Apache Tomcat

Tomcat Clustering: Part 2 – Load balancing

Mark Thomas, 15 April 2015
Introduction

• Apache Tomcat committer since December 2003
  – markt@apache.org

• Tomcat 8 release manager

• Member of the Servlet, WebSocket and EL expert groups

• Consultant Software Engineer @ Pivotal

• Currently focused on Apache Tomcat 9
Terminology
Reverse Proxy

bz.apache.org
httpd instance

Bugzilla (Main)
(bz.apache.org/bugzilla)
httpd instance

Bugzilla (AOO)
(bz.apache.org/ooo)
httpd instance

Bugzilla (SpamAassassin)
(bz.apache.org/SpamAssassin)
httpd instance
Load-balancing

Europe
www.eu.apache.org

US
www.us.apache.org

www.apache.org
geoip
Load-balancing

- Looks like a single host to the clients
- Multiple hosts
- Each host is the same
- Each host is independent
  - No shared state between the hosts
  - May share common services (e.g. authentication, database)
- Node failure may be visible to users
Load-balancing

• Lots of options for distributing the load
  – Hardware load-balancer
  – Round-robin DNS
  – Software load-balancer
    ▪ httpd
    ▪ pen
  – geoip
Load-balancing
Agenda

- Terminology
- Request distribution
- Managing state
- Failover
- Demonstration
  - Time permitting
- Questions
Terminology

• Sticky sessions

• Without clustering, session is created only on node that handled request

• On next request, the load-balancer could send user to a different node where the session doesn’t exist

• Sticky sessions is a mechanism (there are several) that ensures the user returns to the node holding their session
Request Distribution

• Many ways to select node to handle request
  • mod_proxy
    – Number of requests
    – Number of bytes returned
    – Number of current requests
  • mod_jk
    – As mod_proxy plus
    – Number of sessions (estimate)
Request distribution

- Client IP
  - Last octet

- Account number
  - Last digit 0-3, 4-6, 7-9

- Customer type
  - More important customers get priority
Managing State

• Stateless applications are the simple solution

• Application state
  – State includes authentication

• Options
  – HTTP session
  – Database
  – Request parameters

• Load-balancing is impacted by HTTP state
Managing State

- Sticky sessions are used for HTTP State
- Session ID
  - Something in the session ID identifies the correct node
  - Users could change this
- Dedicated cookie
  - Users could change this
- Property of client such as IP
  - Beware of ISP that use forward proxies
Managing State

• Application property
  – Account number
  – Account type

• Often overlaps with load-balancing algorithm
Failover

- Load-balancer needs to know the state of the nodes
- Nodes need to be taken off-line for maintenance
  - Known in advance
  - Several options
- Nodes will fail
  - Not (usually) predictable
  - Need to be able to detect dynamically
- What is the impact on users?
Failover

Maintenance

• More transparent to users means
  – More complex configuration
  – Process takes longer

• Need to drain node of users
  – How long can an HTTP session last?
  – At what point do you stop the node anyway?

• Can Tomcat’s parallel deployment feature help?
Failover

Unexpected

- Typically there is no separate management channel between Tomcat instances and load-balancer
  - There is with mod_cluster from JBoss

- Need to detect failed nodes so failover can happen as early as possible
Failover

Unexpected

• Can use a ‘failed’ request to detect a failed node

• Is a 500 response because the server crashed or because of an application bug?

• Is a timeout because the server crashed or because it is just a long running request?

• Applications that can have long running requests take at least that long to detect failures.
Failover

Unexpected

• Monitoring user initiated requests to detect node failure is fragile

• Load-balancer triggered request to known, working, ‘simple’ page
  – More reliable
  – Still an HTTP request with the associated overhead

• Protocol pings are even faster
Questions