Sentinels Rolling Archive User Access, Operations, Maintenance and Evolutions

DHuS external DB - Installation and Configuration Manual

Role/Title	Name	Signature	Date
Author	Copernicus IVV team		19/03/2019
Verified/Approved	Data Hub Service Manager		19/03/2019

Change register

Version/Rev.	Date	Change	Reason
draft	01/09/2017	--	Draft version of the document
1.0	09/10/2017	The following sections have been update: • Reference Documents • Installation and Configuration procedures (PostgreSQL and DHuS configuration description improvements) The following sections have been added: • System Overview • Migration of embedded HSQLDB to external PostgreSQL DB • Migration of embedded Solr to external Solr • Export/Import of external PostgreSQL DB • Export/Import of external Solr	First issue
1.1	29/01/2018	The following sections have been update: • Reference Documents • Solr section improved with step obtain the correct managed-schema file to be used during installation. • DHuS start.sh section improved with configuration to add in order to allow the correct cache replication. The following section has been added: • PostgreSQL post-installation section containing the procedure to manually delete an index not supported by PostgreSQL database.	Minor revisions

1.2	26/04/2016	The following sections have been update: • DHuS start.sh section improved with configuration to add in order to allow the correct cache replication.	Minor revisions
1.3	19/03/2019	Migration Procedures section has been removed	Minor revisions

Table of Contents

Table of Figures

Table Index

1. Introduction

1.1 Scope

This Installation and Configuration Manual [ICM] applies to DHuS software versions 0.14+ in the externalized DB configuration.

1.2 Purpose

The purpose of this document is to define the basics for installing and configuring the DHuS software with use of externalized DB.

1.3 Structure of the document

Section 1 **Introduction** – This chapter. It contains Purpose and Scope of this document, Document Structure, Acronyms and Abbreviations, Applicable and Reference documents.

Section 2 **System Overview** – It illustrates an overview of DHuS with externalized databases.

Section 3 **Installing and configuring DHuS with externalized DB** – It reports prerequisites and procedures in order to:

- install DHuS in externalized mode
- migrate embedded DHuS databases to external ones
- Export and re-import external databses from different instances

1.4 Acronyms and Abbreviations

Table 1 – Acronyms and Abbreviations

Acronym	Definition
AD	Applicable Document
DB	Database
DHuS	Data Hub Service
EC	European Commission
ESA	European Space Agency
HTTP	Hypertext Transfer Protocol
HW	Hardware
RD	Reference Documents

SW	Software
VM	Virtual Machine

1.5 Reference Documents

Table 2 – Reference Documents

ID	Document Title	Reference
RD-1	DHuS Administration Manual	SPA-COPE-DHUS-UM-001
RD-2	PostgreSQL documentation	https://www.postgresql.org/docs/
RD-3	Solr documentation	http://lucene.apache.org/solr/guide/
RD-4	pgAdmin4 documentation	https://www.pgadmin.org/docs/
RD-5	Zookeeper documentation	https://zookeeper.apache.org/doc/r3.4.10/
RD-6	Java documentation	https://docs.oracle.com

2. System Overview

Since version 0.14.1+ DHuS software supports the externalization of PostgreSQL and Solr databases. The software can be installed on different machines pointing to the same external databases.

The DHuS deployment in externalized mode (named "Scalability 2.0") is finalized to have several DHuS instances acting as one to share the user load and the products information. The deployment in externalized mode is completely transparent to the user.

This document covers all aspects of DHuS deployment in externalized mode, including all installation prerequisites needed and how to deploy the services in hosting infrastructure.

A detailed description of the system context overview, i.e. the list of necessary actors for the deployment of DHuS in this modality, is depicted in Section 2.1 of DHuS Administration Manual [RD-1].

FIG shows a representation of the deployment in Scalability 2.0 mode:

