NICS 6 Install, Configure, and Deploy

Repositories:

nics-core-processor

Contains standalone components for handling specialized NICS functionality.

The collab-feed-manage listens for feature updates to the collaborations rooms and creates a data layer on the mapserver that represents the room. The layer can be exported as a KML or Shape file using NICS or uploaded to NICS and displayed in the Data tree.

nics-assembly

Contains build tools for assembling deployment artifacts.

nics-tools

Contains tools used to interface with OpenAM.

nics-db

Contains scripts for building the NICS database as well as any change files associated with database updates

nics-common

Contains common libarries used across several applications.

nics-dao interfaces with the database.

dao contains utilities to help form database queries.

em-api

RESTful API that serves NICS requests.

iweb-modules

Core web framekwork containing a default ExtJS view and communications Mediator that allows the client to listen and subscribe to the message bus as well as make web service requests.

Map Module containing a default Map View (Open Street Map) and Controller.

Draw Menu that allows drawing lines, shapes, text and other features on the map.

Census Application (Description)

Weather Application (Description)

nics-web

NICS web application for real-time situational awareness.

(List and describe each module) ?

Build NICS:

Prerequisites

- GitHub Account
- Maven 3.3.1
- Oracle JDK 7

Acquire NICS

a. Create NICS6 directory in $HOME directory

```
> mkdir nics6
```

b. Clone the following repositories:
   i. nics-assembly
   ii. nics-common
   iii. nics-core-processor
   iv. nics-db
   v. nics-tools
   vi. nics-web
   vii. em-api
   viii. iweb-modules

Building NICS

a. Go into nics-common directory and build it using the following command (add path to mvn in your $PATH if necessary)

```
> export PATH=$PATH:/usr/local/maven/bin
> mvn clean install -DskipTests=true

[INFO]
[INFO] Next-Generation Incident Command System (NICS) .... SUCCESS [1.761s]
[INFO] DAO ................................................ SUCCESS [1.897s]
[INFO] Encryption Library ................................. SUCCESS [0.105s]
[INFO] NICS System Entities ............................... SUCCESS [1.221s]
[INFO] GeoServer REST API ................................. SUCCESS [0.198s]
[INFO] Hash Function Library .............................. SUCCESS [0.180s]
[INFO] NICS Messages ...................................... SUCCESS [0.375s]
[INFO] NICS Message Parser ................................ SUCCESS [0.211s]
[INFO] RabbitMQ Admin Interface ........................... SUCCESS [0.072s]
[INFO] RabbitMQ Client .................................... SUCCESS [0.162s]
[INFO] Web Service Request Library ........................ SUCCESS [0.070s]
[INFO] Constants .......................................... SUCCESS [0.049s]
[INFO] NICS DAO ........................................... SUCCESS [1.015s]
[INFO]
------------------------------------------------------------------------
[INFO] BUILD SUCCESS
```

b. ./nics-tools

```
> mvn clean install

Reactor Summary:
[INFO]
[INFO] master-pom ......................................... SUCCESS [1.122s]
[INFO] OpenAm Tools ....................................... SUCCESS [9.034s]
[INFO] sso-tools .......................................... SUCCESS [0.446s]
[INFO] NICS User Management Tools ......................... SUCCESS [0.567s]
[INFO]
------------------------------------------------------------------------
[INFO] BUILD SUCCESS
```

c. ./em-api

```
> mvn clean install

[INFO] Reactor Summary:
[INFO]
[INFO] em-api ............................................. SUCCESS [
0.351 s]
[INFO] api-rest-common .................................... SUCCESS [
1.906 s]
[INFO] api-rest-service ................................... SUCCESS [
24.351 s]
[INFO]
------------------------------------------------------------------------
[INFO] BUILD SUCCESS
```

d. ./nics-assembly

```
> mvn clean install

[INFO] Reactor Summary:
[INFO]
[INFO] NICS Build Tools ................................. SUCCESS [
0.328 s]
[INFO] NICS Archive Assembly Tool ....................... SUCCESS [
1.048 s]
[INFO] NICS Spring Deploy Assembly Tool .................. SUCCESS [
0.038 s]
[INFO]
------------------------------------------------------------------------
[INFO] BUILD SUCCESS
```

