JBoss WORLD
CHICAGO 2009

FOLLOW US:

TWITTER.COM/REDHATSUMMIT

TWEET ABOUT US:

ADD #SUMMIT AND/OR #JBOSSWORLD TO THE END OF YOUR EVENT-RELATED TWEET

presented by

JBoss WORLD CHICAGO 2009

Large clusters in JBoss

Bela Ban
Red Hat
Sept 2, 2009

presented by

Agenda

Scenario:

- Clustered web applications
- Apache httpd / mod-jk / mod-cluster
- Large cluster of JBoss application servers
- Large number of clients
- Big variance in the number of clients

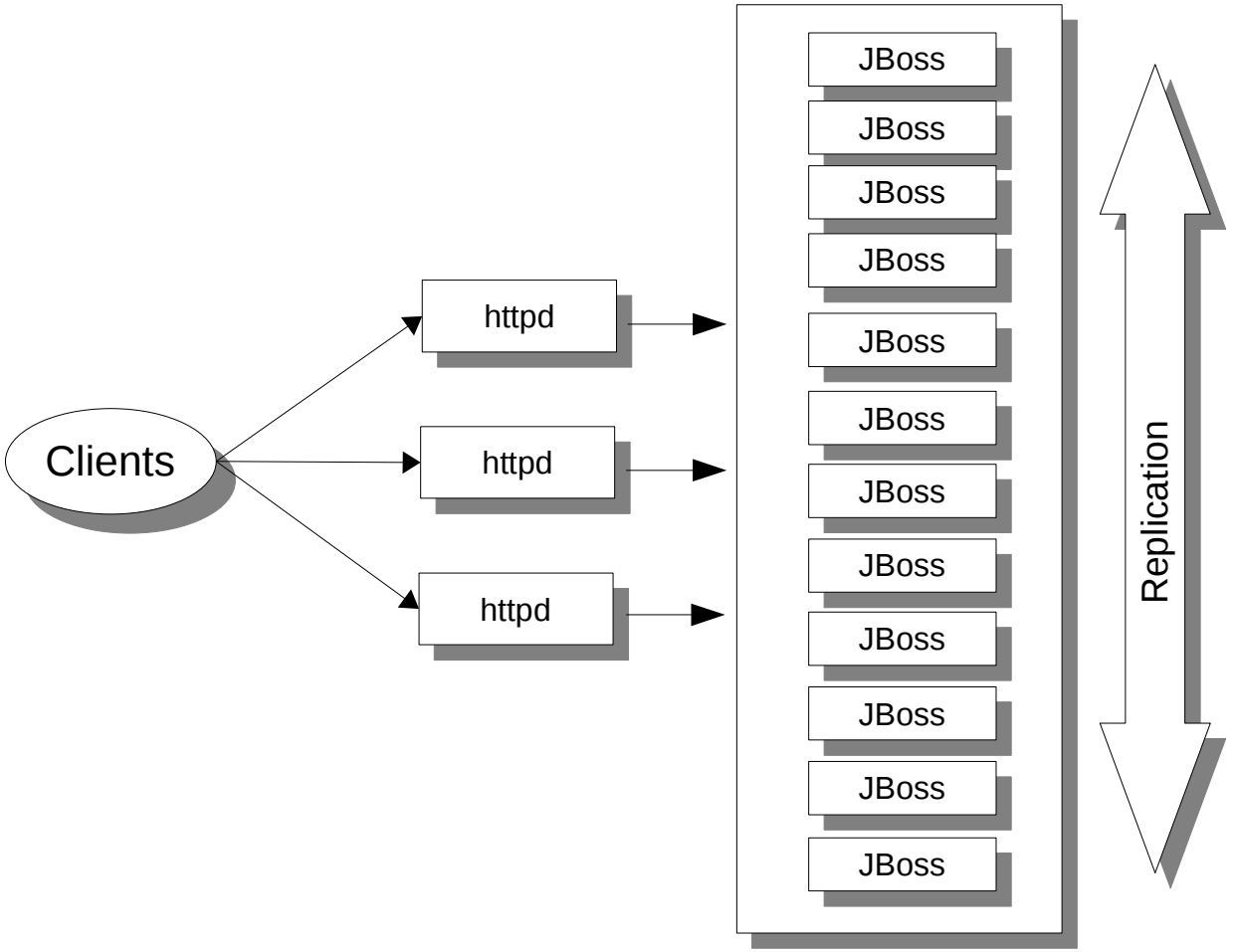
Issues

- Rolling upgrades and static configuration

mod-cluster

Demo (mod-cluster domains in a cloud)

Typical mod_jk based setup



mod_jk configuration (httpd side)

worker.properties:

```
worker.jboss1.host=jboss1
worker.jboss2.host=jboss2
worker.jboss3.host=jboss3
…
worker.loadbalancer.type=lb
worker.loadbalancer.balance_workers=jboss1,jboss2,jboss3
```

uriworkermap.properties:

```
/jmx-console/*=loadbalancer
/web-console/*=loadbalancer
/mywebapp/*=loadbalancer
/mynewapp/*=loadbalancer
```

Issues

- #1 Large flat cluster
  - Rolling upgrades are impossible with (binary) incompatible upgrades
    - New software (JBoss, 3rd party, application)
  - Failures (e.g. network partitions) affect many nodes
- #2 Static configuration
  - workers.properties and uriworkermap.properties have to be modified on all 3 httpd servers when
    - a host is added or removed
    - a web app is deployed / undeployed
  - Not good in a cloud environment where hosts are dynamically added or removed

Problem #1: large flat clusters

Most of the down time is caused by upgrades, not crashes

If a new release is binary incompatible, the restarted node won't be able to participate in the cluster

Incompatibilities are caused by

- New JBoss version, e.g. 4.2 → 4.3
- Individual component upgrades (JGroups, JBossCache)
- DB schema changes
- JDK upgrades (serialVersionUID)
- Application incompatibilities

