Prerequisites

Third party software prerequisites

Table shown below, specifies the supported versions of the third party software or protocol.

Compatible software/protocol	Supported version
MongoDB	v4.4.5
LDAP protocol	Version-3
Memcached	1.5

Third party software availability

1. MongoDB schema with initial data

   For the CAN application to come up successfully, it needs some initial data in MongoDB. However, if it is not available, then Avanseus delivery needs to be contacted for an initial data dump that needs to be imported to a MongoDB schema. MongoDB can either be within the Openshift platform or outside. This MongoDB schema must be accessible from the CAN operator pods, preferably on port 27017.

2. LDAP with initial user data

   For the CAN application, authentication is made possible using LDAP. Usually customers provide an LDAP connection for integration. However, if LDAP is not made available by the customer, then a LDAP pod comes as part of operator installation by default with one initial user. Avanseus delivery team needs to be requested to get the user access details. If the customer provides LDAP, then it can be outside Openshift or within the cluster. The LDAP Host and port details must be configured in a ConfigMap.

3. Memcached

   Memcached is used by CAN to cache small data for fast retrieval. This is also used for tomcat session replication. This application comes by default with the operator.

Configuration prerequisites

1. PersistentVolumeClaim to copy WAR files

By default, the CAN operator does not come with CAN binary. Before deploying, the binaries in WAR format need to be obtained separately and put into a PersistentVolume. Mentioned below is a method to do this.

Create a PersistenceVolumeClaim “tmp-data”. Below is an example configuration YAML

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: tmp-data
spec:
  accessModes:
    - ReadWriteOnce
  resources:
   requests:
    storage: 1G
```

Create a dummy application (Apache server) and mount the created PersistenceVolumeClaim. Below is the YAML content

```
apiVersion: v1
kind: Pod
metadata:
 name: pod-httpd
 labels:
   app: apache_webserver
spec:
 containers:
   - name: cntr-httpd
     image: httpd:latest
     ports:
       - containerPort: 80
     volumeMounts:
     - mountPath: /mnt
       name: tmp-data
 volumes:
 - name: tmp-data
   persistentVolumeClaim:
    claimName: tmp-data
```

Change directory to the place where WAR files are available. Copy the WAR files provided by Avanseus into the mount point "/mnt" using the rsync command.

```
oc rsync . pod-httpd:/mnt
```

Once this is done, the dummy application pod can be stopped. Remember that "tmp-data" PersistentVolume must not be deleted as this is used by Operator components to pull the binaries.

2. PersistentVolumeClaim for CAN application

Create a PersistentVolumeClaim named "candatapvc" & "canlogpvc". This is needed for the CAN application to write files. YAML configuration is given below.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
 name: candatapvc
spec:
 accessModes:
 - ReadWriteOnce
resources:
 requests:
  storage: 5G

---

kind: PersistentVolumeClaim
apiVersion: v1
metadata:
 name: canlogpvc
spec:
 accessModes:
 - ReadWriteOnce
resources:
 requests:
  storage: 5G
```

3. PersistentVolumeClaim for LDAP application

   Create a PersistentVolumeClaim named "ldap-storage". This is needed for LDAP module to store the user DB data in a persistent location. YAML configuration is given below.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
 name: ldap-storage
spec:
 accessModes:
 - ReadWriteOnce
resources:
 requests:
  storage: 5G
```

4. Create ConfigMaps with necessary information

   CAN configmap (canconfig)

   Mentioned below are the ConfigMap entries CAN application needs & hence create 3 files mentioned below:

`catalina.properties` - Catalina file for tomcat:

```
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

#
# List of comma-separated packages that start with or equal this string
# will cause a security exception to be thrown when
# passed to checkPackageAccess unless the
# corresponding RuntimePermission ("accessClassInPackage."+package) has
# been granted.
package.access=sun.,org.apache.catalina.,org.apache.coyote.,org.apache.jasper.,org.apache.tomcat.
#
# List of comma-separated packages that start with or equal this string
# will cause a security exception to be thrown when
# passed to checkPackageDefinition unless the
# corresponding RuntimePermission ("defineClassInPackage."+package) has
# been granted.
#
# by default, no packages are restricted for definition, and none of
# the class loaders supplied with the JDK call checkPackageDefinition.
#
package.definition=sun.,java.,org.apache.catalina.,org.apache.coyote.,\
org.apache.jasper.,org.apache.naming.,org.apache.tomcat.