e. ./nics-core-processor

```
> mvn clean install

[INFO] Reactor Summary:
[INFO]
[INFO] Next-Generation Incident Command System (NICS) Processors  SUCCESS
[1.063s]
[INFO] Collab Feed Manager .............................. SUCCESS [3.898s]
[INFO] GeoDataFeed Consumer ............................. SUCCESS [2.421s]
[INFO] JSON PLI Consumer ................................ SUCCESS [1.305s]
[INFO] gst2gml .......................................... SUCCESS [1.087s]
[INFO] Spring Runner .................................... SUCCESS [0.497s]
[INFO] NICS Component Manager Archive Builder ............ SUCCESS [4.106s]
[INFO]
------------------------------------------------------------------------
[INFO] BUILD SUCCESS
```

f. ./iweb-modules

```
> mvn clean install

[INFO] Reactor Summary:
[INFO]
[INFO] iWeb Modules Pom ................................. SUCCESS [
0.362 s]
[INFO] iWeb Core Module ................................. SUCCESS [
12.790 s]
[INFO] iWeb Map Module .................................. SUCCESS [
1.249 s]
[INFO] iWeb Draw Menu Module ............................ SUCCESS [
1.134 s]
[INFO]
------------------------------------------------------------------------
[INFO] BUILD SUCCESS
```

g. ,/nics-web

```
> mvn clean install

[INFO] Reactor Summary:
[INFO]
[INFO] NICS Web Pom ..................................... SUCCESS [0.405s]
[INFO] NICS Modules Pom ................................. SUCCESS [0.009s]
[INFO] NICS Collab Room Module .......................... SUCCESS [2.152s]
[INFO] NICS Incident Module ............................. SUCCESS [0.232s]
[INFO] NICS Login Module ................................ SUCCESS [1.362s]
[INFO] NICS Report Module ............................... SUCCESS [0.177s]
[INFO] NICS Damage Report Module ........................ SUCCESS [0.140s]
[INFO] NICS General Report Module ....................... SUCCESS [0.143s]
[INFO] NICS Explosives Report Module .................... SUCCESS [0.228s]
[INFO] NICS MITAM Report Module ......................... SUCCESS [0.170s]
[INFO] NICS Datalayer Module ............................ SUCCESS [0.978s]
[INFO] NICS Whiteboard Module ........................... SUCCESS [0.205s]
[INFO] NICS Feature Persistence Module .................. SUCCESS [0.142s]
[INFO] NICS Active Users Module ......................... SUCCESS [0.149s]
[INFO] NICS Admin Module ................................ SUCCESS [0.140s]
[INFO] NICS Common Module ............................... SUCCESS [0.652s]
[INFO] NICS Feature Photos Module ....................... SUCCESS [0.131s]
[INFO] NICS Print Module ................................ SUCCESS [0.141s]
[INFO] NICS Account Info Module ......................... SUCCESS [0.143s]
[INFO] NICS Webapp ...................................... SUCCESS [3.792s]
[INFO]
------------------------------------------------------------------------
[INFO] BUILD SUCCESS
```

Configure and Install NICS:

The NICS web application is deployed to tomcat on a server that is also running RabbitMQ. The RESTful API (em-api) can be deployed on the same server or a different tomcat server. The instructions are the same but the em-api config files need updated to point at the correct RabbitMQ message bus. See an explanation of configuration properties in the NICS Configuration Properties section below.

Prerequisites

- Rabbitmq 3.4.4
- Tomcat 8
- Oracle JDK 7

EM-API

a. Run the following commands in the / directory

```
> mkdir -p /opt/data/nics/config
```

b. Copy and configure the following config files and put them in /opt/data/nics/config
   i. ../em-api/api-rest-service/src/main/config/em-api.properties
   ii. AMConfig.properties
   iii. openam.properties
   iv. openam-tools.properties
   v. sso-tools.properties

c. Deploy the /nics6/em-api/api-rest-service/target/em-api.war file to the /var/lib/tomcat8/webapps directory

NICS

a. Run the following commands in the / directory

```
> mkdir -p /opt/data/nics/config
```

b. Copy and configure the following config files and put them in /opt/data/nics/config
   i. core.properties
   ii. AMConfig.properties
   iii. openam.properties
   iv. openam-tools.properties
   v. sso-tools.properties

c. Run the following command to get the status of your rabbitmq server and to view what your rabbit node name is

```
> sudo service rabbitmq-server status
```

d. Deploy the /nics6/nics-web/web-app/target/nics.war file to /var/lib/tomcat8/webapps directory

