IBM Content Manager OnDemand for Multiplatforms Active/Active Support

April 7, 2017

Greg Felderman

Chief Architect - IBM Content Manager OnDemand

Contents

Introduction

Overview

Before Content Manager OnDemand Version 10.1, the only active/active solution was to use Content Manager OnDemand for z/OS (any version), which supported running in a z/OS Parallel Sysplex environment (including DB2 and OAM). Prior to V10.1, Content Manager OnDemand for Multiplatforms supported a variety of different high availability options. Some customers have used a concept of dual load (shared-nothing), in which two or more independent Content Manager OnDemand instances are used and data is stored to each instance. If an issue occurs on one instance, the other instances can be used to handle the workload. This concept works fine for some scenarios, but is less than ideal if there are requirements for annotations, enhanced retention management, and so on, since those actions would have to be customized to ensure they occur on each instance. Other customers chose to use an active/passive model – given that the Content Manager OnDemand server (prior to V10.1) (ARSSOCKD/ARSOBJD) could only have a single process active at a time, and then have the passive process started once they detect that the active process had become unavailable. Content Manager OnDemand V10.1 now has an additional option for customers looking to achieve high availability. In V10.1, Content Manager OnDemand for Multiplatforms allows for 'n' number of server processes (ARSSOCKD/ARSOBJD) to be running on multiple nodes/LPARs at the same time.

To take advantage of an active/active model, it is important to understand that you must have an underlying database that supports active/active as well. As such, the use of DB2 pureScale, Oracle RAC, or SQL Server Clusters is required. For Content Manager OnDemand V10.1 to be able to become active/active, a clustering facility to synchronize actions across the Content Manager OnDemand processes has to be utilized. As such, Content Manager OnDemand leverages the Apache Zookeeper service which provides a centralized service for maintaining configuration information, naming, distributed synchronization, and group services.

Enabling active/active support

Apache ZooKeeper

Apache ZooKeeper is used to provide the synchronization services needed for multiple ARSSOCKD/ARSOBJD processes to run currently. ZooKeeper requires Java 1.7 or greater. Extensive documentation can be found at https://zookeeper.apache.org/

For a reliable ZooKeeper service, you should deploy ZooKeeper in a cluster known as an ensemble. If the majority of the ensemble are up, the service will be available. A minimum of three servers are required, and it is strongly recommended that you have an odd number of servers because ZooKepper requires a majority. If you only have two servers, then you are in a situation where if one of them fails, there are not enough machines to form a majority quorum. Two servers are inherently less stable than a single server, because there are two single points of failure.

Steps to use Zookeeper:

1) Extract the ZooKeeper code (zookeeper-3.4.9.tar.gz or later) into a location of your choosing:
   UNIX: gzip -c -d zookeeper-3.4.9.tar.gz | tar xvf –
   Windows: Use 7-zip or WinZip

2) Change to the ZooKeeper configuration directory: zookeeper-3.4.9/conf

3) Copy the zoo_sample.cfg file to zoo.cfg and make any necessary changes. (For example, the dataDir – which even on Windows should have a UNIX-like path – e.g. /, not \). Every machine that is part of the ZooKeeper ensemble should know about every other machine in the ensemble. You accomplish this with the series of lines of the form server.id=host:port:port. For example, define any cluster members into the configuration file:
   server.1=zoo1:2888:3888
   server.2=zoo2:2888:3888
   server.3=zoo3:2888:3888

   NOTE: The entries of the form server.X list the servers that make up the ZooKeeper service. When the server starts up, it knows which server it is by looking for the file myid in the data directory. That file contains the server number, in ASCII.

4) In the directory pointed to by dataDir, create a file called myid. The myid file consists of a single line containing only the text of that machine's ID. So myid of server 1 would contain the text "1" and nothing else. The ID must be unique within the ensemble and should have a value between 1 and 255.

5) Change directory to zookeeper-3.4.9/bin
Establish credentials for ZooKeeper service:

UNIX:

export ZOOBINDIR=$(pwd)
. "${ZOOBINDIR}"/zkEnv.sh

java -cp ${CLASSPATH} org.apache.zookeeper.server.auth.DigestAuthenticationProvider userid:password

Windows:

zkEnv.cmd

java -cp %CLASSPATH% org.apache.zookeeper.server.auth.DigestAuthenticationProvider userid:password

NOTE: This is the same userid/password you will put into the Content Manager OnDemand stash file.

arsstash -a 10 -s <stash_file> -u <userid>

6) Start/Stop ZooKeeper

UNIX:

zkServer.sh start
zkServer.sh stop

Windows:

zkServer.cmd

Content Manager OnDemand configuration (ARS.CFG file)

In order for Content Manager OnDemand to connect to the ZooKeeper service, you must define the ZooKeeper server(s) in the ARS.CFG file in the form of host:port and, if specifying multiple servers, they must be comma separated. For example:

ARS_LOCK_HOSTS=zoo1:2181,zoo2:2181,zoo3:2181