#
#
# List of comma-separated paths defining the contents of the "common"
# classloader. Prefixes should be used to define what is the repository type.
# Path may be relative to the CATALINA_HOME or CATALINA_BASE path or absolute.
# If left as blank,the JVM system loader will be used as Catalina's "common"
# loader.
# Examples:
#     "foo": Add this folder as a class repository
#     "foo/*.jar": Add all the JARs of the specified folder as class
#                  repositories
#     "foo/bar.jar": Add bar.jar as a class repository
#
# Note: Values are enclosed in double quotes ("...") in case either the
#       ${catalina.base} path or the ${catalina.home} path contains a comma.
#       Because double quotes are used for quoting, the double quote character
#       may not appear in a path.
common.loader="${catalina.base}/lib","${catalina.base}/lib/*.jar","${catalina.home}/lib","${catalina.home}/lib/*.jar"
```

```
#
# List of comma-separated paths defining the contents of the "server"
# classloader. Prefixes should be used to define what is the repository type.
# Path may be relative to the CATALINA_HOME or CATALINA_BASE path or absolute.
# If left as blank, the "common" loader will be used as Catalina's "server"
# loader.
# Examples:
#     "foo": Add this folder as a class repository
#     "foo/*.jar": Add all the JARs of the specified folder as class
#                  repositories
#     "foo/bar.jar": Add bar.jar as a class repository
#
# Note: Values may be enclosed in double quotes ("...") in case either the
#       ${catalina.base} path or the ${catalina.home} path contains a comma.
#       Because double quotes are used for quoting, the double quote character
#       may not appear in a path.
server.loader=


#
# List of comma-separated paths defining the contents of the "shared"
# classloader. Prefixes should be used to define what is the repository type.
# Path may be relative to the CATALINA_BASE path or absolute. If left as blank,
# the "common" loader will be used as Catalina's "shared" loader.
# Examples:
#     "foo": Add this folder as a class repository
#     "foo/*.jar": Add all the JARs of the specified folder as class
#                  repositories
#     "foo/bar.jar": Add bar.jar as a class repository
# Please note that for single jars, e.g. bar.jar, you need the URL form
# starting with file:.
#
# Note: Values may be enclosed in double quotes ("...") in case either the
#       ${catalina.base} path or the ${catalina.home} path contains a comma.
#       Because double quotes are used for quoting, the double quote character
#       may not appear in a path.
shared.loader=


# Default list of JAR files that should not be scanned using the JarScanner
# functionality. This is typically used to scan JARs for configuration
# information. JARs that do not contain such information may be excluded from
# the scan to speed up the scanning process. This is the default list. JARs on
# this list are excluded from all scans. The list must be a comma separated list
# of JAR file names.
# The list of JARs to skip may be over-ridden at a Context level for individual
# scan types by configuring a JarScanner with a nested JarScanFilter.
# The JARs listed below include:
# - Tomcat Bootstrap JARs
# - Tomcat API JARs
# - Catalina JARs
# - Jasper JARs
# - Tomcat JARs
# - Common non-Tomcat JARs
# - Test JARs (JUnit, Cobertura and dependencies)
tomcat.util.scan.StandardJarScanFilter.jarsToSkip=\
annotations-api.jar,\
ant-junit*.jar,\
ant-launcher.jar,\
ant.jar,\
asm-*.jar,\
aspectj*.jar,\
bootstrap.jar,\
catalina-ant.jar,\
catalina-ha.jar,\
catalina-jmx-remote.jar,\
catalina-storeconfig.jar,\
catalina-tribes.jar,\
catalina-ws.jar,\
catalina.jar,\
cglib-*.jar,\
cobertura-*.jar,\
commons-beanutils*.jar,\
commons-codec*.jar,\
commons-collections*.jar,\
commons-daemon.jar,\
commons-dbcp*.jar,\
commons-digester*.jar,\
commons-fileupload*.jar,\
commons-httpclient*.jar,\
commons-io*.jar,\
commons-lang*.jar,\
commons-logging*.jar,\
commons-math*.jar,\
commons-pool*.jar,\
dom4j-*.jar,\
easymock-*.jar,\
ecj-*.jar,\
el-api.jar,\
geronimo-spec-jaxrpc*.jar,\
h2*.jar,\
hamcrest-*.jar,\
hibernate*.jar,\
httpclient*.jar,\
icu4j-*.jar,\
jasper-el.jar,\
jasper.jar,\
jaspic-api.jar,\
jaxb-*.jar,\
jaxen-*.jar,\
jdom-*.jar,\
jetty-*.jar,\
jmx-tools.jar,\
jmx.jar,\
jsp-api.jar,\
jstl.jar,\
jta*.jar,\
junit-*.jar,\
junit.jar,\
log4j*.jar,\
mail*.jar,\
objenesis-*.jar,\
oraclepki.jar,\
oro-*.jar,\
servlet-api-*.jar,\
servlet-api.jar,\
slf4j*.jar,\
taglibs-standard-spec-*.jar,\
tagsoup-*.jar,\
tomcat-api.jar,\
tomcat-coyote.jar,\
tomcat-dbcp.jar,\
tomcat-i18n-*.jar,\
tomcat-jdbc.jar,\
tomcat-jni.jar,\
tomcat-juli-adapters.jar,\
tomcat-juli.jar,\
tomcat-util-scan.jar,\
tomcat-util.jar,\
tomcat-websocket.jar,\
tools.jar,\
websocket-api.jar,\
wsdl4j*.jar,\
xercesImpl.jar,\
xml-apis.jar,\
xmlParserAPIs-*.jar,\
xmlParserAPIs.jar,\
xom-*.jar