Configure and Install NICS Database:

Prerequisites

- Postgres 9.4
- Postgis 2.1

Configure PostgreSQL

Edit PostgreSQL configuration for Application Server Use

a. Edit the postgresql.conf file to allow external connections to the database:

```
# As root user or postgres user, open the postgresql.conf file with your
favorite editor
> vi /etc/postgresql/9.4/main/postgresql.conf
```

b. Locate the section labeled "CONNECTIONS AND AUTHENTICATION", and edit the original to match the edited value below:

Original:

```
#------------------------------------------------------------------------
----
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------
----
# - Connection Settings -
#listen_addresses = 'localhost'         # what IP address(es) to listen on;
                                        # comma-separated list of
addresses;
                                        # defaults to 'localhost', '*' =
all
                                        # (change requires restart)
#port = 5432                             # (change requires restart)
max_connections = 100                   # (change requires restart)


...

ssl = true                              # (change requires restart)
```

Edited: Set listen_addresses to '*', make sure port is not commented out and comment out the ssl line ,

```
#------------------------------------------------------------------------
----
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------
----
# - Connection Settings -
listen_addresses = '*'                  # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost', '*' = all
                                        # (change requires restart)
port = 5432                             # (change requires restart)
max_connections = 100                   # (change requires restart)
...
#ssl = true                             # (change requires restart)
```

c. After most configuration changes in this file, you'll need to restart the PostgreSQL server

```
> /etc/init.d/postgresql restart
 * Restarting PostgreSQL 9.4 database server
[ OK ]
>
```

d. Ensure PostgreSQL restarted successfully. Several parameter changes may fail due to the VM's kernel resources not being configured to support the PostgreSQL settings

Configure Postgres Users

Configure the Postgres User

The postgres user should have been set up as a nologin user.

a. Edit the pg_hba.conf file:

```
# As the postgres user, open the pg_hba.conf file:
> su - -s /bin/bash postgres
> vi /etc/postgresql/9.4/main/pg_hba.conf
```

b. Edit the local postgres user line to match the edit. We're changing method from peer to md5.
Original:

```
# TYPE  DATABASE        USER            ADDRESS                 METHOD

# Database administrative login by Unix domain socket
local   all             postgres                                peer
# IPv4 local connections:
host    all             all             127.0.0.1/32            md5
```

Edit:

```
# TYPE  DATABASE        USER            ADDRESS                 METHOD

# Database administrative login by Unix domain socket
local   all             postgres                                md5     #
EDIT!
# IPv4 local connections:
host    all             all             127.0.0.1/32            md5
```

Configure the NICS User

a. Create nics user in the database

```
>sudo  su - -s /bin/bash postgres
>psql USER CREATE  nics    #creates new user nics
```

b. Repeat step 2 in the previous section for adding the nics user with the following line. Put it below the one for the postgres user under "# IPv4 local connections":

```
host    all             nics        all                     md5
```

Create a spatially enabled template for PostGIS databases

a. su to the postgres user, and create the template database

```
> sudo  su - -s /bin/bash postgres
> createdb postgistemplate
> createlang plpgsql postgistemplate
createlang: language "plpgsql" is already installed in database
"postgistemplate"
```

NOTE: Since createlang is returning the fact that plpgsql is already installed, then maybe we can remove that step from these instructions? TODO

b. Load the PostGIS functions and objects into the template

```
> psql postgistemplate -f
/usr/share/postgresql/9.4/contrib/postgis-2.1/postgis.sql
```

c. Load the PostGIS coordinate system definitions

```
> psql postgistemplate -f
/usr/share/postgresql/9.4/contrib/postgis-2.1/spatial_ref_sys.sql
sudo
```

Configure NICS Database

a. Create the NICS database : /nics-db

```
> sudo  su - -s /bin/bash postgres
> ./create_db.sh nics
> ./create_data_dbs.sh nics
```

b. ./nics-db/scripts

```
> sudo  su - -s /bin/bash postgres
> ./create_system.sh localhost desc 1 workspacename 1;
> ./create_org.sh <orgName> <orgCounty> <orgState> <orgPrefix> <orgTypeId>;
> ./create_default_user.sh <your email> 1 1;
```

c. Run all the sql scripts in the nics-db/changes directory

```
> sudo  su - -s /bin/bash postgres
> psql -f XXXX_Changes.sql nics
  .....
```

d. Restart postgres

Install Geoserver

It is recommended that geoserver be deployed to a dedicated server running Tomcat8.