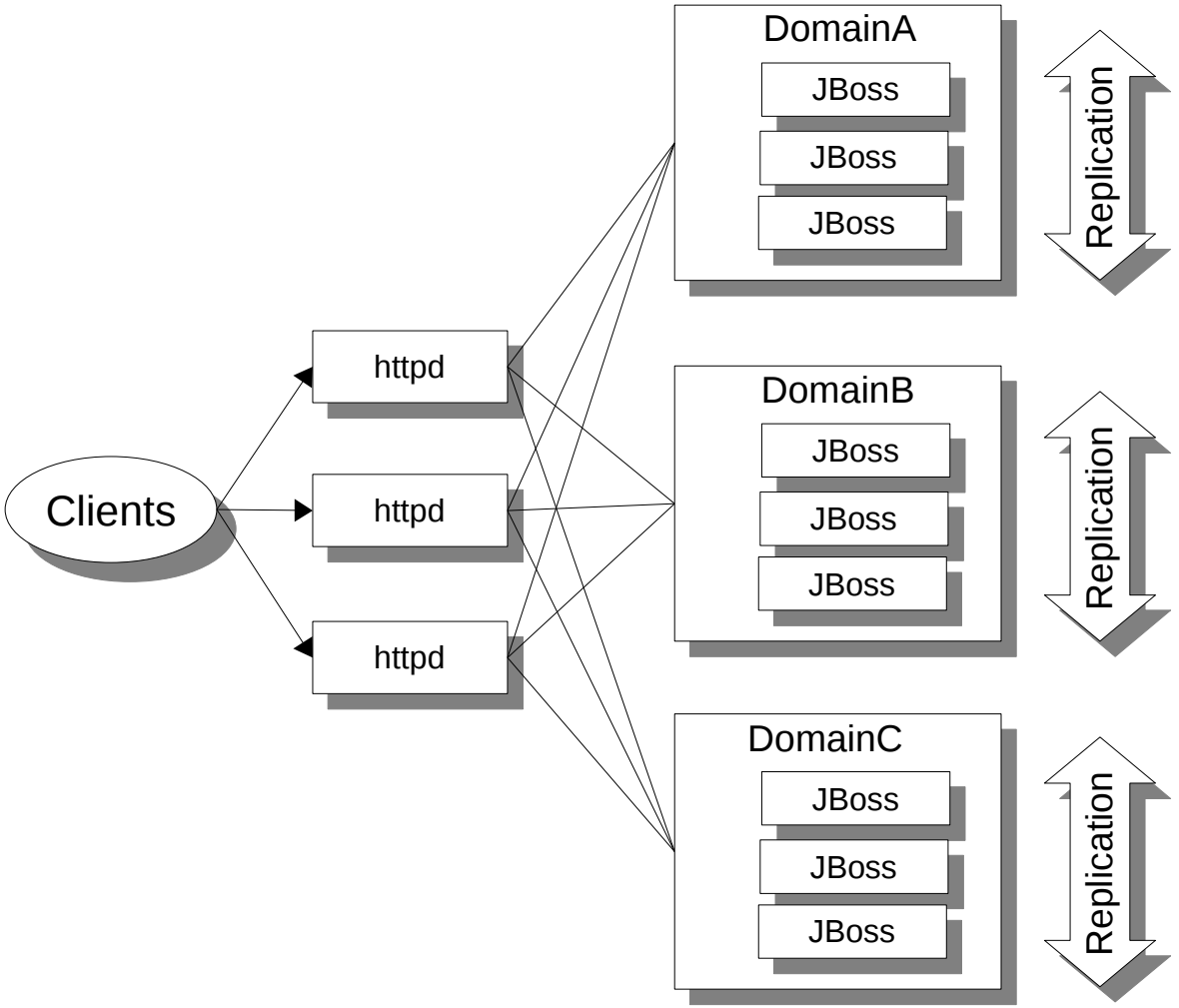
Issues with large flat clusters cont'd

- A node or switch crash affects more nodes
  - State transfer, rebalancing of state (Infinispan's DIST mode), merge handling
  - RPC's across the cluster might block until a node is marked as crashed
- State
  - We cannot use total replication for scalability reasons
- Communication
  - TCP becomes an issue as a message is sent N-1 times

Solution: mod-jk domains

- Instead of a large cluster of 1000 nodes, we create 10 *domains* of 100 nodes each
- httpd creates a new session S on a random node N in a random domain D
  - All requests for S go to N in D (if sticky sessions are configured)
  - If N is shut down or crashes, S is directed to another node *within* D
  - Changes to S are replicated only *within* D
  - If there are no more nodes in D, S is lost
    - No replication between domains
    - A domain == a JBoss cluster

mod-jk domains



Rolling upgrades with domains

An *entire domain* is upgraded, not an individual node

Steps