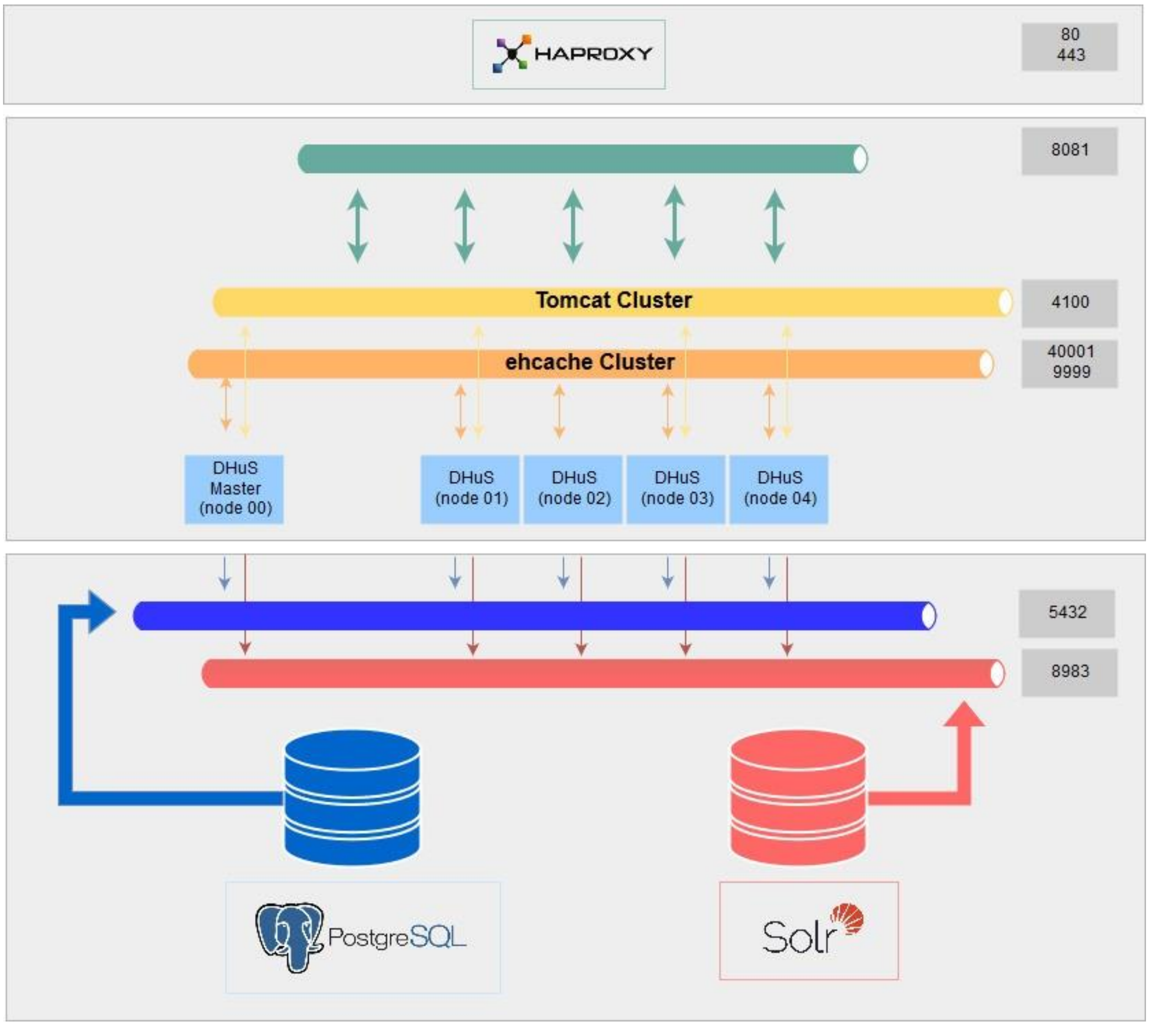

Figure 1 - DHuS Scalability 2.0 deploy

3. Installing and configuring DHuS with externalized DB

This Section covers the installation and the configuration of the DHuS with externalized DB, consisting of the following components:

- DHuS 0.14.1-3 or above
- PostgreSQL
- Solr
- Zookeeper

Installation prerequisites and configuration procedures are described in the following paragraphs.

3.1 Installation prerequisites

In this section, we provide a list of installation prerequisites that must be fulfilled before the installation can begin.

3.1.1 Operating System requirements

The following Operating Systems, along with their minimum hardware requirements, support the software installation:

- Red Hat Enterprise Linux version 6.9 or greater
- CentOS Linux version 7.2 or greater
- Debian 8.6 or greater

3.1.2 Software requirements

To install and run the DHuS software with externalized DB, the following packages must be downloaded and installed on the systems:

- PostgreSQL (with uuid-ossp extension) version 9.6
- management tool for PostgreSQL (pgAdmin4)
- Solr version 5.5.4
- Zookeeper 3.4.10
- Java 8 or above

For this installation scenario, sufficient hard disk space should be taking into account for the base installation and for data stored in Databases.

3.1.3 Networking requirements

Network connection to port 5432 (PostgreSQL), port 8983 (Solr) and port 2181 (Zookeeper) must be established between the system where the Front-End DHuS is installed and the systems where the S/W are installed. See Figure 1 in Section 2.

3.2 Installation and Configuration procedures

To perform all steps below, the credentials of root user must be known (or in alternative a user with sudoer privilege).

Machine configuration for starting all services described in the following paragraphs at System boot is recommended.