Prerequisites

- Tomcat 8
- Oracle JDK 7

Install JAI

a. Acquire the JAI library

```
# As root
> cd /opt
> wget
http://download.java.net/media/jai/builds/release/1_1_3/jai-1_1_3-lib-linux
-amd64-jdk.bin
```

b. Make the .bin file executable

```
> chmod u+x jai-1_1_3-lib-linux-amd64-jdk.bin
```

c. Execute the bin in your JDK directory

```
> cd $JAVA_HOME
> /opt/jai-1_1_3-lib-linux-amd64-jdk.bin
```

d. You will be shown a license. Keep pressing space bar to page through it, and you'll be asked to accept it by typing "yes" and pressing enter

```
Sun Microsystems, Inc.
Binary Code License Agreement
JAVA ADVANCED IMAGING API, VERSION 1.1.3
READ THE TERMS OF THIS AGREEMENT AND ANY PROVIDED SUPPLEMENTAL LICENSE
TERMS (COLLECTIVELY "AGREEMENT") CAREFULLY
BEFORE OPENING THE SOFTWARE MEDIA PACKAGE. BY OPENING THE SOFTWARE MEDIA
PACKAGE, YOU AGREE TO THE TERMS OF THIS
AGREEMENT. IF YOU ARE ACCESSING THE SOFTWARE ELECTRONICALLY, INDICATE YOUR
ACCEPTANCE OF THESE TERMS BY SELECTING
 THE "ACCEPT" BUTTON AT THE END OF THIS AGREEMENT. IF YOU DO NOT AGREE TO
ALL THESE TERMS, PROMPTLY RETURN THE UN
USED SOFTWARE TO YOUR PLACE OF PURCHASE FOR A REFUND OR, IF THE SOFTWARE IS
ACCESSED ELECTRONICALLY, SELECT THE "D
ECLINE" BUTTON AT THE END OF THIS AGREEMENT.

... (truncated)

For inquiries please contact: Sun Microsystems, Inc., 4150 Network Circle,
Santa Clara, California 95054, U.S.A


(LFI#143342/Form ID#011801)

Do you agree to the above license terms? [yes or no]
yes
Unpacking...
Checksumming...
0
0
Extracting...
UnZipSFX 5.50 of 17 February 2002, by Info-ZIP (Zip-Bugs@lists.wku.edu).
  inflating: COPYRIGHT-jai.txt
  inflating: DISTRIBUTIONREADME-jai.txt
  inflating: LICENSE-jai.txt
  inflating: THIRDPARTYLICENSEREADME-jai.txt
  inflating: UNINSTALL-jai
  inflating: jre/lib/amd64/libmlib_jai.so
  inflating: jre/lib/ext/jai_core.jar
  inflating: jre/lib/ext/jai_codec.jar
  inflating: jre/lib/ext/mlibwrapper_jai.jar
Done.
```

e. test

Install IMAGEIO

a. Acquire the imageio library

```
# As root
> cd /opt
> wget
http://download.java.net/media/jai-imageio/builds/release/1.1/jai_imageio-1
_1-lib-linux-amd64.tar.gz
> wget
http://download.java.net/media/jai-imageio/builds/release/1.1/jai_imageio-1
_1-lib-linux-amd64-jre.bin
```

b. Apply a fix to the bin file

```
# This fixes an error in the bin, and makes it so the bin doesn't register
as corrupt
> sed s/+215/-n+215/ jai_imageio-1_1-lib-linux-amd64-jre.bin >
jai_imageio-1_1-lib-linux-amd64-jre-fixed.bin
```

   i. NOTE: You now created a new file: jai_imageio-1_1-lib-linux-amd64-jre-fixed.bin. This one will be used in the next steps, not the original you downloaded