- *Disable* entire domain (mod-jk /status app on httpd)
- httpd won't create new sessions on disabled workers
- Requests for existing sessions will still be forwarded to disabled workers
- When all sessions on of a given domain have expired we can shut down all workers, upgrade and restart
- httpd can now create new sessions in the upgraded domain
  - Different domains can have different software versions
  - DB schema still an issue

mod_jk domain configuration (httpd side)

worker.properties:

```
worker.jboss1.host=jboss1
worker.jboss1.domain=A
worker.jboss2.host=jboss2
worker.jboss2.domain=A
worker.jboss3.host=jboss3
worker.jboss3.domain=B
worker.jboss4.host=jboss4
worker.jboss4.domain=B
```

Make sure you separate cluster traffic for A and B !

```
jboss1: ./run.sh -c all -g A -u 232.1.1.1 -m 7500 -Djboss.jvmRoute=jboss1
jboss2: ./run.sh -c all -g A -u 232.1.1.1 -m 7500 -Djboss.jvmRoute=jboss2
jboss3: ./run.sh -c all -g B -u 232.2.2.2 -m 8500 -Djboss.jvmRoute=jboss3
jboss4: ./run.sh -c all -g B -u 232.2.2.2 -m 8500 -Djboss.jvmRoute=jboss4
```

