Apache™
HTTP SERVER PROJECT

Apache httpd v2.4:
It’s Not Your Daddy’s Web Server!

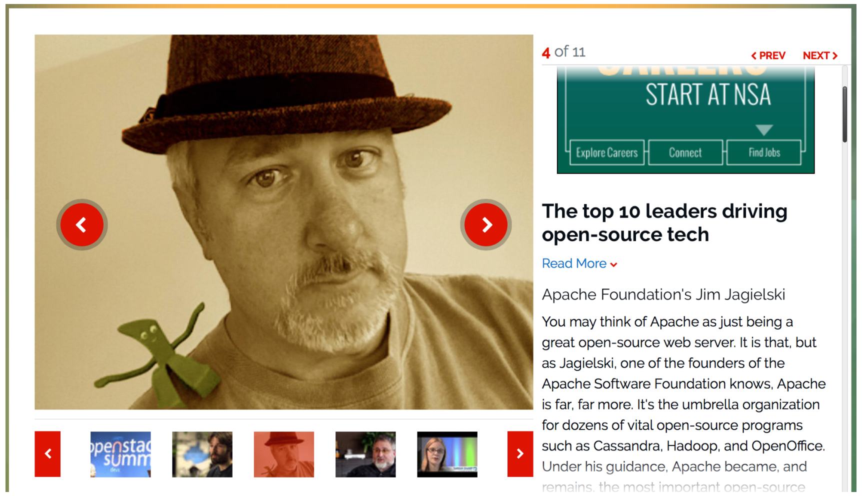
Jim Jagielski

@jimjag

APACHE
ROADSHOW
Berlin, June 11-14, 2018

About Me

- Apache Software Foundation
  - Co-founder, Director Emeritus, Member and Developer
- Director Emeritus
  - Outercurve, MARSEC-XL, OSSI, OSI (ex)…
- Developer
  - Mega FOSS projects
- O’Reilly Open Source Award: 2013
- European Commission: Luminary Award
- Open Source Chef: ConsenSys



The top 10 leaders driving open-source tech

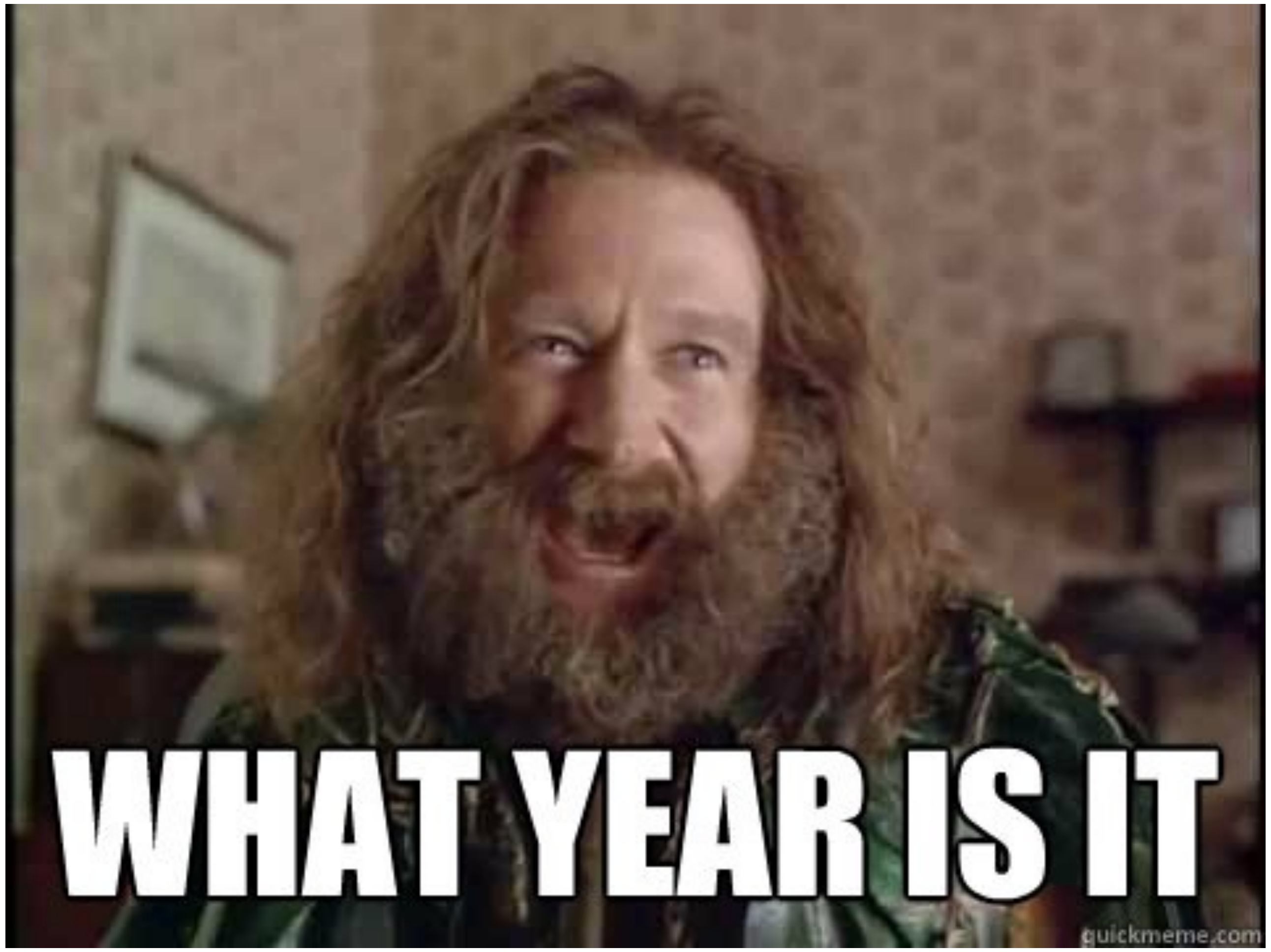
Read More ⌄

Apache Foundation's Jim Jagielski

You may think of Apache as just being a great open-source web server. It is that, but as Jagielski, one of the founders of the Apache Software Foundation knows, Apache is far, far more. It's the umbrella organization for dozens of vital open-source programs such as Cassandra, Hadoop, and OpenOffice. Under his guidance, Apache became, and remains, the most important open-source

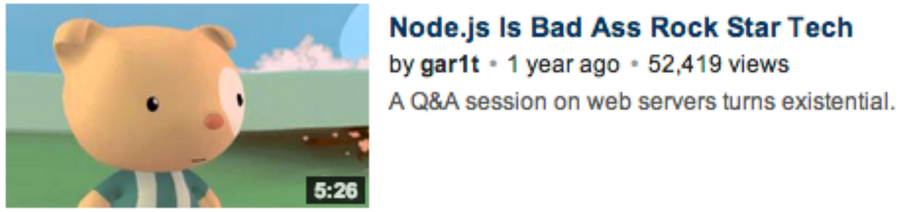
Hold on a tic

- How do you define “new”??



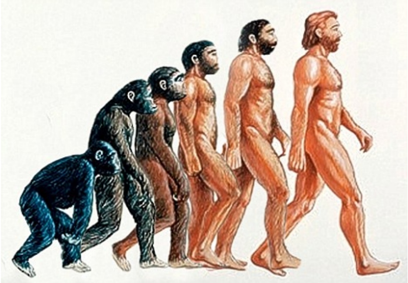
httpd is sooo old school (aka fud)

- Apache doesn’t scale (its SLOW)
  - http://www.youtube.com/watch?v=bzkRVzciAZg



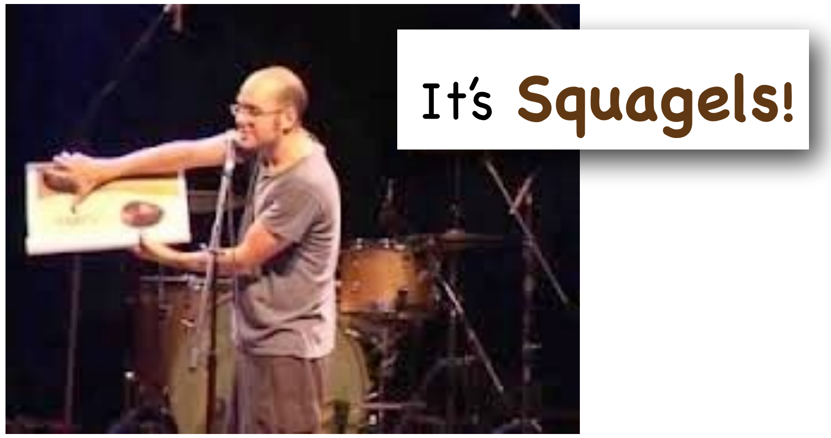
Node.js Is Bad Ass Rock Star Tech
by gar1t · 1 year ago · 52,419 views
A Q&A session on web servers turns existential.