c. Make the newly fixed .bin file executable

```
> chmod u+x jai_imageio-1_1-lib-linux-amd64-jre-fixed.bin
```

d. Change directories to your $JAVA_HOME/jre directory, and execute the bin

```
> cd $JAVA_HOME/jre
> /opt/jai_imageio-1_1-lib-linux-amd64-jre-fixed.bin
```

e. You'll be shown a license. Page through it by pressing space bar until you see a prompt asking you to agree to the terms. Enter yes and press enter

```
... (truncated)

Please contact Sun Microsystems, Inc. 4150 Network Circle, Santa Clara,
California 95054 if you have questions.
Do you agree to the above license terms? [yes or no]
yes
Unpacking...
Checksumming...
0
0
Extracting...
UnZipSFX 5.50 of 17 February 2002, by Info-ZIP (Zip-Bugs@lists.wku.edu).
  inflating: COPYRIGHT-jai_imageio.txt
  inflating: DISTRIBUTIONREADME-jai_imageio.txt
  inflating: ENTITLEMENT-jai_imageio.txt
  inflating: LICENSE-jai_imageio.txt
  inflating: THIRDPARTYLICENSEREADME-jai_imageio.txt
  inflating: UNINSTALL-jai_imageio
  inflating: lib/amd64/libclib_jiio.so
  inflating: lib/ext/jai_imageio.jar
  inflating: lib/ext/clibwrapper_jiio.jar
Done.
>
```

Alternate tar.gz method

a. NOTE: Using the .tar.gz because both jre.bin and jdk.bin files complain about being corrupt, along with invalid use of switches passed to the tail command in the bin
   i. Extract the tar

```
> tar -zxvf jai_imageio-1_1-lib-linux-amd64.tar.gz
```

   ii. There's now an /opt/jai_imageio-1_1 directory

```
> ll /opt/jai_imageio-1_1
total 56
drwxrwxr-x 3 35320 staff  4096 Oct 13  2006 ./
drwxr-xr-x 4 root  root   4096 Jan 23 11:38 ../
-rw-rw-r-- 1 35320 staff  2462 Oct 13  2006 COPYRIGHT-jai_imageio.txt
-rw-rw-r-- 1 35320 staff  1530 Oct 13  2006
DISTRIBUTIONREADME-jai_imageio.txt
-rw-rw-r-- 1 35320 staff  2477 Oct 13  2006
ENTITLEMENT-jai_imageio.txt
drwxrwxr-x 2 35320 staff  4096 Oct 13  2006 lib/
-rw-rw-r-- 1 35320 staff 14412 Oct 13  2006 LICENSE-jai_imageio.txt
-rw-rw-r-- 1 35320 staff  8688 Oct 13  2006
THIRDPARTYLICENSEREADME-jai_imageio.txt
-rw-rw-r-- 1 35320 staff   477 Oct 13  2006 UNINSTALL-jai_imageio
```

   iii. Copy the contents of the lib folder to the following jre folders

```
> cp /opt/jai_imageio-1_1/lib/libclib_jiio.so
$JAVA_HOME/jre/lib/amd64/
> cp /opt/jai_imageio-1_1/lib/jai_imageio.jar $JAVA_HOME/jre/lib/ext/
> cp /opt/jai_imageio-1_1/lib/clibwrapper_jiio.jar
$JAVA_HOME/jre/lib/ext/
```

b. TODO: Still need to verify the above workaround works.
c. After installing GeoServer in the following step, there'll be a step for verifying that JAI and JAI IMAGEIO were successfully installed

Install GeoServer

a. Acquire the latest GeoServer here: http://geoserver.org/display/GEOS/Stable
    i. As of this writing, 2.2.4 is the latest. Download the "Web Archive" (war)

```
# As root
> cd /opt
> wget
http://downloads.sourceforge.net/geoserver/geoserver-2.2.4-war.zip
```

b. Extract geoserver

```
> unzip geoserver-2.2.4-war.zip -d geoserver-2.2.4
Archive:  geoserver-2.2.4-war.zip
  inflating: geoserver-2.2.4/geoserver.war
  inflating: geoserver-2.2.4/GPL.txt
  inflating: geoserver-2.2.4/LICENSE.txt
   creating: geoserver-2.2.4/target/
  inflating: geoserver-2.2.4/target/VERSION.txt
>
```

