:80/  loadfactor=4 status=+h
  ProxySet lbmethod=bytraffic
</Proxy>
```

```
proxy: grabbed scoreboard slot 0 in child 371 for worker http://www1.example.com/
proxy: initialized single connection worker 0 in child 371 for (www1.example.com)
proxy: grabbed scoreboard slot 0 in child 369 for worker http://www1.example.com/
proxy: worker http://www1.example.com/ already initialized
proxy: grabbed scoreboard slot 0 in child 372 for worker http://www1.example.com/
proxy: worker http://www1.example.com/ already initialized
proxy: grabbed scoreboard slot 2 in child 371 for worker http://www3.example.com/
proxy: initialized single connection worker 2 in child 371 for (www3.example.com)
proxy: initialized single connection worker 0 in child 369 for (www1.example.com)
proxy: grabbed scoreboard slot 2 in child 369 for worker http://www3.example.com/
...
proxy: grabbed scoreboard slot 6 in child 369 for worker proxy:reverse
proxy: initialized single connection worker 6 in child 369 for (*)
proxy: grabbed scoreboard slot 6 in child 372 for worker proxy:reverse
proxy: worker proxy:reverse already initialized
proxy: grabbed scoreboard slot 1 in child 369 for worker http://www1.example.com/
proxy: initialized single connection worker 6 in child 372 for (*)
```

# Workers and worker

- Don’t get too confused
- Both the worker MPM and the proxy balancer use the term “worker”

# 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
  - Do you really want/need it?

# Load Balancer

- Cluster set with failover
  - Lump 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/
```