- Apache is too generalized



VS

- Apache is too complex (config file)
  - really?
- Apache is too old
  (yeah, just like Linux)



APACHE ROADSHOW
Berlin, June 11-14, 2018

@jimjag

Apache httpd 2.4 - design drivers

- New features and improve old ones
- Support for async I/O w/o dropping support for older systems
- Larger selection of usable MPMs: added Event, Motorz, etc...
- Leverage higher-performant versions of APR
- Increase performance
- Reduce memory utilization
- The Cloud

Currently at version 2.4.33 (2.4.1 went GA Feb 21, 2012)

APACHE ROADSHOW Berlin, June 11-14, 2018

@jimjag

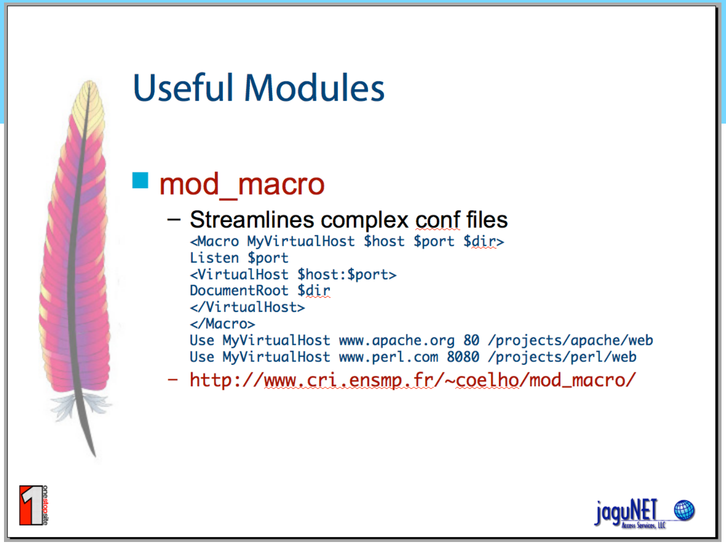
What’s New: Apache httpd 2.4

- Configuration / Runtime Improvements
- New Modules / Capabilities
- Cloud / Proxy Enhancements
- Performance Increases
- HTTP/2

Configuration - Runtime

- mod_macro

From my ApacheCon 2000 Preso



Useful Modules

- mod_macro
  - Streamlines complex conf files

```
<Macro MyVirtualHost $host $port $dir>
Listen $port
<VirtualHost $host:$port>
DocumentRoot $dir
</VirtualHost>
</Macro>
Use MyVirtualHost www.apache.org 80 /projects/apache/web
Use MyVirtualHost www.perl.com 8080 /projects/perl/web
```

  - http://www.cri.ensmp.fr/~coelho/mod_macro/

```
<Macro VHost $name $domain>
<VirtualHost *:80>
    ServerName $domain
    ServerAlias www.$domain

    DocumentRoot /var/www/vhosts/$name
    ErrorLog /var/log/httpd/$name.error_log
    CustomLog /var/log/httpd/$name.access_log combined
</VirtualHost>
</Macro>

Use VHost example example.com
Use VHost myhost hostname.org
Use VHost apache apache.org

UndefMacro VHost
```

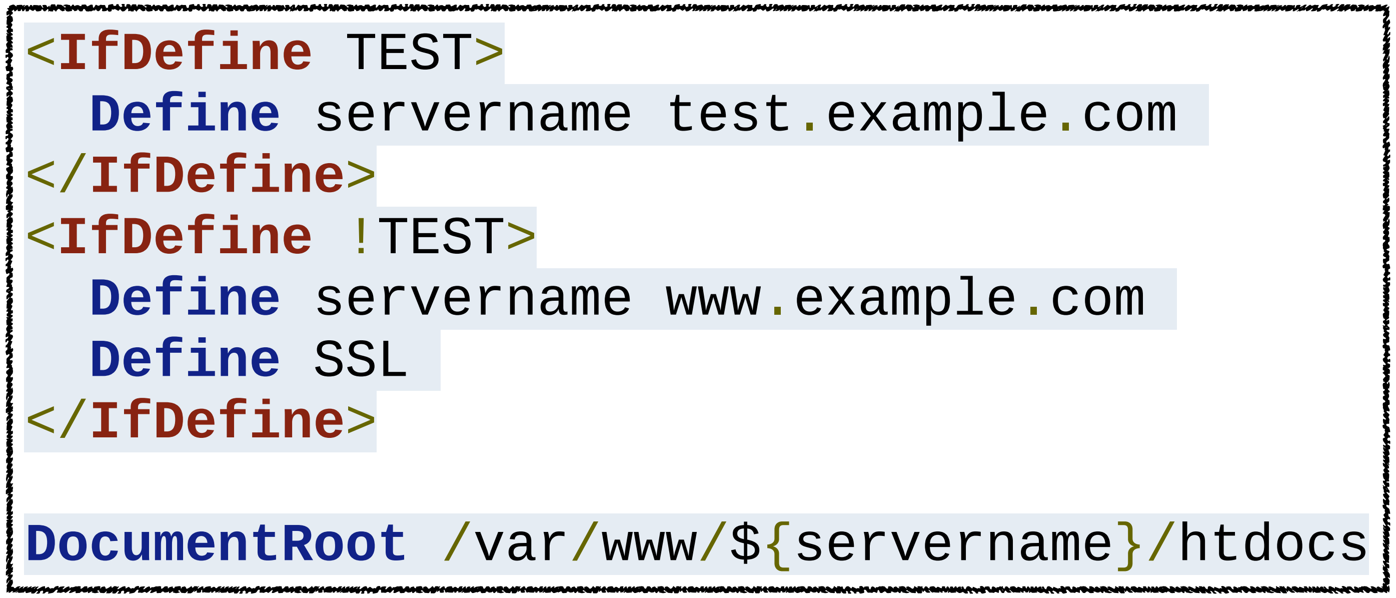
Configuration - Runtime

- ➡ <If> supports per-request conditions

```
# Compare the host name to example.com and
# redirect to www.example.com if it matches
<If "%{HTTP_HOST} == 'example.com'">
    Redirect permanent / http://www.example.com/
<ElseIf "%{HTTP_HOST} == ‘foobarfoo.com'">
    Redirect permanent / http://www2.example.com/
</If>
```

Configuration - Runtime

- Simple config-file variables: <Define>



```
<IfDefine TEST>
  Define servername test.example.com
</IfDefine>
<IfDefine !TEST>
  Define servername www.example.com
  Define SSL
</IfDefine>

DocumentRoot /var/www/${servername}/htdocs
```

Configuration - Runtime

- Finer control of timeouts, esp. during requests
  - mod_reqtimeout
  - KeepAliveTimout down to the millisecond
- Finer control over logging
  - per module/per directory
  - new logging levels (TRACE[1-8])

```
LogLevel info ssl:warn
<Directory "/usr/local/apache/htdocs/foo">
    LogLevel debug
</Directory>
```

Configuration - Runtime

- Other stuff:
  - No more NameVirtualHost
  - General purpose expression parser (BNF compatible)
  - AllowOverrideList

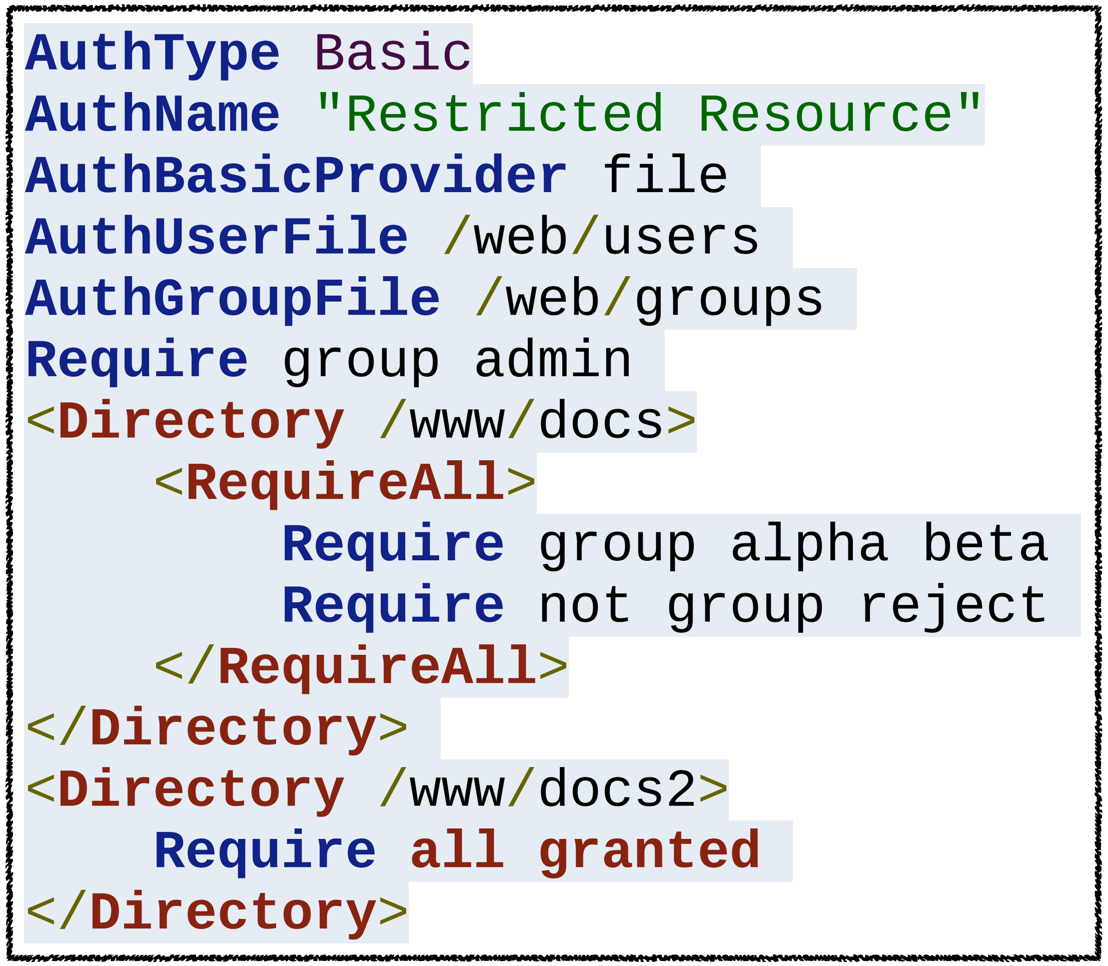
```
AllowOverride None
AllowOverrideList Redirect RedirectMatch Header
```

  - Loadable MPM modules
    - Recall that different MPMs have different config directives!

```
./configure —enable-mpms-shared=all
LoadModule mpm_event_module modules/mod_mpm_event.so
```