c. Copy the war directory to the tomcat webapps directory

```
> cp geoserver-2.2.4/geoserver.war /var/lib/tomcat8/webapps
```

d. Restart tomcat7

```
> service tomcat7 restart
 * Stopping Tomcat servlet engine tomcat8
[ OK ]
 * Starting Tomcat servlet engine tomcat8
[ OK ]
```

    i. Ensure tomcat7 restarted successfully
e. Test GeoServer
    i. Bring up: http://<vmhost/ip>:<tomcat port>/geoserver/ in a browser, and you should get GeoServer's home screen. If tomcat7 was set to run on a port other than 8080, it may not initially be reachable. But the default 8080 port should be reachable by default.
f. Verify JAI and JAI ImageIO were successfully installed, and are being used by GeoServer
    i. Login with admin/geoserver (unless you've changed the default password, in which case login with those credentials)

ii. On the upper left menu, click Server Status
iii. In the list, you should see two entries: Native JAI, and Native JAI ImageIO. Both should show "true". If one or both say "false", then they were not installed correctly

Configure Java System Properties

a. In `/usr/share/tomcat7/bin/` create a file called `setenv.sh`
b. Inside place the following:

```
JAVA_OPTS="-server -Xms1024m -Xmx1024m -XX:NewSize=256m -XX:MaxNewSize=256m
-XX:PermSize=256m -XX:MaxPermSize=256m"
GEOSERVER_DATA_DIR=/data/geoserver
```

The `GEOSERVER_DATA_DIR` property specifies where geoserver will store its data. This is normally done on a mounted drive on the mapserver VM. Normally this is the /data/geoserver directory, but it could potentially be different.
c. Restart tomcat7 as above for the settings to take effect

Clean up installation

a. Read the file located at `/var/lib/tomcat7/webapps/geoserver/data/security/masterpw.info` and enter the root username/password combo to the NICS Password Safe.
b. Now remove the masterpw.info and user.properties.old file:

```
> sudo rm /var/lib/tomcat8/webapps/geoserver/data/security/masterpw.info
> sudo rm
/var/lib/tomcat8/webapps/geoserver/data/security/users.properties.old
```

c. Change the admin password through the admin web interface. Save the admin username/password combo to the NICS Password Safe.
d. Through the web interface, remove all the workspaces and styles (built-in styles will not be removed - this is okay). You should be left with a clean GeoServer setup.

Configure and Install CollabFeedManager:

The collabfeed manager runs on the web machine and listens for feature updates to collaboration rooms. It creates a datalayer in geoserver that can be imported into and exported from NICS.

1. Run the following commands in the / directory. Create a directory to run NICS components from.

```
> mkdir -p /opt/nics/deploy
```

2. Navigate to the collab-feed-manager directory, and check the contents

```
> cd /data/nics-5.6/archive/processor/collab-feed-manager/target

> ll
total 29572
drwxr-xr-x 2 nics nics     4096 Jan 14 16:04 ./
drwxr-xr-x 3 nics nics     4096 Jan 14 16:04 ../
-rw-r--r-- 1 nics nics    30960 Jan 14 16:04 collab-feed-manager-X.X.jar
-rw-r--r-- 1 nics nics 30237871 Jan 14 16:04
collab-feed-manager-X.X-jar-with-dependencies.jar
```

1. Copy the files to the nics deploy directory

```
# Copy collab-feed-manager to nics deploy
> cp collab-feed-manager-X.X.X.tar.gz /home/nics/deploy/collab-feed-manager/
```

2. Untar the file

```
# Untar all files to nics deploy
> tar -xvzf collab-feed-manager-X.X.X.tar.gz .
```

3. Configure properties file
    a. Open the properties file:

```
> vi /opt/nics/deploy/collab-feed-manager/collabFeedManager.properties
```

    b. Configure: dbHost should point to a database vm instance used by the MapServer

```
dbName=nics
dbUsername=postgres
dbPassword=postgrespassword
dbHost=123.45.55.66
dbPort=5432
geoserverUrl=http://111.22.33.44:8080/geoserver/rest
geoserverUsername=admin
geoserverPassword=geoserverpassword
workspaceName=dev.nics.collaborationfeed #this needs to be created in
geoserver
dataStoreName=dev.nics #this need to be created in geoserver
syncInterval=3600
kmlPublishInterval=10
kmlTimeout=120
kmlFilepath=/var/www/collabfeedkml/#not currently implemented
collabSrcUrl=rabbitmq://localhost:5672?amqExchange=amq.topic&amqExchangeTyp
e=topic&requestedHeartbeat=0&routingKey=iweb.NICS.collabroom.#.&noAck=false
&user=guest&password=guest&msgPersistent=false&msgContentType=text
kmlUrl=http://hostname/collabfeedkml/
```