3.2.1 PostgreSQL

Step ID	User	Tasks and Commands

1.	root	As root user, install PostgreSQL 9.6 software, following the standard installation procedure for your Operating System.
		`[e.g. for CentOS 7]:`
		`> netstat rpm -Uvh https://yum.postgresql.org/9.6/redhat/rhel-7-x86_64/pgdg-centos96-9.6-3.noarch.rpm`
		`> yum install postgresql96 postgresql96-contrib postgresql96-server`

2.	root	Create a new PostgreSQL database cluster specifying the directory where the database cluster should be stored. `> /usr/pgsql-9.6/bin/postgresql96-setup initdb --pgdata="PGDATA_path"` Where: • `PGDATA_path` is the directory where database cluster will be stored. Please note that the database file can reach a big size, so configure this directory in a file system partition with enough space ("`/data`" folder is recommended).
3.	root	Start the PostgreSQL 9.6 service. (Please note that the exact command depends upon your Operating System.)
4.	root	Verify that PostgreSQL port 5432 is listening. This action can be performed using the netstat command: `> netstat -an | grep 5432` A LISTEN output is expected.

5.	root	Set the access privileges for PostgreSQL database. PostgreSQL can be accessed from the localhost or from other external systems. Please find below the related configuration steps to be performed.
		<u>Configuration to access PostgreSQL from the localhost</u>
		a) Open the `pg_hba.conf` PostgreSQL configuration file in charge of client authentication management.
		`> vi /<PGDATA_path>/lib/pgsql/9.6/data/pg_hba.conf`
		Please note that the exact path depends upon your Operating System.
		b) Set the row related to IPv4 local connections as the following:
		`# TYPE  DATABASE  root`

		`USER        ADDRESS            METHOD`
		`# IPv4 local connections:`
		`host    all            all         127.0.0.1/32        trust`
		<u>Configuration to access PostgreSQL from other external systems</u>
		a) Open the `pg_hba.conf` PostgreSQL configuration file in charge of client authentication management.
		`> vi /<PGDATA_path>/lib/pgsql/9.6/data/pg_hba.conf`
		Please note that the exact path depends upon your Operating System.
		b) Add a row related to IPv4 local connections as the following:
		`# TYPE  DATABASE    USER        ADDRESS            METHOD`
		`# IPv4 local connections:`
		`host    all            all             <ACCESSING_SUBNET>/24 trust`
		c) Open the `postgresql.conf` PostgreSQL configuration file in charge of all general configuration.
		`> vi /<PGDATA_path>/lib/pgsql/9.6/data/postgresql.conf`
		Please note that the exact path depends upon your Operating System.
		d) Set the row related to IP listen addresses as needed.
		`listen_addresses = '<Remote_System_IP>'`
		`# or eg:`
		`listen_addresses = '*'`
		e) Verify that the Network or System firewall is not blocking access to port 5432, and eventually allow the access.
		(Please note that the exact command depends upon your Operating System.)

6.	root	Restart the PostgreSQL service. (Please note that the exact command depends upon your Operating System.)

7.	root	[OPTIONAL]
		Database Maintainer user “postgres” has no password by default. If a password if wanted, it is possible to set it.
		a) Change the login session's owner in order to use the “postgres” user
		`> su - postgres`
		b) As postgres user, set the password:
		`> psql`
		`> ALTER USER postgres WITH PASSWORD '<new_password>';`
		`> \q`
		c) Exit from postgres user

8.	root	[OPTIONAL]
		Enable the PostgreSQL error reporting and logging adding related configuration parameters in the `postgresql.conf` configuration file.
		a) Open the `postgresql.conf` PostgreSQL configuration file.
		`> vi /<PGDATA_path>/lib/pgsql/9.6/data/postgresql.conf`
		Please note that the exact path depends upon your Operating System.
		b) Add settings as follow at the end of the file:
		`# Add settings for extensions here`
		`log_destination = 'csvlog'`
		`log_directory = 'pg_log'`
		`logging_collector = on`
		`log_filename = 'postgres-%Y-%m-%d_%H%M%S'`
		`log_rotation_age = 1d`
		`log_rotation_size = 1GB`
		`log_min_duration_statement = 250ms`
		`log_checkpoints = on`
		`log_connections = on`
		`log_disconnections = on`
		`log_lock_waits = on`
		`log_temp_files = 0`
		Please refer to PostgreSQL configuration [RD-2] for details about parameters definition.

9.	root	[OPTIONAL]
		Configure PostgreSQL for having high performance adding related configuration parameters in the `postgresql.conf` configuration file.
		a) Open the `postgresql.conf` PostgreSQL configuration file.
		`> vi /<PGDATA_path>/lib/pgsql/9.6/data/postgresql.conf`
		Please note that the exact path depends upon your Operating System.
		b) Add settings as follow at the end of the file:
		`max_connections = 1000`
		`shared_buffers = 8GB`
		`effective_cache_size = 96GB`
		`work_mem = 33554kB`
		`maintenance_work_mem = 2GB`
		`min_wal_size = 1GB`
		`max_wal_size = 2GB`
		`checkpoint_completion_target = 0.9`

		`wal_buffers = 16MB`
		`default_statistics_target = 100`
		`fsync = on`
		Please refer to PostgreSQL configuration [RD-2] for details about parameters definition.