Configuration - Runtime

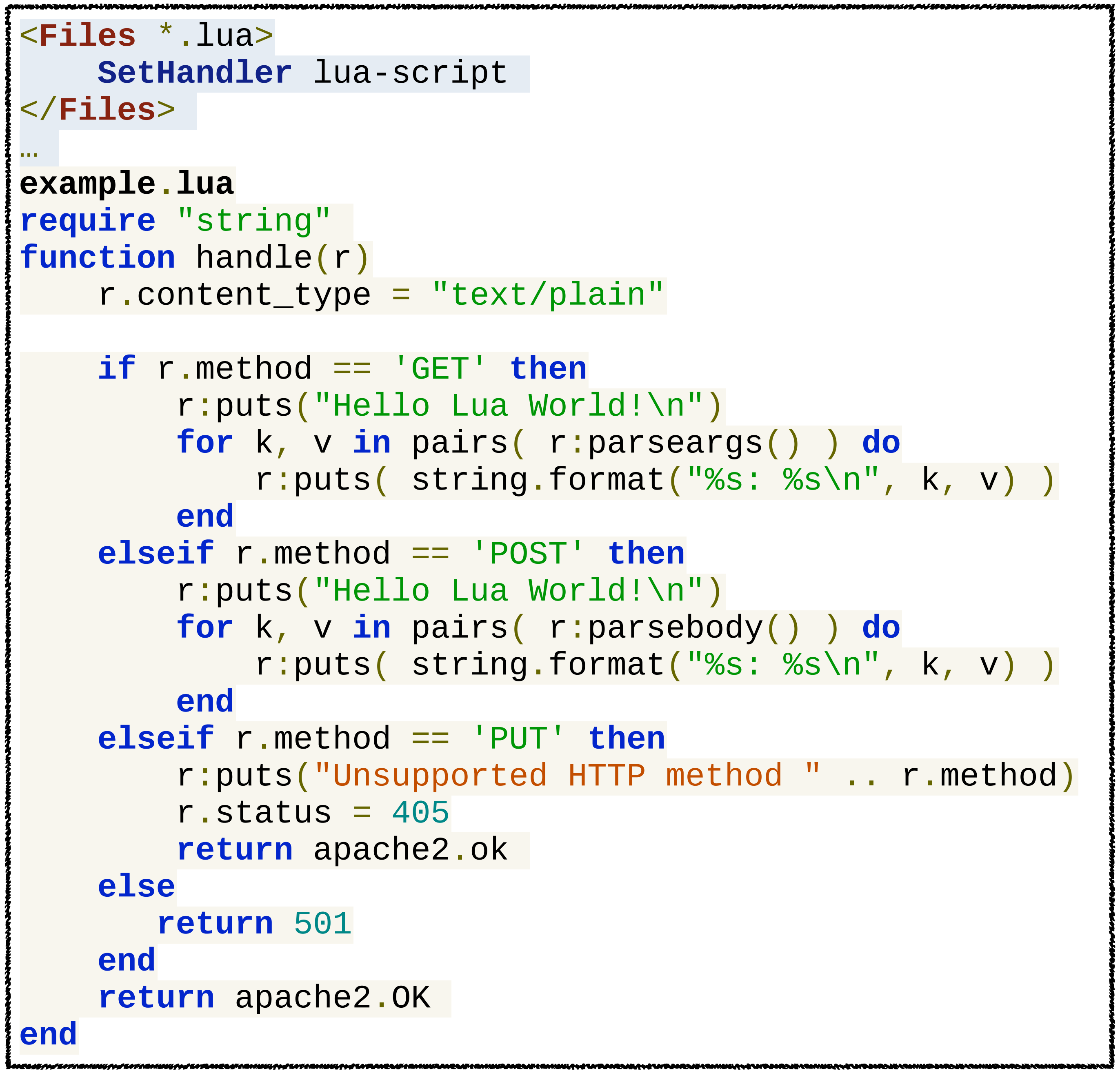
- Require
  - Removes order deny/allow insanity!
  - `mod_access_compat` for backwards combat



```
AuthType Basic
AuthName "Restricted Resource"
AuthBasicProvider file
AuthUserFile /web/users
AuthGroupFile /web/groups
Require group admin
<Directory /www/docs>
    <RequireAll>
        Require group alpha beta
        Require not group reject
    </RequireAll>
</Directory>
<Directory /www/docs2>
    Require all granted
</Directory>
```

New Modules

- ➡ mod_lua



```
<Files *.lua>
    SetHandler lua-script
</Files>
…
example.lua
require "string"
function handle(r)
    r.content_type = "text/plain"

    if r.method == 'GET' then
        r:puts("Hello Lua World!\n")
        for k, v in pairs( r:parseargs() ) do
            r:puts( string.format("%s: %s\n", k, v) )
        end
    elseif r.method == 'POST' then
        r:puts("Hello Lua World!\n")
        for k, v in pairs( r:parsebody() ) do
            r:puts( string.format("%s: %s\n", k, v) )
        end
    elseif r.method == 'PUT' then
        r:puts("Unsupported HTTP method " .. r.method)
        r.status = 405
        return apache2.ok
    else
       return 501
    end
    return apache2.OK
end
```

New Modules

- mod_buffer
  - buffer the i/o stacks w/i httpd
- mod_sed
  - True sed functionality, alternate to mod_substitute

```
<Directory "/var/www/docs/status">
    AddOutputFilter Sed html
    OutputSed "s/complete/DONE/g"
    OutputSed “s/in-progress/TODO/g"
</Directory>
```

- mod_remoteip
  - allow access to the *real* client IP address

```
RemoteIPHeader X-Client-IP
```

  - Also provides HA PROXY support

APACHE ROADSHOW Berlin, June 11-14, 2018

New Modules

- mod_session
  - easily maintain application server state
- mod_auth_form
  - Form-based auth can now be handled internally

```
<Location /dologin.html>
    SetHandler form-login-handler
    AuthFormLoginRequiredLocation http://example.com/login.html
    AuthFormLoginSuccessLocation http://example.com/success.html
    AuthFormProvider file
    AuthUserFile conf/passwd
    AuthType form
    AuthName realm
    Session On
    SessionCookieName session path=/
    SessionCryptoPassphrase secret
</Location>
```

New Modules

- mod_log_debug
  - Add debug logging at any hook

```
<Location /foo>
  LogMessage “subreq to foo” hook=type_checker expr=%{IS_SUBREQ}
</Location>
```

- mod_ratelimit
  - (basic) bandwidth limiting for clients

```
<Location /downloads>
    SetOutputFilter RATE_LIMIT
    SetEnv rate-limit 400
</Location>
```

Even more!

- mod_cache
  - Can serve stale data if required
  - `X-Cache-Header` now supports `HIT/MISS/REVALIDATE`
  - Can cache `HEAD`
  - `htcacheclean` improvements
  - *Redis* and *memcached* (And *Apache Geode*)
- mod_socache / mod_slotmem
  - Data object/blog storage mechanisms
- mod_brotli

New Modules

- mod_proxy submodules:
  - mod_proxy_fcgi
  - mod_proxy_scgi
  - mod_proxy_uwsgi
  - mod_proxy_wstunnel
  - mod_proxy_html
  - mod_proxy_express
  - mod_proxy_hcheck

Cloud and Performance

- The Cloud is a game changer for web servers
  - Horizontal scalability is no longer as painful
  - Concurrency is no longer the sole consideration
  - ... or maybe even the primary one
  - What's important now? Transaction Time! (because it *CAN* be)
    - Low latency
    - Fast req/resp turnover
  - Does density still matter? *Of course!*
  - *micro-services*
  - Are there environs where *super-mega* concurrency is the bugaboo? *You betcha!* *(but the cloud makes these more and more rare, and you're likely using a bad architecture anyway)*

Cloud and Dynamics

- The Cloud is a game changer for web servers
  - The cloud is a dynamic place
  - automated reconfiguration
  - horizontal, not vertical scaling
  - self-aware environments

OK, maybe not THAT self-aware

Why Dynamic Proxy Matters

- Apache httpd still the most frequently used front-end
- Proxy capabilities must be cloud friendly
- Front-end must be dynamic friendly

Apache httpd 2.4 proxy

- Reverse Proxy Improvements
  - Supports FastCGI, SCGI, Websockets in balancer
  - Additional load balancing mechanisms
  - Runtime changing of clusters w/o restarts
  - Support for dynamic configuration
  - mod_proxy_express
  - mod_fcgid and fcgistarter
  - Support for Unix Domain Sockets

Backend Status

- Dynamic Health Checks !
  - TCP/IP Ping
  - *OPTIONS*
  - *HEAD*
  - *GET*

```
ProxyHCExpr ok234 {%{REQUEST_STATUS} =~ /^[234]/}
ProxyHCExpr gdown {%{REQUEST_STATUS} =~ /^[5]/}
ProxyHCExpr in_maint {hc('body') !~ /Under maintenance/}

<Proxy balancer://foo/>
  BalancerMember http://www.example.com/  hcmethod=GET hcexpr=in_maint hcuri=/status.php
  BalancerMember http://www2.example.com/  hcmethod=HEAD hcexpr=ok234 hcinterval=10
  BalancerMember http://www3.example.com/ hcmethod=TCP hcinterval=5 hcpasses=2 hcfails=3
  BalancerMember http://www4.example.com/
</Proxy>

ProxyPass "/" “balancer://foo/"
ProxyPassReverse "/" “balancer://foo/"
```

Mass Reverse Proxy

- Use the new mod_proxy_express module
  - ProxyPass mapping obtained via db file
  - Fast and efficient
  - Still dynamic, with no config changes required
  - micro-services? You betcha!