4. Start the component

```
# Copy collab-feed-manager to nics deploy
> cd /opt/nics/deploy/collab-feed-manager/
> nohup ./start.sh > logs/collab-feed-manager.log &
```

NICS Configuration Files

/nics-web/web-app/src/main/config/core.properties

```
endpoint.rest=http://localhost:8080/em-api/v1
token.timeout = 1810000
geoserver.endpoint=https://<mapserver_hostname>/geoserver
maps.bing.apikey=GET_MICROSOFT_BING_API_KEY

rabbitmq.hostname=localhost
rabbitmq.username=guest
rabbitmq.userpwd=guest
rabbitmq.exchange.name=amq.topic
rabbitmq.maxconntries=
rabbitmq.failover.hostname=
rabbitmq.bindingkeys=iweb.#
rabbitmq.msgver=1.2.3

feedback.topic=iweb.nics.alert.email
# comma separated list of email addresses
feedback.email.to=recipients
feedback.email.subject=Feedback Report from

#Beginning the property with the keyword private prevents the property from being
shared with the client
#private.key = secret

feedback.topic=iweb.nics.alert.email
# comma separated list of email addresses
feedback.email.to=recipients
feedback.email.subject=Feedback Report from
```

/em-api/api-rest-service/src/main/config/em-api.properties

```
em.api.exchange.name=amq.topic
em.api.rabbitmq.hostname=<hostname>
em.api.rabbitmq.bindingkeys=LDDRS.notifications.forms.#
em.api.rabbitmq.username=<username>
em.api.rabbitmq.userpwd=<password>
em.api.rabbitmq.msgver=1.2.3
em.api.db.get.maxrows=500
em.api.cache.user.refreshminutes=60
em.api.service.incident.foreverid=800
em.api.resource.chat.stalemsg.factor.mins=15
em.api.resource.chat.stalemsg.factor.string=*STALE>
em.api.resource.incident.getall.accessibleOnly=false

# File Upload Properties
em.api.service.file.upload.path=/opt/data/nics/upload/
em.api.service.file.upload.url=https://<hostname>/static/

# SR Report Properties
em.api.resource.report.sr.storagepath=<path>
#em.api.resource.report.sr.url=<path>
em.api.resource.report.sr.path=https://<hostname>/<path>

# Export Data Layer Properties
em.api.service.export.kmlExportURL=/<workspace>/wms?request=GetMap&service=wms&styles=
collabRoomStyle&format_options=SUPEROVERLAY:false;KMPLACEMARK:false;KMSCORE:40;KMATTR:
true;&height=1024&width=1024&format=application/vnd.google-earth.kmz&transparent=false
&version=1.1.1&srs=EPSG:4326
em.api.service.export.mapserverURL=<geoserver>
em.api.service.export.mapserverUsername=<username>
em.api.service.export.mapserverPassword=<password>
em.api.service.export.collabroomStore=<store>
em.api.service.export.workspaceName=<workspace>

# Import Data Layer Properties
em.api.service.import.shapefileWorkspace=<workspace>
em.api.service.import.shapefileStore=<store>

# MDT Properties
em.api.service.mdt.topic=NICS.mdt.gml
em.api.service.mdt.nicsSchemaLocationURI=<schema>
em.api.service.mdt.wfsSchemaURI=<xsd>
em.api.service.mdt.wfsServiceURI=<wfsService>
em.api.service.mdt.typeName=phi_mdt
#em.api.service.mdt.srsName=EPSG:3857
em.api.service.mdt.srsName=EPSG:4326

#FROM user account on registration emails
em.api.user.alert.email=<one email address>
```

