A4M36JEE

Clustering & Scalability

LAB SESSION

Václav Chalupa
Quality Engineer
Red Hat

Radoslav Husár
Software Engineer
Red Hat

November 21, 2014

WildFly Clustering Configuration

Clustering Configuration

Clustering configuration prepared in two profiles:

- **ha** `standalone/configuration/standalone-ha.xml`
- **full-ha** `standalone/configuration/standalone-full-ha.xml`

We will use ha profile:

- `run server with -c standalone-ha.xml`

You can find xml schemas in docs/schema

Let's see the diff!

New Extensions for Clustering

- **jgroups**
- **modcluster**

- Infinispan extension is already in default profile
  - Why?

Distributed Cache for SFSB

```
<stateful cache-ref="distributable"
```

References cache:

```
<cache name="distributable" passivation-store-ref="infinispan"
```

References passivation store:

```
<passivation-store name="infinispan" cache-container="ejb"
```

Uses 'ejb' Infinispan cache container (later).

Infinispan Subsystem

Session replication, SFSB fail-over, Hibernate cache:

- New 'server' cache-container, replicated cache (for WF)
- Distributed cache for 'web' cache-container
- Distributed cache for 'ejb' cache-container
- Invalidation cache for Hibernate entities
- Replicated cache for Hibernate timestamps

JGroups Subsystem

Predefined two stacks:

- **udp (default)**
- **tcp**

Notice:

In case you cannot use udp (udp not allowed), use tcp stack.

- By default, tcp stack use MPING protocol – and it uses udp
- You have to replace MPING with TCPPING

Modcluster Subsystem

Load balancing, default is AJP connector:

- AJP listener added to 'undertow' subsystem:
  - `<ajp-listener name="ajp" socket-binding="ajp"/>`
- Load metric set to '**CPU**', you can choose different:
  - Memory
  - Requests
  - Heap
  - ...

New Socket Bindings

For:

- **JGroups**
- **mod_cluster**

- See **jboss.default.multicast.address** in JGroups bindings:
- All nodes with the same value will be in the same cluster
- You can create more clusters by changing this value

Clustering Demos

Starting point

```
$ cd 04-clustering-seminar # in main project
$ git checkout clustering-00 # should be set
$ mvn clean install
```

You can clone standalone clustering seminar:

```
$ git clone https://github.com/qa/a4m36jee-2014-clustering-seminar.git
```


Prepare Two Instances of WF

```
$ # unzip / rename WF to create two instances
$ ll
wildfly-8.1.0.Final
wildfly-8.1.0.Final-2
```


Run both Instances of WF

```
$ cd wildfly-8.1.0.Final

$ # export JBOSS-HOME if necessary

$ bin/standalone.sh -c standalone-ha.xml
-Djboss.node.name=`whoami`


$ cd ../wildfly-8.1.0.Final-2

$ # export JBOSS-HOME if necessary

$ bin/standalone.sh -c standalone-ha.xml
-Djboss.node.name=`whoami`2
-Djboss.socket.binding.port-offset=100
```

- **Why using offset?**

Chat over JGroups

JGroups Chat Task

Finish implementation of a simple chat using JGroups API using default UDP stack.

- Notify when new member joins a chat “room”
- Display messages from all member
- Send messages to all members
- You can add bonus features but keep the message format the same

Running

- Read Readme.md
- You can change your name by:
  - `-Duser.name="<your name>"`

Highly-Available WebApp

Task

Implement missing pieces in the WebApp to make it highly-available.

- Let the simple Servlet return number of times the Servlet has been invoked so that in case of fail-over it will enable us to verify if the session state is as expected.
- Add `?readonly=true` and `?invalidate=true` options.

Checkout **`clustering-01`** branch to start working.

Deploy the WebApp

- Read Readme.md
- How to simulate fail-over?

Load-balancing Remote SLSB
&
Remote Cluster-Aware SFSB

Task

- Implement clustered Stateless Session Bean and clustered Stateful Session Bean.
- Implement a remote EJB clients which connects to WildFly cluster and invokes operations on the beans.
- Demonstrate the load-balancing of SLSB and SFSB fail-over.

Checkout `clustering-02` branch to start working.

Running

- Read Readme.md

WebApp Load Balancing Demo

~FINISHED~
~~Happy Week!~~