ProxyExpress map file

```
##
##express-map.db:
##

www1.example.com     http://192.168.002.2:8080
www2.example.com     http://192.168.002.12:8088
www3.example.com     http://192.168.002.10
 ...
www6341.example.com  http://192.168.211.26
```

httpd.conf file

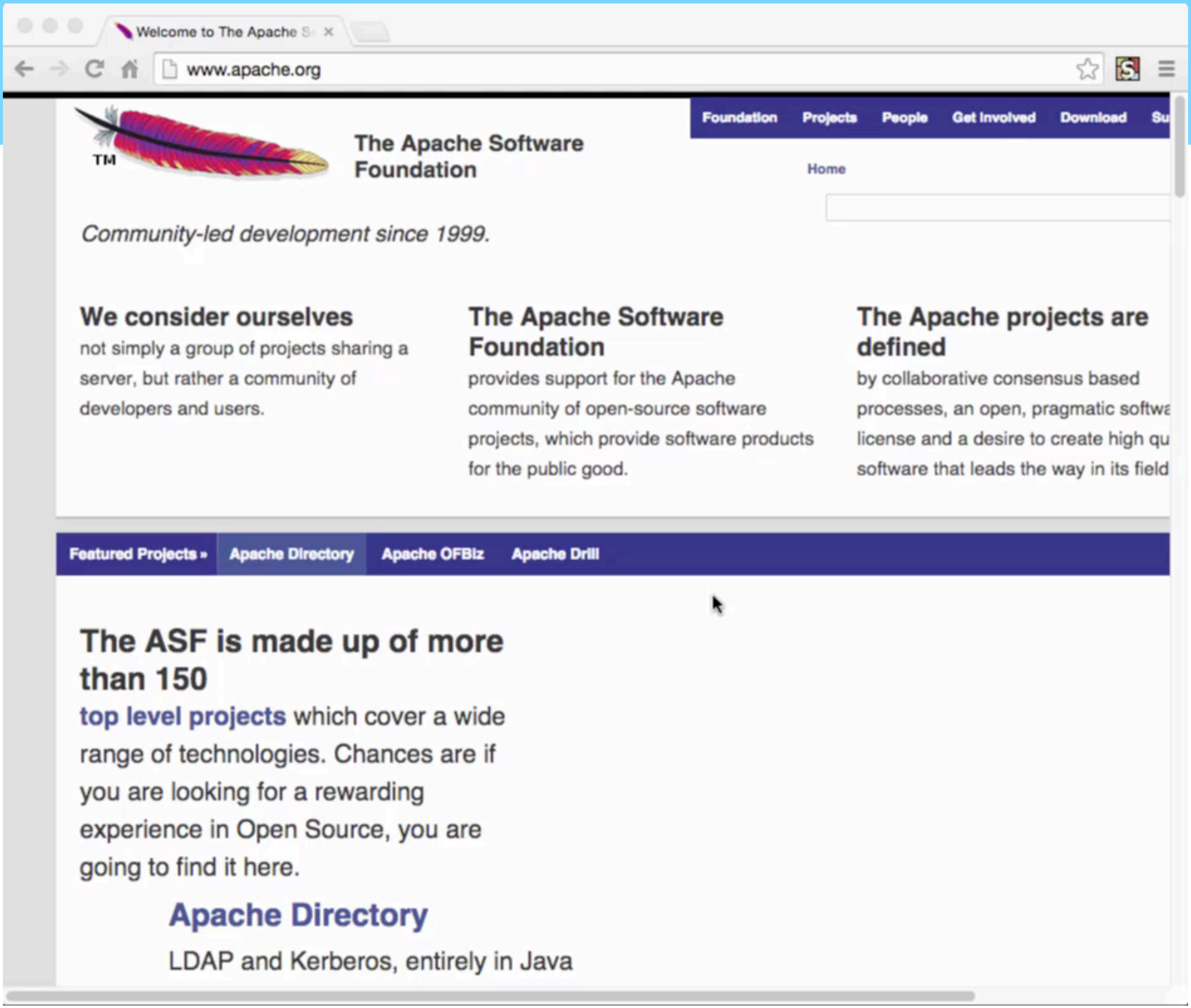
```
ProxyExpressEnable On
ProxyExpressDBMFile express-map.db
```