/nics-tools/sso-tools/src/main/resources/sso-tools.properties

```
openam.creator.user=encrypteduserhere
openam.creator.pass=encryptedpasswordgoeshere
algorithm=xxxxxxxxxxxxx

# will combine to make the openam url: http://identity_server_hostname:80/openam
openam.protocol=http
openam.host=identity_server_hostname
openam.port=8080
openam.path=openam
```

/nics-tools/openam-tools/src/main/resources/openam.properties

```
openam.url=https://identity_server_hostname/openam
auth.openam.url=https://identity_server_hostnam/openam/identity/authenticate?
attr.openam.url=https://identity_server_hostnam/openam/identity/attributes?
attr.openam.subject=subjectid

cookie.domain=domain
cookie.path=/

auth.openam.service.url=https://identity_server_hostnam/openam
auth.openam.rest.create=/identity/create?
auth.openam.rest.update=/identity/update?
auth.openam.rest.delete=/identity/delete?
auth.openam.rest.search=/identity/search?
auth.openam.rest.authenticate=/identity/authenticate?
auth.openam.rest.token.validate=/identity/isTokenValid?

auth.openam.identity.name=identity_name
auth.openam.identity.attribute.names=identity_attribute_names
auth.openam.identity.attribute.values.prefix=identity_attribute_values_
auth.openam.identity.attribute.values.sn=sn
auth.openam.identity.attribute.values.cn=cn
auth.openam.identity.attribute.values.userpassword=userpassword
auth.openam.identity.attribute.values.mail=mail
auth.openam.identity.attribute.values.active=inetuserstatus
auth.openam.identity.attribute.values.firstname=givenname
auth.openam.identity.auth.username=username
auth.openam.identity.auth.password=password
auth.openam.identity.realm=identity_realm
auth.openam.identity.type=identity_type
auth.openam.token.admin=admin
auth.openam.token.id=tokenid
auth.openam.user.realm=/
auth.openam.values.active=Active
auth.openam.values.inactive=Inactive

auth.openam.user.creator.username=username
auth.openam.user.creator.password=password
auth.openam.user.default.temp.password=password
algorithm=algorithm

# Comma delimited list of root URLs of endpoints protected by OpenAM
openam.protected.endpoints=https://hostname_mapserver/geoserver
```

/nics-tools/sso-tools/src/main/resources/sso-tools.properties

```
openam.url=https://identity_server_hostname:443/openam
openam.rest.create=/identity/create?
openam.rest.update=/identity/update?
openam.rest.delete=/identity/delete?
openam.rest.search=/identity/search?
openam.rest.authenticate=/identity/authenticate?
openam.rest.token.validate=/identity/isTokenValid?

openam.identity.name=identity_name
openam.identity.attribute.names=identity_attribute_names
openam.identity.attribute.values.prefix=identity_attribute_values_
openam.identity.attribute.values.sn=sn
openam.identity.attribute.values.cn=cn
openam.identity.attribute.values.userpassword=userpassword
openam.identity.attribute.values.mail=mail
openam.identity.attribute.values.active=inetuserstatus
openam.identity.attribute.values.firstname=givenname
openam.identity.auth.username=username
openam.identity.auth.password=password
openam.identity.realm=identity_realm
openam.identity.type=identity_type
openam.token.admin=admin
openam.token.id=tokenid
openam.user.realm=/
openam.values.active=Active
openam.values.inactive=Inactive

openam.default.uri.param=uri
openam.default.uri.value=realm%3D/%26service%3DldapService
algorithm=algorithm
openam.user.mach.username=username
openam.user.mach.password=password

openam.user.creator.username=username
openam.user.creator.password=password

openam.user.default.temp.password=password

default.realm=/
default.locale=en_US
default.store=DataStore
```

Import Data Layers Configuration

Data Imports – ArcGisRest, GPX, GeoJson, KML

1. Create the following folders in /opt/data/nics/upload/ on the Data VM
    a. gpx
    b. arcgisrest
    c. geojson
    d. kml
    e. kmz
2. Insert a new datasource entry into the datasource table where the internalurl is the accessible location of the previously created folders and the corrected datasourcetype
    a. https://<your hostname>/static/upload/gpx/

3. The directories on the Data VM should be mounted to the Web VM so that they are web accessible from the UI.
4. Update apache config to match the internal URL inserted in the database