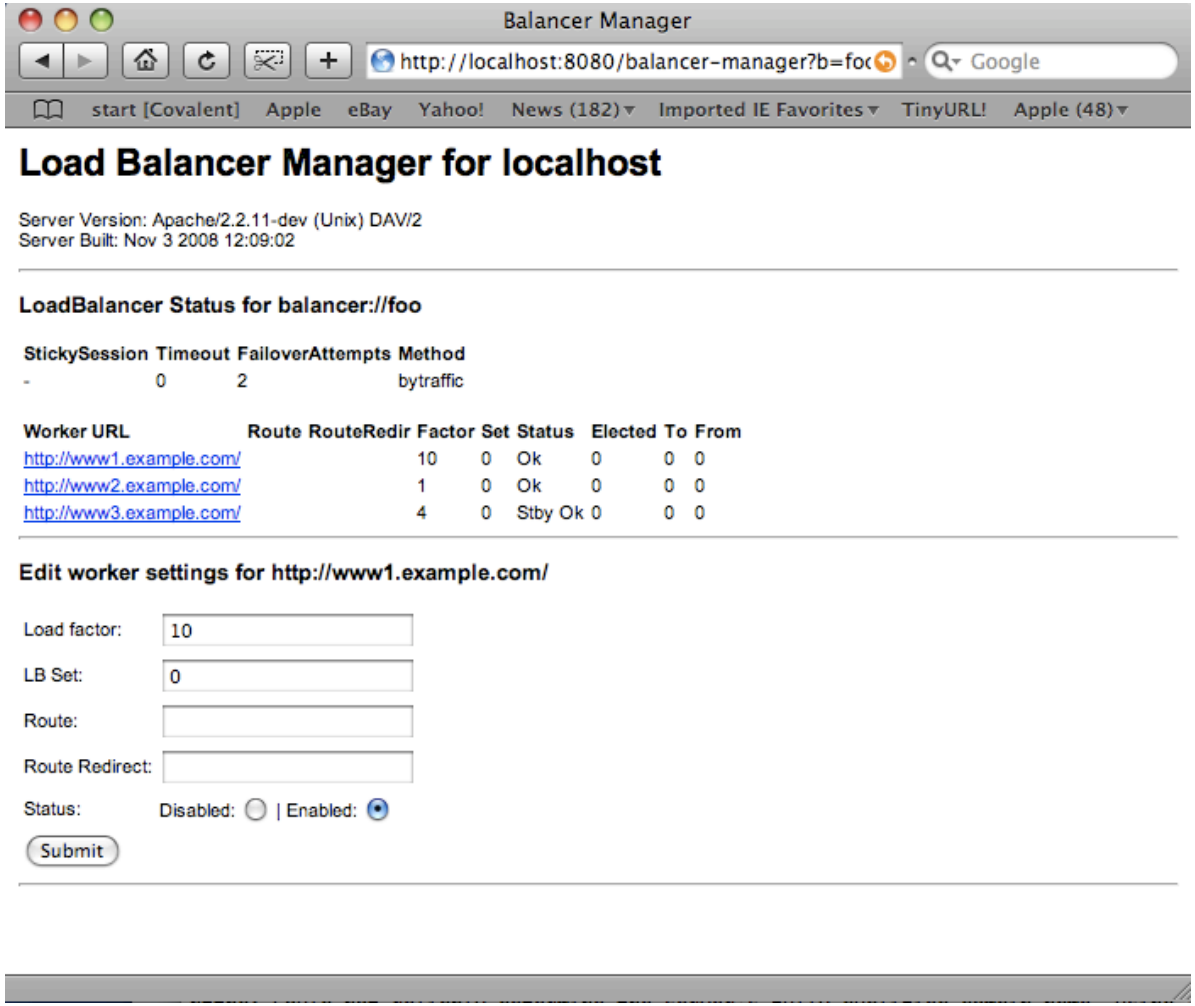
# Embedded Admin

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

# Easy setup

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

# Admin


Balancer Manager
http://localhost:8080/balancer-manager?b=foc
Google
start [Covalent] Apple eBay Yahoo! News (182) Imported IE Favorites TinyURL! Apple (48)
Load Balancer Manager for localhost
Server Version: Apache/2.2.11-dev (Unix) DAV/2
Server Built: Nov 3 2008 12:09:02
LoadBalancer Status for balancer://foo
StickySession Timeout FailoverAttempts Method
- 0 2 bytraffic
Worker URL Route RouteRedir Factor Set Status Elected To From
http://www1.example.com/ 10 0 Ok 0 0 0
http://www2.example.com/ 1 0 Ok 0 0 0
http://www3.example.com/ 4 0 Stby Ok 0 0 0
Edit worker settings for http://www1.example.com/
Load factor: 10
LB Set: 0
Route:
Route Redirect:
Status: Disabled: | Enabled:
Submit

# Some tuning params

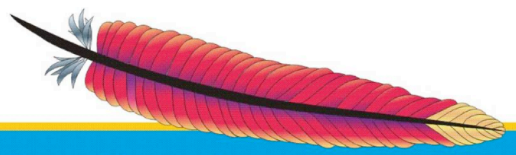
- For workers:
  - loadfactor
    - normalized load for worker [1]
  - lbset
    - worker cluster number [0]
  - retry
    - retry timeout, in seconds, for non-ready workers [60]

# 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]

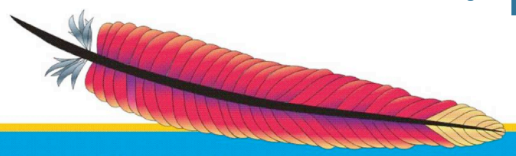
# Some tuning params

- For workers - connection pool:
  - disablereuser:
    - 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



# Some tuning params

- For workers (cont):
  - ping *
    - Ping backend to check for availability; value is time to wait for response
  - 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

# Some tuning params

- For balancers:
  - nofailover
    - pretty freakin obvious
- For both:
  - ProxySet
    - Alternate method to set various params

```
ProxySet balancer://foo timeout=10
  ...
ProxyPass / balancer://foo timeout=10
```

# Oh yeah

- ProxyPassMatch
  - ProxyPass can now take regex's instead of just "paths"
    - `ProxyPassMatch ^(/.*\.gif)$ http://backend.example.com$1`
  - JkMount migration
- Shhhh
  - `ProxyPass ~ ^(/.*\.gif)$ http://backend.example.com$1`
- mod_rewrite is balancer aware

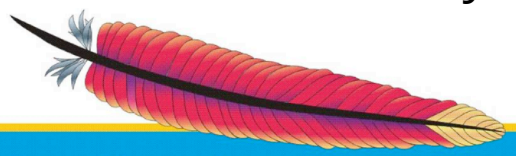
# Neat

- ProxyPassReverse is NOW balancer aware! (as of 2.2.9)
- The below will work:

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

ProxyPass /apps/ balancer://foo/

ProxyPassReverse /apps balancer://foo/
```



# Workaround for <=2.2.8

- Instead, do this

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

ProxyPass /apps/ balancer://foo/

ProxyPassReverse /apps http://php1:8080/
ProxyPassReverse /apps http://php2:8080/
```

# 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>

ProxyPass /apps/ balancer://foo/
ProxyPass /serv/ balancer://javaapps/

ProxyPass /images/ http://images:8080/
```

# What's on the horizon?

- Some additional potential backports
  - FastCGI proxy module
  - HTTP "ping" (OPTIONS *)
- More LB methods
- Enhancing ProxyPassReverse
- Even Better RFC compliance
- Improving AJP

# Thanks!

- Q&A
- Resources:
  - http://httpd.apache.org/
  - dev@httpd.apache.org
  - A certain Open Source support provider