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
  Require 192.168.2.22
</Location>
```



The Apache Software Foundation

Community-led development since 1999.

We consider ourselves
not simply a group of projects sharing a server, but rather a community of developers and users.

The Apache Software Foundation
provides support for the Apache community of open-source software projects, which provide software products for the public good.

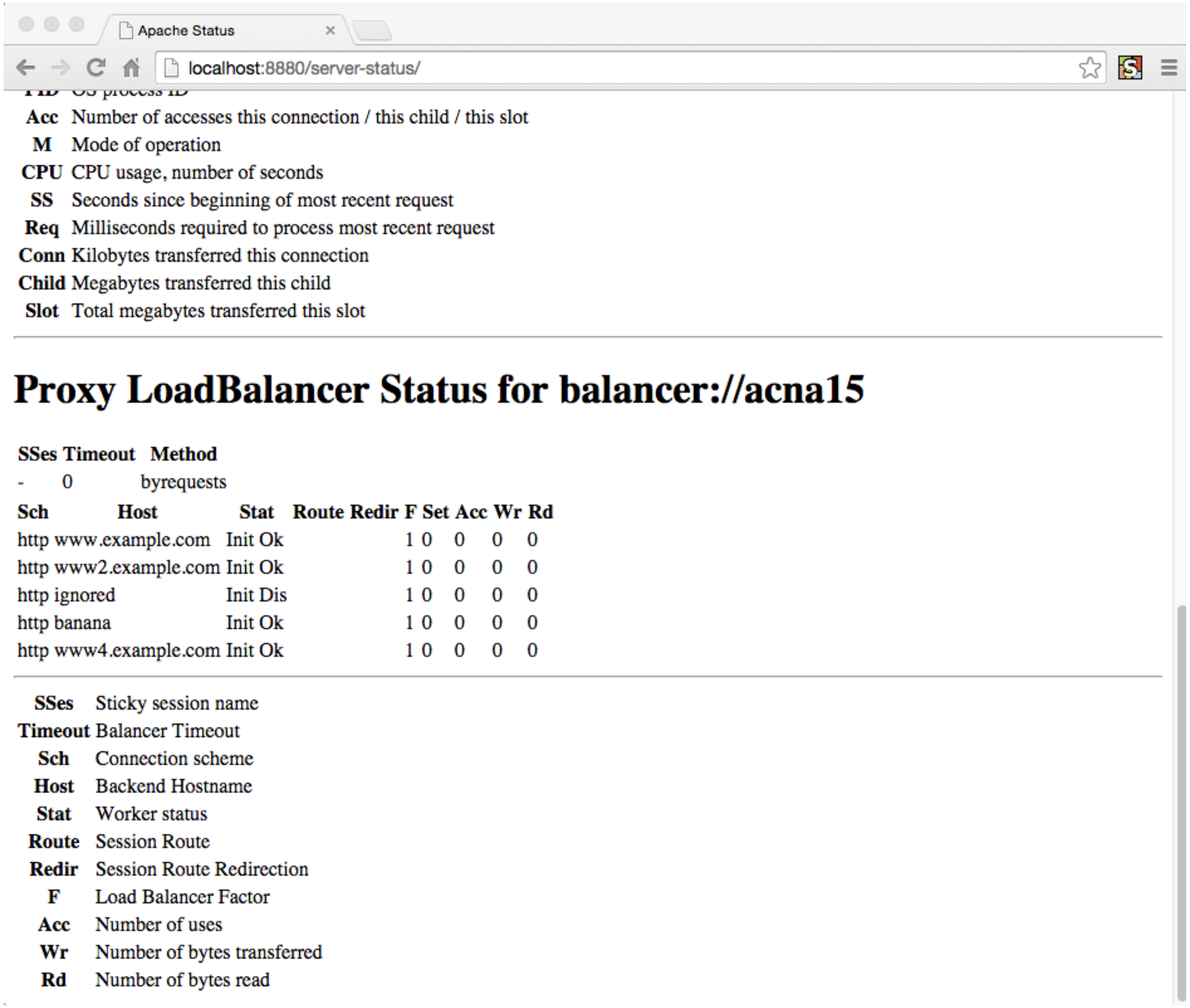
The Apache projects are defined
by collaborative consensus based processes, an open, pragmatic softwa license and a desire to create high qu software that leads the way in its field

The ASF is made up of more than 150
top level projects which cover a wide range of technologies. Chances are if you are looking for a rewarding experience in Open Source, you are going to find it here.

Apache Directory

LDAP and Kerberos, entirely in Java

server-status aware



PID	OS process ID
Acc	Number of accesses this connection / this child / this slot
M	Mode of operation
CPU	CPU usage, number of seconds
SS	Seconds since beginning of most recent request
Req	Milliseconds required to process most recent request
Conn	Kilobytes transferred this connection
Child	Megabytes transferred this child
Slot	Total megabytes transferred this slot

Proxy LoadBalancer Status for balancer://acna15

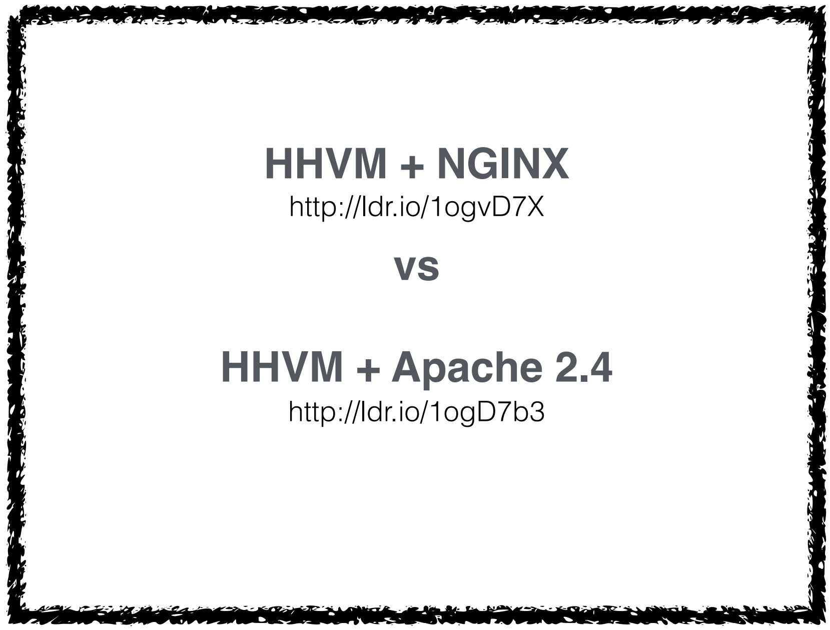
SSes	Timeout	Method
-	0	byrequests

Sch	Host	Stat	Route	Redir	F	Set	Acc	Wr	Rd
