ERJGroupsSynchronizer Framework

Getting Started

`ERJGroupsSynchronizer` is a multicast synchronizer built on top of the JGroups library. It is a much more robust implementation than the default synchronizer used by `ERXObjectStoreCoordinatorSynchronizer`. An excellent introductory resource is Mike Schrag's Project Wonder in Depth screencast from WOWODC West 2009.

Using `ERJGroupsSynchronizer.framework` on Amazon EC2

Problem

The problem here is simple: EC2 does not support multicast communication between hosts (even on VPC), and multicast is the default mode of operation for JGroups.

Solution

The jgroups-aws project from meltmedia "provides auto discovery for other cluster members on AWS using both tag matching and filters. It is a drop in replacement for TCPPING, allowing you to remove the definition of your initial members from your configuration file."

1. Include jgroups-aws in your project.
    a. If you're using Maven, include the following dependency in your POM:

    ```
    <dependency>
      <groupId>com.meltmedia.jgroups</groupId>
      <artifactId>jgroups-aws</artifactId>
      <version>1.4.1</version>
    </dependency>
    ```

    b. If you're using the Fluffy Bunny project layout, you can download the JAR directly from Maven Central and put it in `Libraries`.
    c. Otherwise you can just grab `AWS_PING.java` and include it somewhere appropriate.
2. Include `TCP` and `AWS_PING` sections in your JGroups XML configuration file.
3. Include these properties:

    ```
    er.extensions.ERXObjectStoreCoordinatorPool.maxCoordinators=1
    er.extensions.remoteSynchronizer.enabled=true
    er.extensions.remoteSynchronizer=er.jgroups.ERJGroupsSynchronizer
    ```

4. Follow the instructions in the Setting Up EC2 section of the jgroups-aws project page.
5. Although the jgroups-aws documentation seems to suggest it's not mandatory, I was unable to get initial discovery to occur *without* using the "tags" feature of the `AWS_PING` configuration. That is, it would *seem* to be the case that you need to apply at least one arbitrary tag to your instances, and `AWS_PING` then picks up all instances where the nominated tag has the same value.

`AWS_PING` will discover *all* instances *in the entire region* that have the matching key-value pair. You can't, for example, reuse the same key-value pair on instances in different VPCs *unless those instances are able to see each other*. (That is, there is no sense in which the VPC encapsulates some subset of all the instances tagged with the same key-value pair.) `AWS_PING` will discover *all* of the instances in the region and will try and sync *all* of those instances. If they can't see each other, you'll see *long* delays as JGroups tries to merge the view and eventually fails. If the VPCs (or other organisational groups) are distinct, use distinct key-value pairs.

The details above are not comprehensive. If anyone has an interest in getting this set up, but finds the overview on this page insufficient, by all means contact me.