10.	Standard user	Create an empty database using a management tool for PostgreSQL (e.g. pgAdmin). [e.g pgAdmin4] a) Create a new Server. 1. On the Graphic tool, right click on Servers, and select Create and then Server; 2. On the pop-up window: - In the General tab enter the server name; - In the Connection tab enter: • the hostname/address of the PostgreSQL instance • the Port where PostgreSQL instance is listening to (the default one is default 5432) • the Password of user “postgres” if set in previous step (see Step 7 of this procedure), or leave it empty. 3. Click on Save. As result, the new Server is added to the Servers list. b) Create the user that will be the owner of the database. 1. Expand the Server just created, and right click on “Login/Group Roles” > Create > Login/Group Role; 2. On the pop-up window: - In the General tab enter the login user name; - In the Definition tab enter the password; - In the Privileges tab, enable all privileges; - In the Membership tab, use the drop-down listbox, select postgres and thick the box; 3. Click on Save. As result, the new user is added to the “Login/Group Roles” list. c) Create the database. 1. On the Server previously created, right click on “Databases” > Create > Database; 2. On the pop-up window: - In the General tab enter: • the Database name; • set the Owner field using the user created at point b)

		using the drop-down list; 3. Click on Save. As result, the new Database is added to the Databases list. d) Add the uuid-ossp extension to the Database just created. 1. Expand the Database just created, and right click on “Extensions” > Create > Extension; 2. On the pop-up window: - In the General tab, use the drop-down listbox and select “uuid-ossp” extension; 3. Click on Save. As result, the new extension is added to the Extensions list. Please refer to pgAdmin4 documentation for further information on how to use the tool [RD-4]. The creation and configuration of the empty database is finished.

3.2.2 ZooKeeper

Step ID	User	Tasks and Commands
1.	Standard user	Access the machine where ZooKeeper software will be installed and create a dedicated directory. `> mkdir /path/to/ZOOKEEPER_INST_DIR` Installation of ZooKeeper on /data folder is recommended.

2.	Standard user	Install the ZooKeeper 3.4.10 software in the dedicated directory, following the standard installation procedure for your Operating System.
		`> cd /path/to/ZOOKEEPER_INST_DIR`
		`> wget http://mirror.nohup.it/apache/zookeeper/zookeeper-3.4.10/zookeeper-3.4.10.tar.gz`
		`> tar xvzf zookeeper-3.4.10.tar.gz`

3.	Standard user	Configure ZooKeeper setting the path for the `dataDir` folder (i.e. the location where ZooKeeper will store the in-memory database snapshots and logs).
		a) Create a configuration file and open it.
		`> cd <ZOOKEEPER_INST_DIR>/zookeeper-3.cd /4.10/conf`
		`> cp zoo_sample.cfg zoo.cfg`
		`> vi zoo.cfg`
		b) Assign to the `dataDir` variable the following path:
		`dataDir=<ZOOKEEPER_INST_DIR>/zookeeper-3.4.10/data`

4.	Standard user	Start ZooKeeper service.
		`> cd <Zookeeper_INST_DIR>/zookeeper-3.4.10/bin`
		`> ./zkServer.sh start`

5.	Standard	Verify that ZooKeeper port 2181 is listening.

	user	This action can be performed using the netstat command: `> netstat -an | grep 2181` A LISTEN output is expected.

3.2.3 Solr

For the sake of DHuS deployment in externalized mode (with PostgreSQL and Solr), the following procedure details the Solr installation in Cloud mode in order to use ZooKeeper as orchestrator.

Please refer to Solr documentation for Solr installation in Standalone mode [RD-3].

Step ID	User	Tasks and Commands
1.	Standard user	Access the machine where Solr software will be installed and create a dedicated directory. `> mkdir /path/to/SOLR_INST_DIR` Installation of Solr on /data folder is recommended.

2.	Standard user	Install the Solr 5.5.4 software in the dedicated directory, following the standard installation procedure for your Operating System.
		`> cd /path/to/SOLR_INST_DIR`
		`> wget https://archive.apache.org/dist/lucene/solr/5.5.4/solr-5.5.4.zip`
		`> unzip solr-5.5.4.zip`

3.	Standard user	Download the CONF.zip file, containing useful configuration needed for the Solr and DHuS configuration, available at:
		https://github.com/SentinelDataHub/DataHubSystem/raw/gh-pages/documents/CONF.zip
		Create a directory in which unzip the provided CONF.zip file.cd /data
		`> mkdir /path/to/CONF_DIR`
		`> cd /path/to/CONF_DIR`
		`> unzip CONF.zip`