Problem #2: static configuration

When a worker is added or removed, we need to modify workers.properties or uriworkermap.properties on all httpd servers

- Tedius for large clusters, or dynamically changing clouds

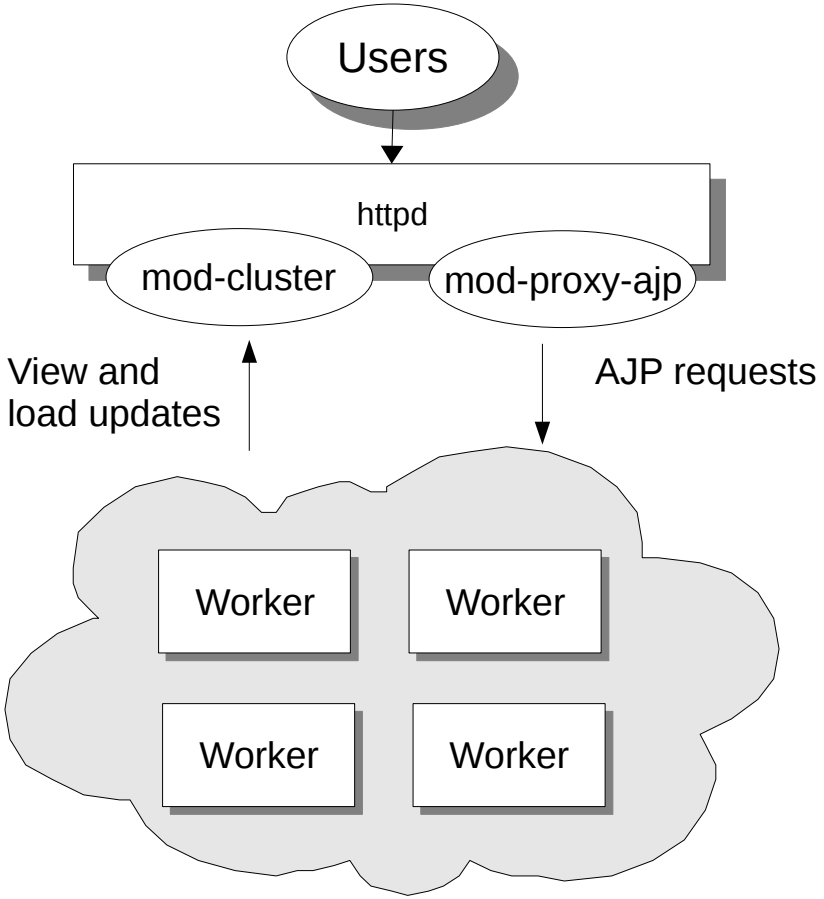
Enter **mod-cluster**

- (Almost) no configuration on the httpd side
- Works with httpd 2.2.8+, JBoss AS 5+, JBossWeb 2.2.1 and Tomcat 6
- Based on mod-proxy
- http://www.jboss.org/mod_cluster

mod-cluster

- Workers register *themselves* with httpd server(s)
  - No more workers.properties
  - Changes in cluster topology are also sent to httpd
- Deployed webapps are automatically registered with httpd, undeployed webapps are un-registered
  - No more uriworkermap.properties
  - No 404s for undeployed webapps
- Workers send their load factors to httpd, so httpd can forward requests based on actual load
  - Load computation is pluggable

mod-cluster architecture



mod_cluster configuration (httpd side)

httpd.conf:

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
LoadModule slotmem_module modules/mod_slotmem.so
LoadModule manager_module modules/mod_manager.so
LoadModule proxy_cluster_module modules/mod_proxy_cluster.so
LoadModule advertise_module modules/mod_advertise.so