# Default list of JAR files that should be scanned that overrides the default
# jarsToSkip list above. This is typically used to include a specific JAR that
# has been excluded by a broad file name pattern in the jarsToSkip list.
# The list of JARs to scan may be over-ridden at a Context level for individual
# scan types by configuring a JarScanner with a nested JarScanFilter.
tomcat.util.scan.StandardJarScanFilter.jarsToScan=\
log4j-taglib*.jar,\
log4j-web*.jar,\
log4javascript*.jar,\
slf4j-taglib*.jar

# String cache configuration.
tomcat.util.buf.StringCache.byte.enabled=true
#tomcat.util.buf.StringCache.char.enabled=true
#tomcat.util.buf.StringCache.trainThreshold=500000
#tomcat.util.buf.StringCache.cacheSize=5000


# This system property is deprecated. Use the relaxedPathChars relaxedQueryChars
# attributes of the Connector instead. These attributes permit a wider range of
# characters to be configured as valid.
# Allow for changes to HTTP request validation
# WARNING: Using this option may expose the server to CVE-2016-6816
#tomcat.util.http.parser.HttpParser.requestTargetAllow=|
PASSWORD_KEY=<AVANSEUS_PASSWORD_KEY>
```

`config.properties` - Application configuration

```
#Domain details
```

```
avanseus.app.cas.domain=cas-<NAMESPACE>.<ROUTER_DEFAULT_HOSTNAME>
avanseus.app.can.domain=can-<NAMESPACE>.<ROUTER_DEFAULT_HOSTNAME>


avanseus.vbi.ip=vbi
avanseus.vbi.port=12001


#Domain protocol
avanseus.protocol=https


#Host details
avanseus.app.cas.host=canapp:2003
avanseus.app.can.host=canapp:2000


#Memcached configuration
avanseus.memcached.url=memcached:11211


#MongoDb configuration
avanseus.mongodb.ssl.enabled=<MONGODB_SSL_ENABLED>
avanseus.mongodb.validHostName=<MONGODB_VERIFY_HOST>
avanseus.mongodb.keystore.path=/data/workspace/pemfile/mongo.pkcs12
avanseus.mongodb.keystore.password=<MONGODB_KEYSTORE_PASSWORD>
avanseus.mongodb.ip=<MONGODB_IP_HOST>
avanseus.mongodb.port=<MONGODB_PORT>
avanseus.mongodb.username=<MONGODB_USERNAME>
avanseus.mongodb.password=<MONGODB_PASSWORD>
avanseus.mongodb.dbName=<MONGODB_SCHEMA_NAME>


avanseus.mongodb.admin.username=<MONGODB_ADMIN_USERNAME>
avanseus.mongodb.admin.password=<MONGODB_ADMIN_PASSWORD>
avanseus.mongodb.admin.dbName=<MONGODB_ADMIN_SCHEMA_NAME>


#log configuration
avanseus.log.path=/data/workspace/logs/
avanseus.log.thread.poolsize=20


#LDAP server configuration
avanseus.ldap.organisation=<LDAP_ORG>
avanseus.ldap.domain=<LDAP_DOMAIN>
avanseus.ldap.url1=ldap://<LDAP_URL1>:<LDAP_PORT1>
avanseus.ldap.url2=ldap://<LDAP_URL2>:<LDAP_PORT2>
avanseus.ldap.userDn=cn=Directory Manager
avanseus.ldap.password=<LDAP_PASSWORD>


avanseus.app.googleKey=<GOOGLE_MAP_KEY>
#NAS path
avanseus.app.nas.path=/data/workspace/nasmount/
```

```
#Email Config
avanseus.email.fromId=<EMAIL_FROM_ID>
avanseus.email.pwd=<EMAIL_PASSWORD>
avanseus.email.host=<EMAIL_SERVER>
avanseus.email.fromName=<EMAIL_SENDER_NAME>
avanseus.email.port=<EMAIL_PORT>