4.	Standard user	Copy the JTS library into Solr lib folder, in order to allow indexing polygons with Solr.
		`> cd /path/to/SOLR_INST_DIR/solr-5.5.4/server/solr-webapp/webapp/WEB-INF/lib`
		`> cp /path/to/CONF_DIR/CONF/solr/lib/jts-1.13.jar .`

5.	Standard user	Insert the correct managed-schema file in the CONF directory. The managed-schema file is not automatically generate by DHuS in case of Externalized configuration. SolR creates the file automatically in DHuS embedded mode. It allows SolR to determine the type of information to retrieve and present.
		a) Backup the existing managed-schema file in the CONF directory.
		`> cd /path/to/CONF_DIR/solr/conf`
		`> mv managed-schema managed-schema.bak`
		b) Start a DHuS instance in embedded mode and copy the managed-schema file automatically generated by DHuS into the matching directory of the SolR externalized instance.

		`> cd <EMBEDDED_DHUS_INSTALL_DIR>/var/solr/dhus/conf/`
		`> cp managed-schema /path/to/temporary/director`
		`> cp /path/to/temporary/directory/managed-schema /path/to/CONF_DIR/solr/conf`

6.	Standard user	Configure Solr modifying the solr.in.sh file.
		a) Open the configuration file.
		`> cd /path/to/SOLR_INST_DIR/solr-5.5.4/bin`
		`> vi solr.in.sh`
		b) Assign to the `ZK_HOST` variable the address of the ZooKeeper service to use (refer to Section 3.2.2 for the installation of Zookeeper service):
		`> ZK_HOST="<ZOOKEEPER_SERVICE_IP>:2181/solr"`
		Please note that when the `ZK_HOST` variable is set, Solr will launch in Cloud mode.
		c) Assign to the `SOLR_HOST` variable the address of the machine in which Solr is installed (localhost):
		`> SOLR_HOST="127.0.0.1"`

7.	Standard user	Use the Solr's `zkcli.sh` Command Line Utility to create a chroot path in ZooKeeper and to upload specific configuration contained in the provided CONF.tar file. Solr's `zkcli.sh` has command line arguments specific to dealing with Solr data in ZooKeeper.
		`> cd /path/to/SOLR_INST_DIR`
		`> solr-5.5.4/server/scripts/cloud-scripts/zkcli.sh -zkhost <ZOOKEEPER_SERVICE_IP>:2181 -cmd makepath /solr`
		`> solr-5.5.4/server/scripts/cloud-scripts/zkcli.sh -zkhost <ZOOKEEPER_SERVICE_IP>:2181/solr -cmd upconfig –confdir /path/to/CONF_DIR/CONF/solr/conf -confname data_driven_schema_configs`

8.	Standard user	Start Solr service.
		`> cd /<SOLR_INST_DIR>/solr-5.5.4`
		`> bin/solr start`

9.	Standard user	Verify that Solr port 8983 is listening. This action can be performed using the netstat command: `> netstat -an | grep 8983` A LISTEN output is expected.
10.	Standard user	If Zookeeper and Solr are installed on different machines, verify that the Network or System firewall is not blocking communication between the systems, and eventually allow the access. The exact commands depend upon your Operating System.
11.	Standard user	Create a Collection on Solr. a) Connect to http://localhost:8983/solr in order to use the Solr's Web Browser User Interface. b) On the Browser Solr page, click on "Try New UI" on top right. c) On the new page, click on Collections and then "Add Collection".

		d) On the pop-up window enter: • name: the collection name • config set: select "data_driven_schema_configs" using the drop-down list • numShards: 1 • replicationFac: 1 e) Click on "Add Collection" button. As result, the new Collection is added to Solr.

3.2.4 DHuS

DHuS with externalized DB (PostgreSQL + Solr) can be deployed with one single node or with multiple nodes; in the latter case Tomcat session and Ehcache replication mechanisms are needed (refer to Sections 3.2.4.4 and 3.2.4.5 to configure these methods). Please refer to Sections 3.2.4.1, 3.2.4.2 and 3.2.4.3 to allow DHuS software to use external Solr and PostgreSQL instances.

When DHuS instance/s have been installed and configured properly, it/they can be started accordingly.

Please refer to DHuS documentation for details about DHuS software [RD-1].

3.2.4.1 Installation

Step ID	User	Tasks and Commands
1.	Standard user	Install the DHuS software following the procedure in the "DHuS Administration Manual" (RD-1).

2.	Standard user	Depending from the version of Java installed on your machine, copy the proper Postgres Jar file to the DHuS lib directory.
		• For Java 7:
		`> cp <CONF_INST_DIR>/CONF/DHuS/lib/postgresql-9.4.1212.jre7.jar <DHuS_INST_DIR>/lib/`
		• For Java 8:
		`> cp <CONF_INST_DIR>/CONF/DHuS/lib/postgresql-9.4.1212.jar <DHuS_INST_DIR>/lib/`

3.2.4.2 start.sh

Configure the start.sh file in order to allow DHuS software to use external PostgreSQL database. For all other configurations and customizations regarding DHuS start.sh file, please refer to RD-1.