<Location /mod_cluster_manager>
    SetHandler mod_cluster-manager

    Order deny,allow
    Deny from all
    Allow from 192.168.1. 127.0.0.1
</Location>
```

mod_cluster configuration (JBossAS)

JBOSS/server/all/deploy/jbossweb.sar/server.xml:

```
<Listener
  className="org.jboss.web.tomcat.service.deployers.MicrocontainerIntegrationLifecycleListener"
  delegateBeanName="HAModClusterService"/>
…

 <Engine name="jboss.web" defaultHost="localhost" jvmRoute="${jboss.jvmRoute}">
```

Copy mod_cluster.sar to JBOSS/server/all/deploy/

JBOSS/server/all/deploy/mod_cluster.sar/META-INF/:

```
 <bean name="HAModClusterConfig" class="org.jboss.modcluster.config.ha.HAModClusterConfig">

    <!-- Comma separated list of address:port listing the httpd servers where mod_cluster is running -->
    <property name="proxyList">${jboss.modcluster.proxyList:httpd-1:8000,httpd-2:8000}</property>
    <property name="domain">${jboss.Domain:DefaultDomain}</property>
    …
</bean>
```

Starting the JBossAS instances

```
jboss1: ./run.sh -c all -g A -u 232.1.1.1 -m 7500 -Djboss.Domain=A -Djboss.jvmRoute=jboss1
jboss2: ./run.sh -c all -g A -u 232.1.1.1 -m 7500 -Djboss.Domain=A -Djboss.jvmRoute=jboss2
jboss3: ./run.sh -c all -g B -u 232.2.2.2 -m 8500 -Djboss.Domain=B -Djboss.jvmRoute=jboss3
jboss4: ./run.sh -c all -g B -u 232.2.2.2 -m 8500 -Djboss.Domain=B -Djboss.jvmRoute=jboss4
```

We now have 2 clusters (A and B) which have nodes jboss1 and jboss2 (A) and jboss3 and jboss4 (B)

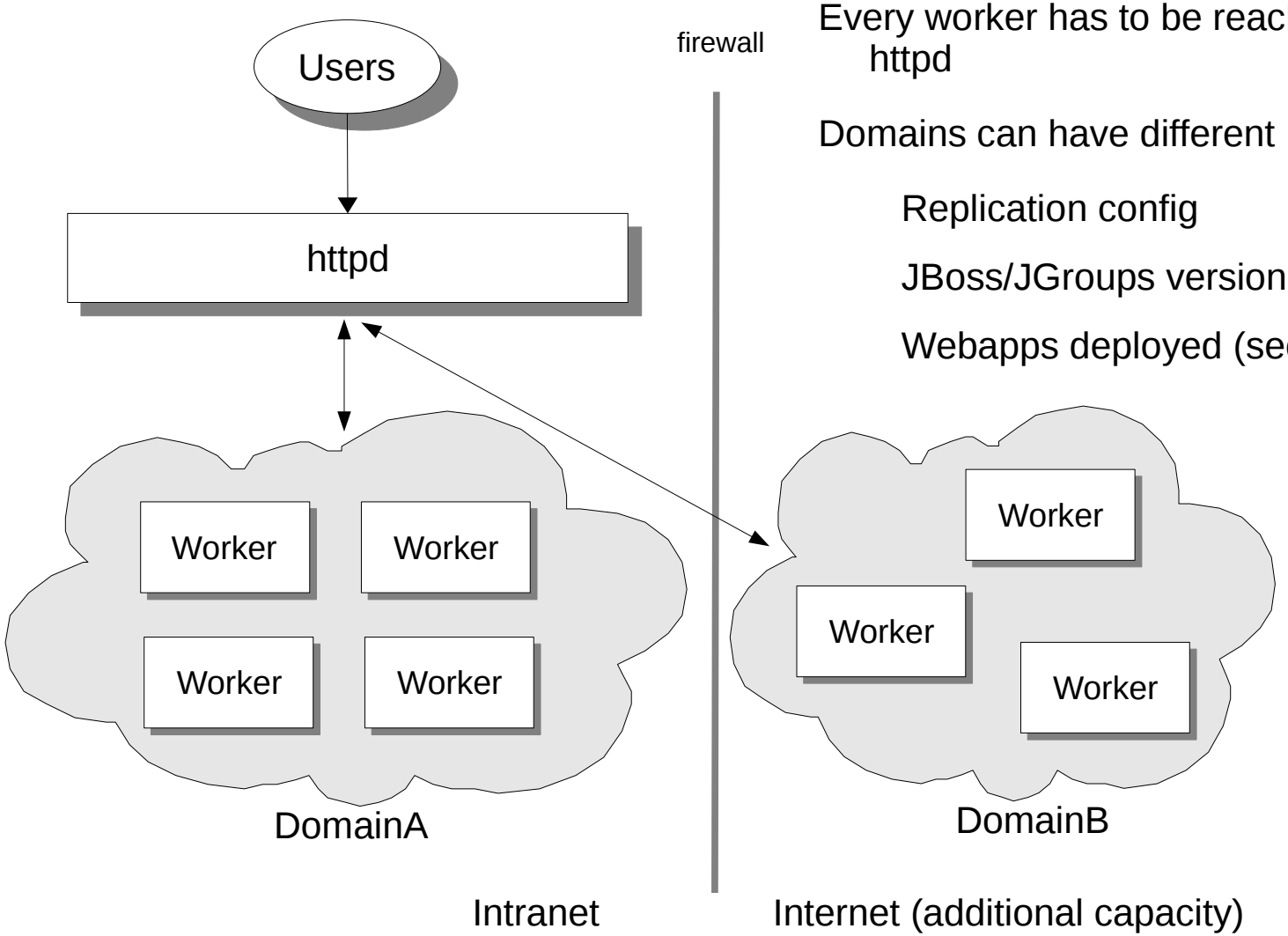
jboss1 and jboss2 register with httpd under domain A

jboss3 and jboss4 register with httpd under domain B

Every server registers the webapps it serves

- The same app doesn't need to be present in all nodes
- mod-cluster will only send requests to nodes which have it

Dynamically adding hosts when load increases