http	www.example.com	Init Ok			1	0	0	0	0
http	www2.example.com	Init Ok			1	0	0	0	0
http	ignored	Init Dis			1	0	0	0	0
http	banana	Init Ok			1	0	0	0	0
http	www4.example.com	Init Ok			1	0	0	0	0

SSes	Sticky session name
Timeout	Balancer Timeout
Sch	Connection scheme
Host	Backend Hostname
Stat	Worker status
Route	Session Route
Redir	Session Route Redirection
F	Load Balancer Factor
Acc	Number of uses
Wr	Number of bytes transferred
Rd	Number of bytes read

Performance

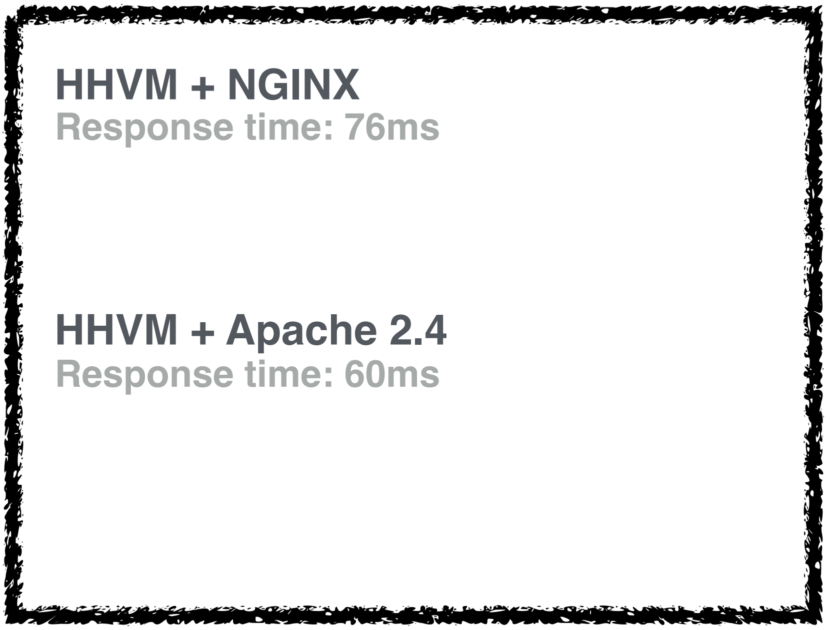
- From Nic Rosenthal Battle of the stacks (http://www.slideshare.net/AllThingsOpen/battle-of-the-stacks)



HHVM + NGINX
http://ldr.io/1ogvD7X

vs

HHVM + Apache 2.4
http://ldr.io/1ogD7b3



HHVM + NGINX
Response time: 76ms

HHVM + Apache 2.4
Response time: 60ms

HHVM + Apache 2.4

Champion of the Battle Of The Stacks ATO Edition

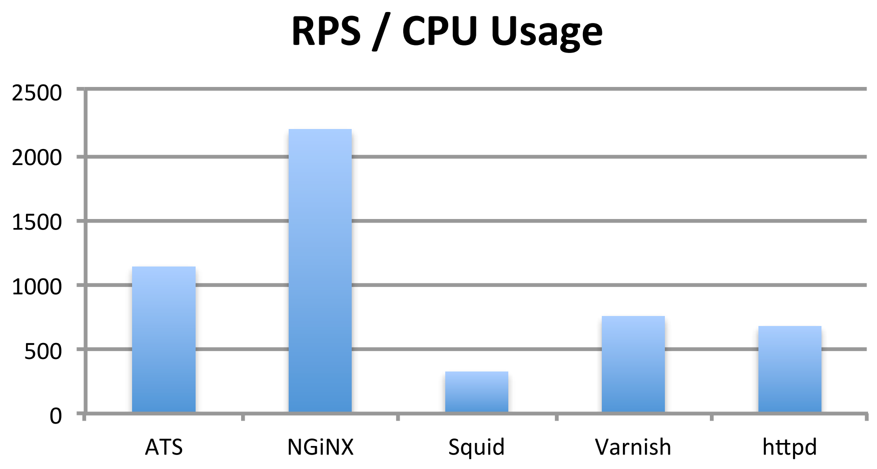
Image by Articularnos.com https://www.flickr.com/photos/articularnos/

@jimjag

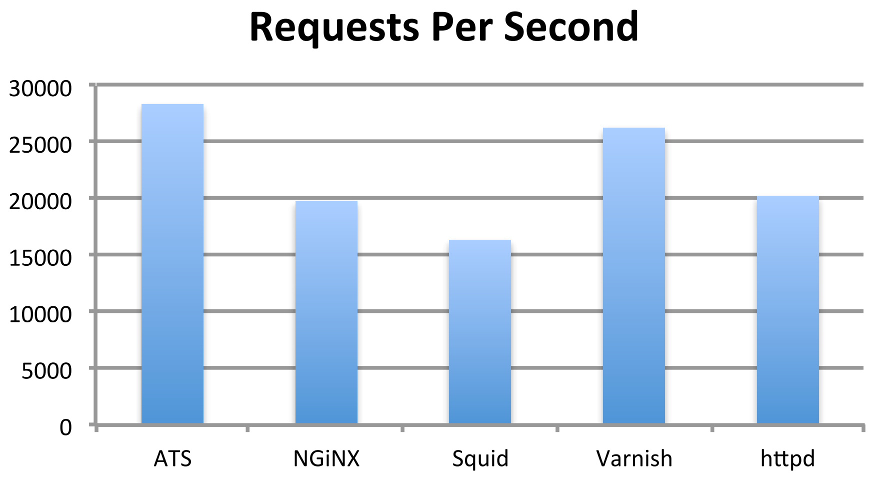
Performance

➡ **From Bryan Call's 2014 ApacheCon preso (http://www.slideshare.net/bryan_call/choosing-a-proxy-server-apachecon-2014)**

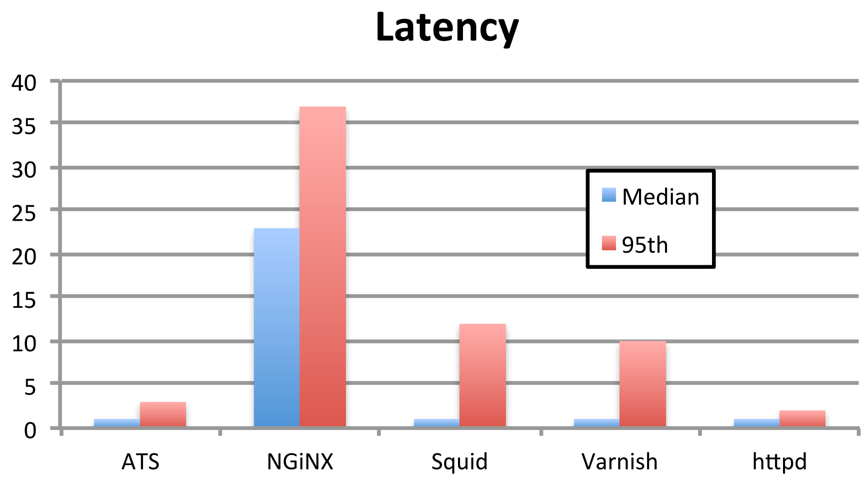
- Squid used the most CPU again
- NGiNX had latency issues
- ATS most throughput



RPS / CPU Usage



Requests Per Second



Latency

APACHE ROADSHOW Berlin, June 11-14, 2018

@jimjag

Raw Performance

- Event MPM : no longer experimental
  - non-blocking
  - async
- Faster, more efficient APR