Step ID	User	Tasks and Commands

1.	Standard user	Modify the line:
		`-cp "etc:lib/*" fr.gael.dhus.DHuS &`
		with
		`-cp "etc:lib/postgresql-9.4.1212.jar:lib/*" fr.gael.dhus.DHuS &`

2.	Standard user	Insert the following line in the java options in order to allow clients to invoke methods on the remote object (needed for the cache replication, see Section 3.2.4.5): `-Djava.rmi.server.hostname=<DHuS_VM_IP>               \`

		Please note that: • `DHuS_VM_IP` is the IP address of the VM hosting the DHuS service. Please also note that this step shall be executed for every DHuS node.
3.	Standard user	Insert the following line in the java options in order to specify the port where peers access each other's cache (needed for the cache replication, see Section 3.2.4.5): `-Djava.rmi.activation.port=40001          \` Please note that this step shall be executed for every DHuS node.

4.	Standard user	Insert the following lines in the java options in order to set Java RMI runtime time-out times (needed for the cache replication, see Section 3.2.4.5):
		`-Dsun.rmi.transport.connectionTimeout=15000            \`
		`-Dsun.rmi.transport.tcp.handshakeTimeout=15000         \`
		`-Dsun.rmi.transport.tcp.responseTimeout=15000          \`
		`-Dsun.rmi.transport.tcp.readTimeout=15000              \`
		Please refer to Java documentation [RD-6] for details about parameters definition.

3.2.4.3 dhus.xml

The aim of this procedure is to configure the dhus.xml file in order to allow DHuS software to use external Solr and PostgreSQL instances. For all other configurations and customizations regarding DHuS, please refer to RD-1.

Step ID	User	Tasks and Commands

1.	Standard user	Modify the Solr setting (line starting with `<search:solr`) with the proper configuration to use Solr in Standalone Mode or to use Zookeeper to manage the Solr instance.
		Please note that:
		• The Standalone setting can be used with Solr installed in both Standalone and Cloud Modes;
		• the Zookeeper use is possible only with Solr installed in Cloud Mode.
		Standalone Solr setting
		`[if Solr installed in Standalone Mode]    <search:solrStandalone serviceURL= http://<SOLR_SYSTEM_IP>:PORT/solr/<SOLR_CORE_NAME>/>`
		or
		`[if Solr installed in Cloud Mode]    <search:solrStandalone serviceURL= http://<SOLR_SYSTEM_IP>:PORT/solr/<SOLR_COLLECTION_NAME>/>`
		The `serviceURL` parameter should contain the IP of the system where Solr is installed, the port and the Solr Core/Collection name, according to the installation mode.
		Zookeeper Solr setting (only with Solr installed in Cloud Mode)
		`<search:solrCloud`

		`zkHosts="<ZOOKEEPER_SYSTEM_IP>:PORT/solr/<SOLR_COLLECTION_NAME>"/>` The `zkHosts` parameter should contain the interested Zookeeper IP, the port and the Solr Collection name.

2.	Standard user	Modify the Database setting (line starting with `<system:database`) with the proper configuration in order to use external PostgreSQL database.
		`<system:database JDBCDriver="org.hsqldb.jdbcDriver"`
		`hibernateDialect="org.hibernate.dialect.PostgreSQLDialect"`
		`JDBCUrl="jdbc:postgresql://<POSTGRES_SYSTEM_IP>:PORT/DATABASE_NAME"`
		`login="DATABASE_OWNER_NAME"`
		`password="DATABASE_OWNER_PASSWORD" />`
		Please note that:
		• `JDBCUrl` parameter should contain the PostgreSQL database to be used;
		• `DATABASE_NAME` is the name of the PostgreSQL database to be used present in the external Postgres instance;
		• `DATABASE_OWNER_NAME` is the user name of the database's owner to connect the database;
		• `DATABASE_OWNER_PASSWORD` is the password of the database's owner.

3.2.4.4 server.xml

The aim of the following procedure is to set up a Tomcat cluster in order to allow the session replication among DHuS nodes; this requires the modification of the original server.xml file, coming from the DHuS distribution. In case of DHuS deployed with only one node, the default server.xml file should be used.

This procedure assumes that all DHuS instances (nodes), part of the Tomcat cluster, are stopped. The following procedure must be executed separately for every DHuS node.