- httpd has to be reachable from every worker
- Every worker has to be reachable from httpd
- Domains can have different
  - Replication config
  - JBoss/JGroups versions
  - Webapps deployed (security)

Demo

Start a bunch of nodes in two different domains

→ mod-cluster allows for dynamically adding or removing of nodes to accommodate client traffic

This is all done on Amazon's EC2 cloud service, but of course this can be run on an internal cluster, too

Have we solved our 2 problems now ?

- Static configuration: yes
  - We only configure httpd once (adding modules)
  - Every JBoss instance is started with the following props
    - jboss.jvmRoute
    - jboss.Domain
    - jboss.modcluster.proxyList (not necessary if multicast advertisements are used)
- Rolling upgrades: yes
  - Same as mod-jk, with some added features
    - shutdown-when-drained.sh (in mod-cluster 1.1)
    - Disable, drain and shutdown entire domain (TBD, in JON/JOPR)
    - Start and enable entire domain (JON / JOPR)

Conclusion

- Divide-and-conquer
  - Divide large clusters into smaller subclusters (domains) to help rolling upgrades
- Use mod-cluster to
  - Dynamically add / remove nodes to / from a cluster
  - Dynamically deploy / undeploy webapps
  - Provide actual load information to help httpd distribute load optimally

Outlook

- Optimize JBoss clusters to run in the cloud
  - (Oftentimes) no multicasting available, large scale
    - Eliminate the TCP N-1 problem
  - Instances often don't have a static IP address
- Use of JON/JOPR to
  - Start, stop, disable and enable an entire cluster
  - Upgrading a domain will be a 3 step process
    - Drain and shut down the cluster
    - Upgrade all nodes of the cluster
    - Start the cluster again
  - View of a cluster topology, vitals, stats

QUESTIONS?

TELL US WHAT YOU THINK:
REDHAT.COM/JBOSSWORLD-SURVEY