- Smaller memory footprint
- More efficient data structures (worker and event)

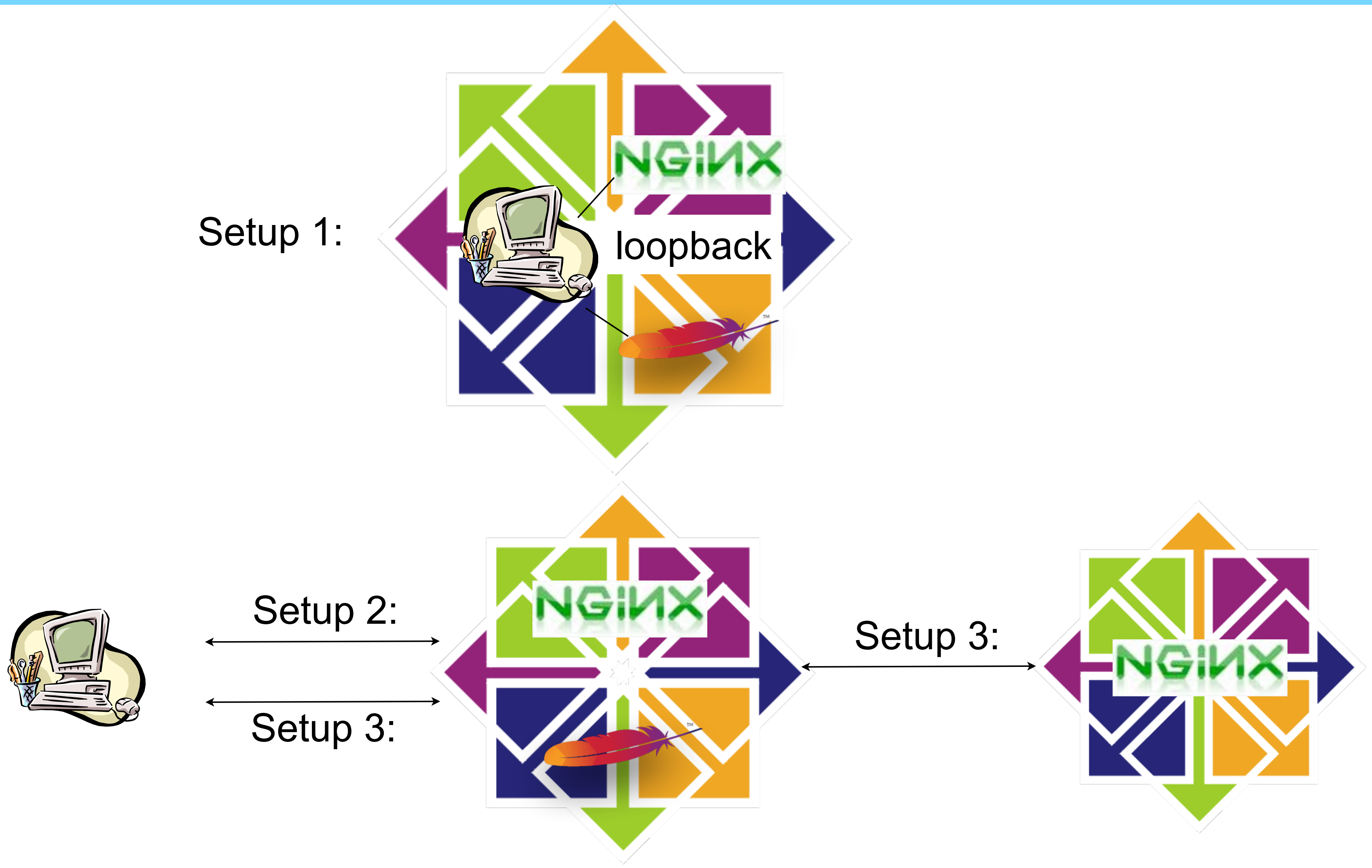
Apache httpd vs nginx

- Why nginx? Everyone asks about it...
- Benchmark: local and reverse proxy transaction times
  - Apache httpd 2.4.22-dev, nginx 1.8.1
  - CentOS6, Dual Xeon 3.33GHz
  - 4GB memory
  - localhost loopback and external (no firewall)
  - Double checked results: OSX 10.11.2 (8-core), Fedora 23 (4-core)

APACHE ROADSHOW
Berlin, June 11-14, 2018

@jimjag

Setup



Setup 1:

loopback

Setup 2:

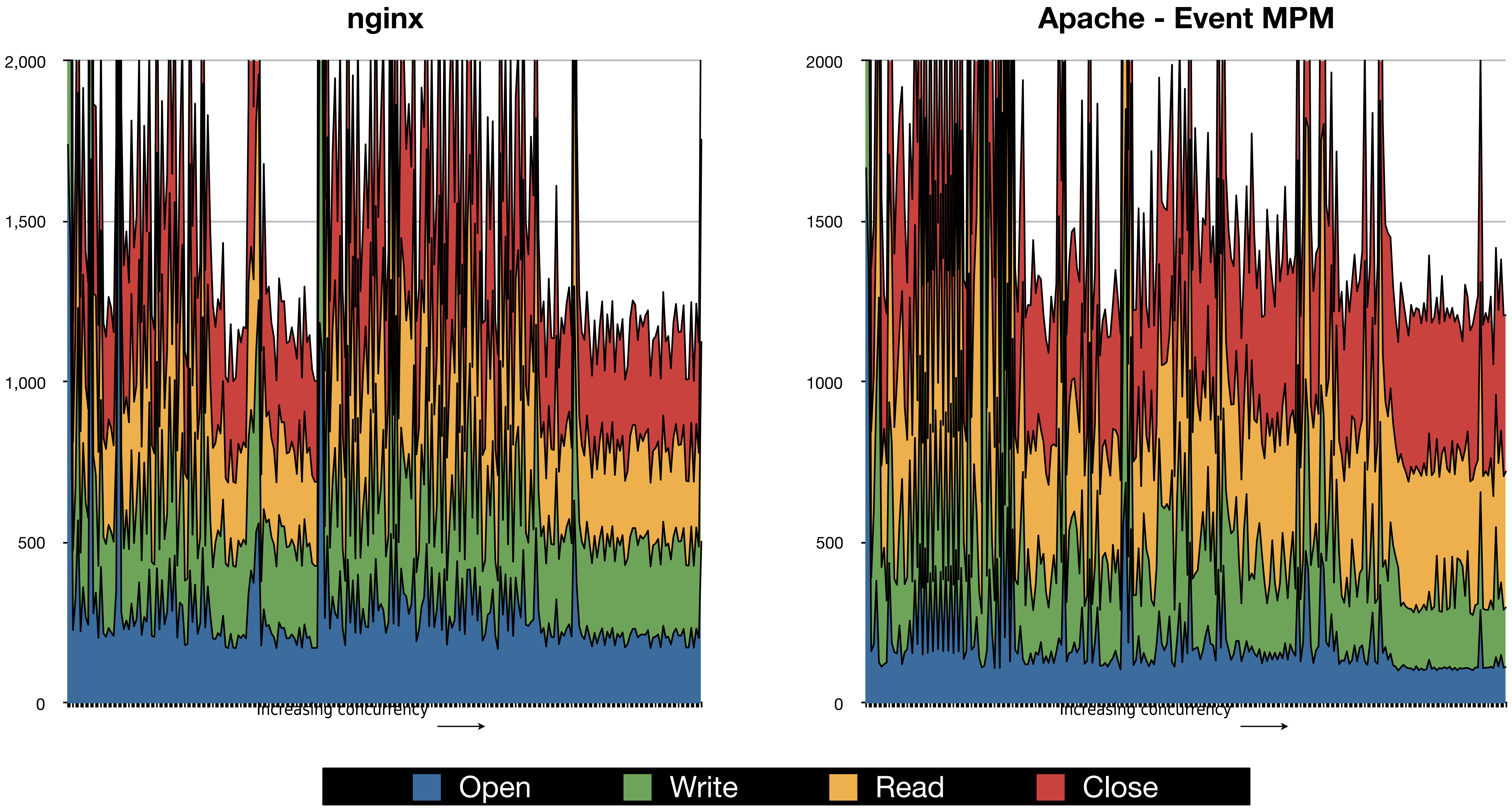
Setup 3:

Setup 3:

Considerations

- Multiple benchmarking systems:
  - flood (50/250/5/2, 50/100/5/2, 50/5/5/2)
  - httperf (num-conns=100->20000, numcalls=3,10,100)
  - weighttp
- Full URL requests (www.example.com/index.html)
- Static local requests
- Static reverse proxy requests
- All Apache httpd MPMs
- No significant “tuning” efforts (mostly out of the box configs)

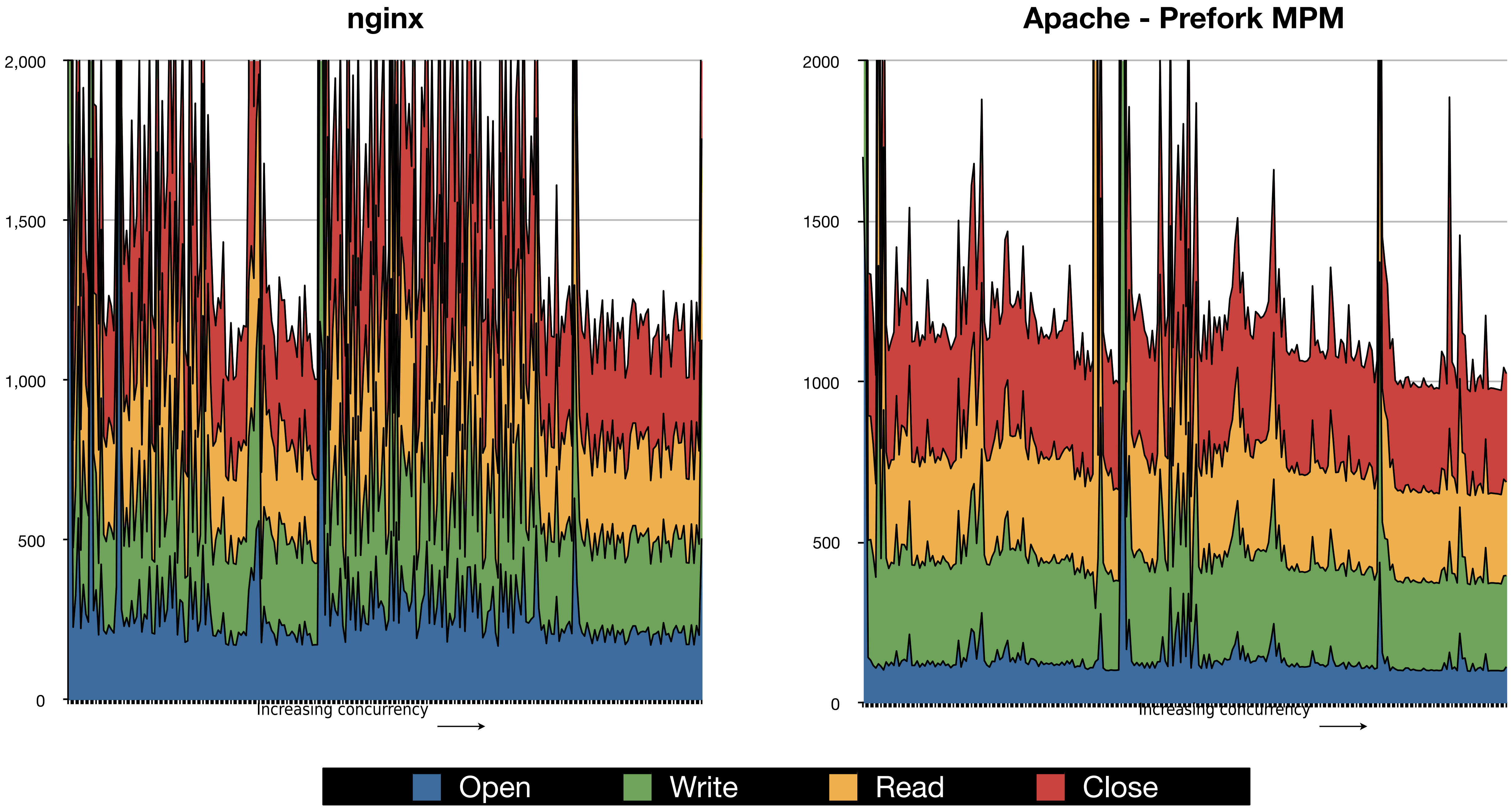
nginx vs Event (typical)



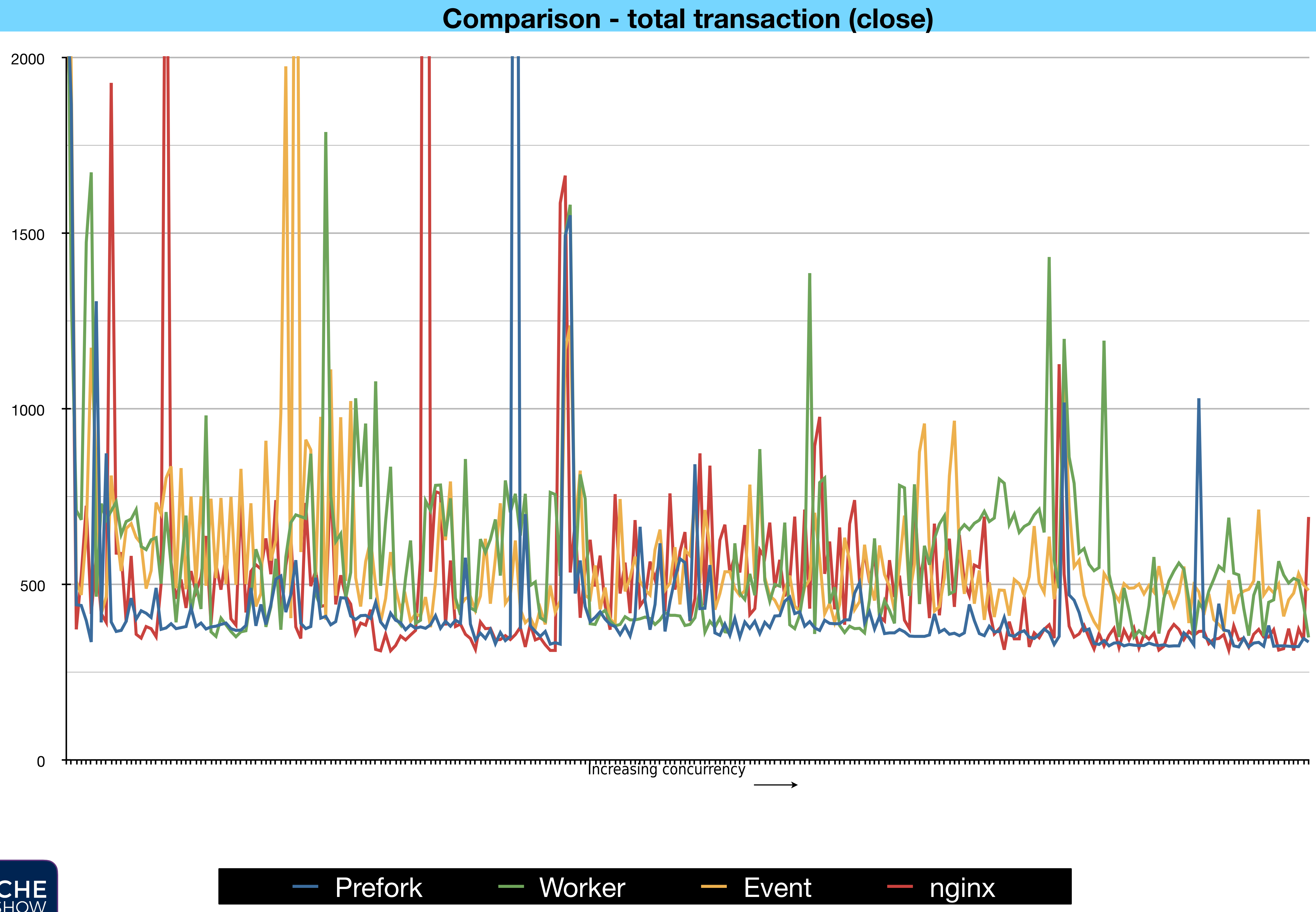
APACHE ROADSHOW
Berlin, June 11-14, 2018

@jimjag

nginx vs Prefork (typical)

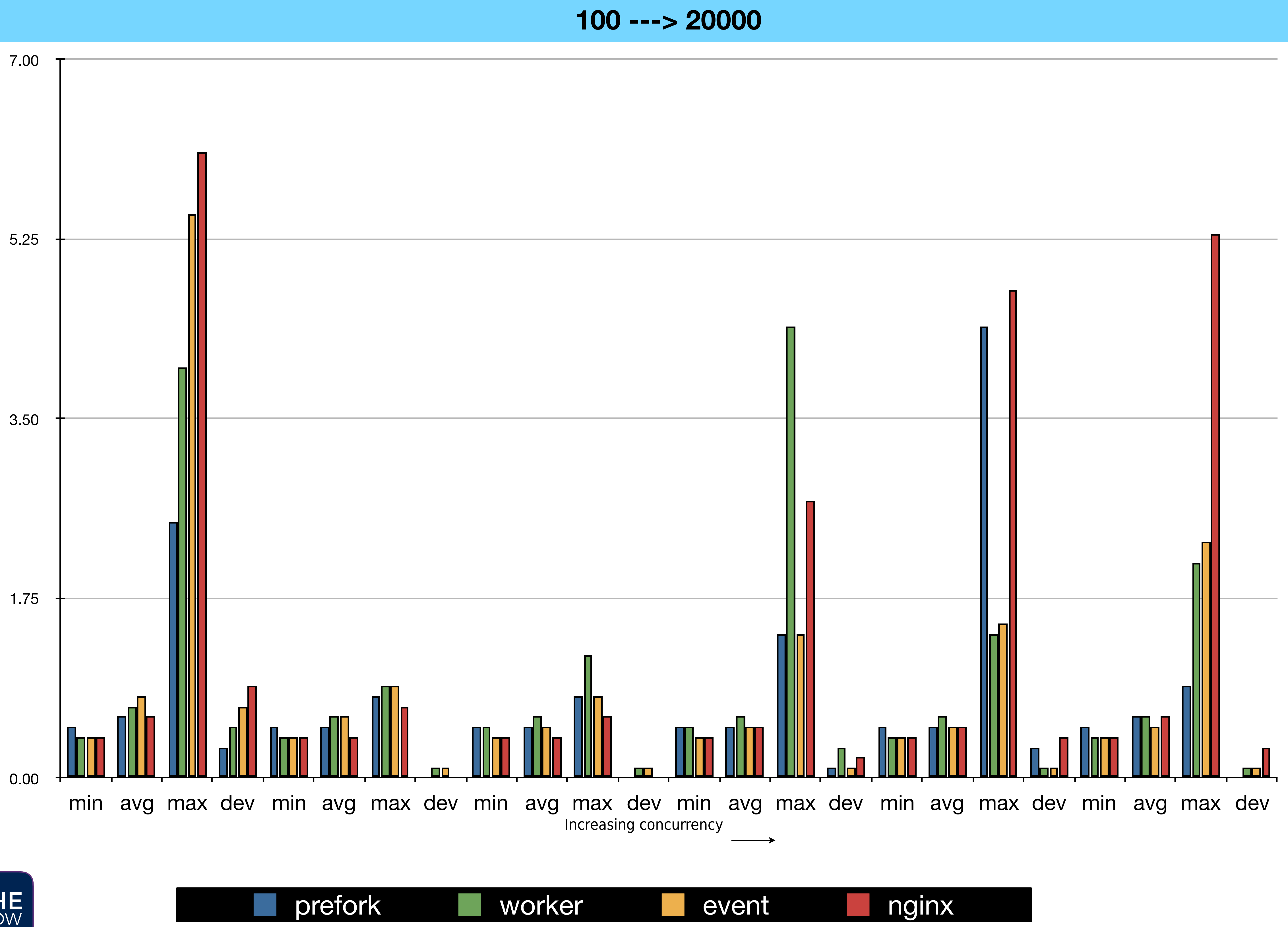


Total req/resp time



@jimjag

Resp to Req. Bursts - httperf



@jimjag

Independent benchmarks

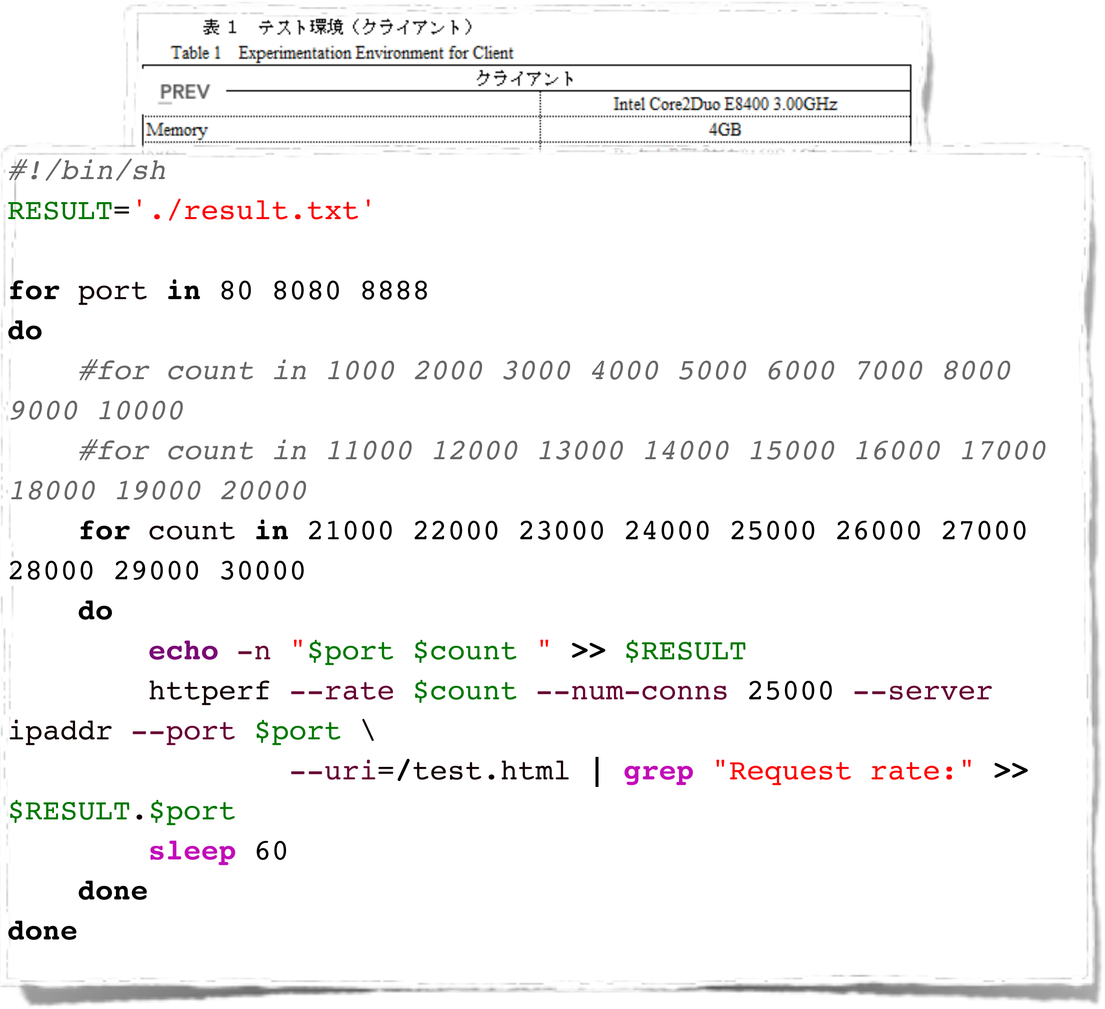


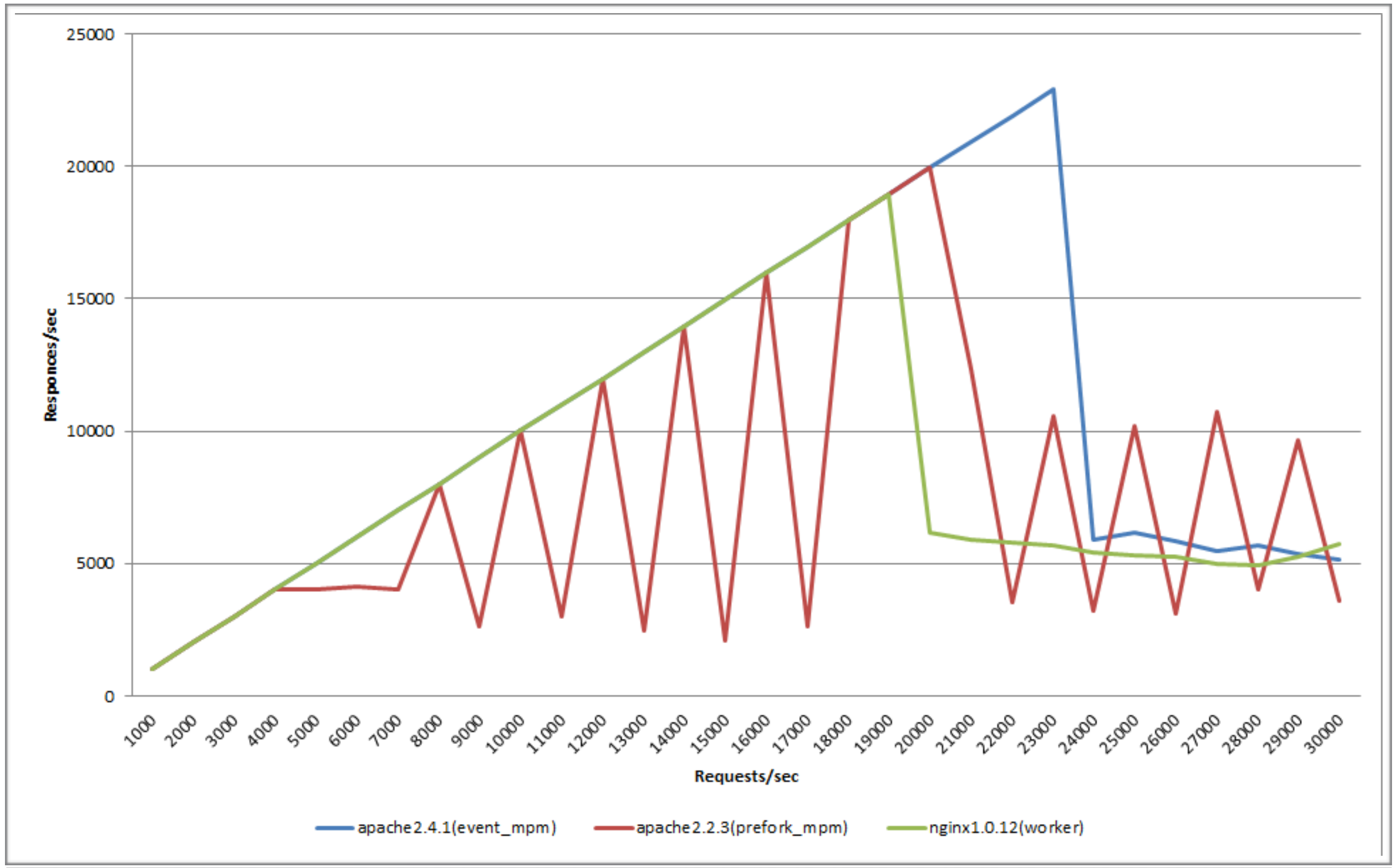
表 1　テスト環境（クライアント）

Table 1　Experimentation Environment for Client

	クライアント
PREV	Intel Core2Duo E8400 3.00GHz
Memory	4GB