Step ID	User	Tasks and Commands

1.	Standard user	Access the DHuS `etc` configuration folder in the machine where the selected DHuS instance (node) has been installed and perform a backup of the original `server.xml` file.
		`> cd <DHuS_INST_DIR>/etc`
		`> cp server.xml server.xml.bk`

2.	Standard user	Open the `server.xml` file and add the Cluster section in the `<Host></Host>` element to enable clustering (refer to Step 4 of this procedure for an example containing the Cluster configuration): `<Cluster className="org.apache.catalina.ha.tcp.SimpleTcpCluster"/>`

3.	Standard user	Inside the `StaticMembershipInterceptor` element specify the other static members of the cluster (excluding itself) listen for cluster messages, using the following syntax:
		`<Member`
		`className="org.apache.catalina.tribes.membership.StaticMember"`
		`host="172.30.246.46" port="4100"`
		`uniqueId="{0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,2}"/>`

<table>
<tr>
<td></td>
<td></td>
<td>

Please note that:

- `host` is the IP address of the static member listens for cluster messages.
- `port` is the Tomcat port of the static member
- `uniqueId` is a universally uniqueId for this static member. The values must be 16 bytes array in string form.

</td>
</tr>
<tr>
<td>4.</td>
<td>Standard user</td>
<td>

Please find below an example of Cluster configuration to allow Tomcat session replication; two static members are detailed (i.e. the total cluster is composed by three static members). Using the below configuration will enable session replication among all the other members in the cluster using the `DeltaManager` to replicate session deltas.

```
<Cluster channelSendOptions="8" channelStartOptions="3"
className="org.apache.catalina.ha.tcp.SimpleTcpCluster">

                     <Manager
className="org.apache.catalina.ha.session.DeltaManager"
expireSessionsOnShutdown="false"
notifyListenersOnReplication="true"/>

                     <Channel
className="org.apache.catalina.tribes.group.GroupChannel">

                              <Sender
className="org.apache.catalina.tribes.transport.ReplicationTransmit
ter">

                                      <Transport
className="org.apache.catalina.tribes.transport.nio.PooledParallelS
ender"/>

                              </Sender>

                              <Receiver address="172.30.246.20"
autoBind="0"
className="org.apache.catalina.tribes.transport.nio.NioReceiver"
maxThreads="6" port="4100" selectorTimeout="5000"/>

                              <!-- <Interceptor
className="com.dm.tomcat.interceptor.DisableMulticastInterceptor"
/>-->

                              <Interceptor
className="org.apache.catalina.tribes.group.interceptors.TcpPingInt
erceptor" staticOnly="true"/>

                              <Interceptor
className="org.apache.catalina.tribes.group.interceptors.TcpFailure
Detector"/>

                              <Interceptor
className="org.apache.catalina.tribes.group.interceptors.StaticMemb
ershipInterceptor">

                                           <Member
className="org.apache.catalina.tribes.membership.StaticMember"
host="172.30.246.46" port="4100"
uniqueId="{0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,2}"/>

                                           <Member
```

</td>
</tr>
</table>

		className="org.apache.catalina.tribes.membership.StaticMember"
		host="172.30.246.11" port="4100"
		uniqueId="{0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,3}"/>
		</Interceptor>
		<Interceptor
		className="org.apache.catalina.tribes.group.interceptors.MessageDispatch15Interceptor"/>
		</Channel>
		<Valve
		className="org.apache.catalina.ha.tcp.ReplicationValve" filter=""/>
		<Valve
		className="org.apache.catalina.ha.session.JvmRouteBinderValve" />
		<ClusterListener
		className="org.apache.catalina.ha.session.ClusterSessionListener"/>
		</Cluster>

3.2.4.5 dhus_ehcache.xml

The aim of the following procedure is to set up a Replicated caching mechanism using Ehcache and Java RMI (Remote Method Invocation) providing remote communication between DHuS instances (nodes); this requires the overwriting of the original dhus_ehcache.xml file, present as default in the DHuS distribution. Ehcache has the notion of a group of caches acting as a replicated cache; each of the caches is a peer to the others. Ehcache comes with a built-in RMI-based distribution system.:

In case of DHuS deployed with only one node, the default dhus_ehcache.xml file should be used.

This procedure assumes that all DHuS instances (nodes) are stopped. The following procedure must be executed separately for every DHuS node.

Step ID	User	Tasks and Commands

1.	Standard user	Access the DHuS `etc` configuration folder in the machine where the selected DHuS instance (node) has been installed and create a copy of the dhus_ehcache_distributed.xml file. The new file (named dhus_ehcache.xml) will be the Ehcache configuration file that will overwrite the default one.
		`> cd <DHuS_INST_DIR>/etc`
		`> cp dhus_ehcache_distributed.xml dhus_ehcache.xml`

2.	Standard user	Open the `dhus_ehcache.xml` file and set the properties of `cacheManagerPeerProviderFactory` attribute in order to use manual peer discovery, as follows:
		<cacheManagerPeerProviderFactory
		class="net.sf.ehcache.distribution.RMICacheManagerPeerProviderFactory"
		properties="peerDiscovery=manual, rmiUrls=<RMI_URLS>"
		propertySeparator=","/>

		Manual peer configuration requires the IP address and port of each listener to be known. Peers cannot be added or removed at runtime. The rmiUrls property is a list of the cache peers of the server being configured. Do not include the server being configured in the list. Refer to Step NN of this procedure for an example containing `cacheManagerPeerProviderFactory` attribute setting.