#Cluster node configuration
avanseus.predictionNode.name=nodeA
```

`mongo.pkcs12` - Use a valid key file if provided or create an empty file

In the above mentioned files, replace the text enclosed in `<>` with necessary information. Mentioned below are their details:

- `<AVANSEUS_PASSWORD_KEY>` : Replace this with the password key to be used. Request the Avanseus delivery team for this. This is very specific for a given DB dump.
- `<NAMESPACE>` : Namespace where the operator and application would run.
- `<ROUTER_DEFAULT_HOSTNAME>` : Replace this with the Openshift cluster default router hostname.
- `<MONGODB_SSL_ENABLED>` : `true` when MongoDB is enabled with SSL. `false` otherwise.
- `<MONGODB_VERIFY_HOST>` : `true` when MongoDB host name needs to be verified with SSL certificate. `false` otherwise.
- `<MONGODB_KEYSTORE_PASSWORD>` : Encrypted password of MongoDB keystore. The encryption technique is defined here
- `<MONGODB_IP>` : Replace this with MongoDB IP/Host which is reachable from the pods.
- `<MONGODB_PORT>` : Replace this with MongoDB Port which is reachable from the pods.
- `<MONGODB_USERNAME>` : Replace this with MongoDB schema read/write username.
- `<MONGODB_PASSWORD>` : Replace this with above username's password (Encrypted). The encryption technique is defined here
- `<MONGODB_SCHEMA_NAME>` : Replace this with DB schema name.
- `<MONGODB_ADMIN_USERNAME>` : Replace this with MongoDB admin username
- `<MONGODB_ADMIN_PASSWORD>` : Replace this with MongoDB admin password (Encrypted). The encryption technique is defined here
- `<MONGODB_ADMIN_SCHEMA_NAME>` : Replace this with the MongoDB admin schema name.
- `<GOOGLE_MAP_KEY>` : Google map key.
- `<LDAP_URL1>` : Replace this with LDAP URL 1. LDAP IP reachable from pods.
- `<LDAP_URL2>` : Replace this with LDAP URL 2. Same as above if the customer does not provide failover LDAP URL.
- `<LDAP_PORT1>` : Replace this with LDAP Port 1. LDAP port reachable from pods.
- `<LDAP_PORT2>` : Replace this with LDAP Port 2. Same as above if the customer does not provide failover LDAP instance.
- `<LDAP_PASSWORD>` : Replace this with LDAP password (Encrypted). The encryption technique is defined here
- `<LDAP_ORG>` : Replace this with LDAP organisation. Customers usually provide this or contact Avanseus support team for this.

- `<LDAP_DOMAIN>` : Replace this with LDAP domain. Customers usually provide this or contact Avanseus support team for this.
- `<EMAIL_FROM_ID>` : Replace this with Email ID of sender.
- `<EMAIL_SENDER_NAME>` : Replace this with Email sender's name.
- `<EMAIL_SERVER>` : Replace this with Email server host or domain.
- `<EMAIL_PORT>` : Replace this with Email server port.
- `<EMAIL_PASSWORD>` : Replace this with Email sender's password.

The last 5 entries must be filled up with relevant SMTP server or relay server information which is reachable from the Openshift cluster.

If you would use the Default LDAP server coming with the operator, then the LDAP configuration section can be as follows:

```
#LDAP server configuration
avanseus.ldap.organisation=employee
avanseus.ldap.domain=avanseus
avanseus.ldap.url1=ldap://ldap:1389
avanseus.ldap.url2=ldap://ldap:1389
avanseus.ldap.userDn=cn=Directory Manager
avanseus.ldap.password=r7qs0Kobh5s+q0Wtlj1JZbtkdqGxBdx/
```

Finally create the ConfigMap as follows:

```
oc create configmap canconfig --from-file=catalina.properties --from-file=config.properties
--from-file=mongo.pkcs12
```

Prediction controller configmap (controllerconfig)

Mentioned below is the ConfigMap entry Prediction controller application needs & hence create the file mentioned below:

`application.properties` - Application configuration

```
server.port=5052
logging.level.root = INFO
spring.application.name=PredictionControllerApplication
predictionworker.endpoint=http://avanseusworkernode:5050/PredictionWorker/
#predictionworker.endpoint=http://localhost:5050/
# Max file size.
spring.servlet.multipart.max-file-size=40MB
# Max request size.
spring.servlet.multipart.max-request-size=50MB
```

Finally create the ConfigMap as follows:

```
oc create configmap controllerconfig --from-file=application.properties
```

Installation

CAN Operator is available in the OperatorHub. Below is a link to the video which shows how to install the CAN operator.

https://avanseuscandev.com/releases/CAN/5.5/Videos/can-operator-openshift-installation-guide.mp4

Post installation

After installation, CAN application would be available on following URL:

https://can-<NAMESPACE>.<ROUTER_DEFAULT_HOSTNAME>/CAN/

If CAN tool was integrated with LDAP on customer premise, then the user credentials would be known.
If CAN tool was integrated with Avanseus's light weight LDAP, then request the Avanseus delivery team for credentials.