```
#!/bin/sh
RESULT='./result.txt'

for port in 80 8080 8888
do
    #for count in 1000 2000 3000 4000 5000 6000 7000 8000
9000 10000
    #for count in 11000 12000 13000 14000 15000 16000 17000
18000 19000 20000
    for count in 21000 22000 23000 24000 25000 26000 27000
28000 29000 30000
    do
        echo -n "$port $count " >> $RESULT
        httperf --rate $count --num-conns 25000 --server
ipaddr --port $port \
                --uri=/test.html | grep "Request rate:" >>
$RESULT.$port
        sleep 60
    done
done
```

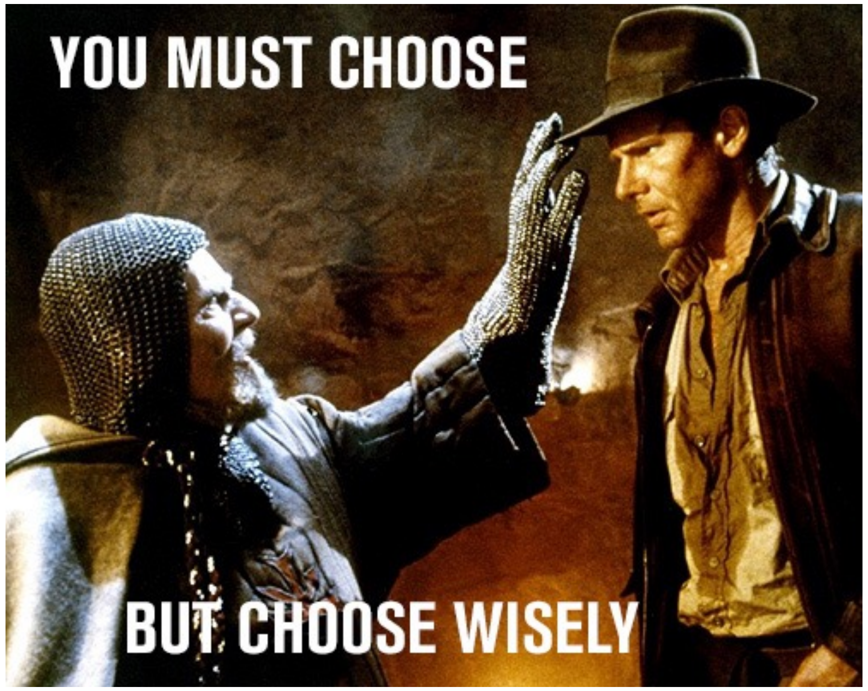


Source: Ryosuke Matsumoto : http://blog.matsumoto-r.jp/?p=1812

APACHE ROADSHOW
Berlin, June 11-14, 2018

Take-away

- Today, the web-server isn’t the slow link in the chain.
- Benchmarks get stale… fast!
- Real world trumps test environs
- Choose the right tool for the right job



HTTP/2

- Implements RFC 7540
- Supports both h2 (HTTP/2 over TLS) and h2c (HTTP/2 over TCP[cleartext])
- Enterprise-ready regarding stability, performance, etc.
- Also supported in mod_proxy
- Perfect compliance

Thanks

Twitter: @jimjag

Emails:

jim@jaguNET.com
jim@apache.org
jimjag@gmail.com

http://www.slideshare.net/jimjag/

APACHE ROADSHOW
Berlin, June 11-14, 2018