3.	Standard user	Configure the CacheManagerPeerListener listening for messages from peers to the current CacheManager. In the `cacheManagerPeerListenerFactory` attribute set properties as follows:
		`<cacheManagerPeerListenerFactory`
		`class="net.sf.ehcache.distribution.RMICacheManagerPeerListenerFactory"`
		`properties="hostName=<HOST>,port=<PORT>,remoteObjectPort=<RO_PORT>, socketTimeoutMillis=<TIMEOUT>"/>`
		Please note that:
		• `hostname` is the hostname/IP of the host the listener is running on;
		• `port` is the port the listener listens on;
		• `remoteObjectPort` is the port number on which the remote objects bound in the registry receive calls;
		• `socketTimeoutMillis` is the TCP/IP Socket timeout when waiting on response (i.e. number of seconds client sockets will wait when sending messages to this listener until they give up).

4.		Please find below an example of `cacheManagerPeerProviderFactory` element setting. The `rmiUrls` property contains the list of all cache peers for all cache listener (detailed via IP address and port); cache peers are separated by means of "|" character.
		`<cacheManagerPeerProviderFactory`
		`class="net.sf.ehcache.distribution.RMICacheManagerPeerProviderFactory"`
		`properties="peerDiscovery=manual,`
		`rmiUrls=//172.30.246.46:40001/user_connections|//172.30.246.46:40001/userByName|//172.30.246.46:40001/product_eviction_date|//172.30.246.46:40001/network_download_count|//172.30.246.46:40001/network_download_size|//172.30.246.46:40001/user|//172.30.246.46:40001/json_user|//172.30.246.46:40001/product_count|//172.30.246.46:40001/indexes|//172.30.246.46:40001/product|//172.30.246.46:40001/products|//172.30.246.46:40001/boundariesWKT|//172.30.246.46:40001/current_quotas|//172.30.246.46:40001/user_requests|//172.30.246.46:40001/security_context|//172.30.246.11:40001/user_connections|//172.30.246.11:40001/userByName|//172.30.246.11:40001/product_eviction_date|//172.30.246.11:40001/network_download_count|//172.30.246.11:40001/network_download_size|//172.30.246.11:40001/user|//172.30.246.11:40001/json_user|//172.30.246.11:40001/product_count|//172.30.246.11:40001/indexes|//172.30.246.11:40001/product|//172.30.246.11:40001/products|//172.30.246.11:40001/boundariesWKT|//172.30.246.11:40001/current_quotas|//172.30.246.11:40001/user_requests|//172.30.246.11:40001/security_context"propertySeparator=","/>`

		Please find below an example of `cacheManagerPeerListenerFactory` element setting.
		`<cacheManagerPeerListenerFactory`
		`class="net.sf.ehcache.distribution.RMICacheManagerPeerListenerFactory"`
		`properties="hostName=172.30.246.20,port=40001,remoteObjectPort=9999,socketTimeoutMillis=2000"/>`

3.3 Post-installation Procedures

3.3.1 PostgreSQL

After the DHuS starting process and database schema creation, the following procedure is necessary to allow the correct synchronization of Sentinel-3 product with long footprint values.

Step ID	User	Tasks and Commands
1.	Standard user	[Optional – needed only for PostgreSQL databases running on DHuS 0.14.1-X] Using a management tool for PostgreSQL (e.g. pgAdmin [RD-4]) remove the index "sys_pk_10229" from the "metadata_indexes" table since it is not supported by PostgteSQL database. [e.g pgAdmin4] • Click on Servers > "YOUR_SERVER_NAME" > Databases • Right click on "DATABASE_NAME" and select "Query Tool" • On the new opened panel enter the command: `ALTER TABLE public.metadata_indexes DROP CONSTRAINT sys_pk_10229;` Click on the execute button (button with the "spark" icon)

3.4 Database Export/Import procedures

3.4.1 Export/Import of external PostgreSQL DB

The aim of this procedure is to export and re-import a PostgreSQL database from an original PostgreSQL instance to another one.

This procedure assumes that a non-empty PostgreSQL database is already existing on a PostgreSQL instance.

Step ID	User	Tasks and Commands
1.	root	Access the machine where the PostgreSQL instance containing the database to be exported is present. If not running, start PostgreSQL service. (Please note that the exact command depends upon your Operating System.)
2.	root	Create a working directory on the machine to host the dump file of the exported PostgreSQL database.

		Please note that the working directory should be reachable from all the machines hosting the PostgreSQL services used for the Export/Import of the interested database. Please also note that the dump file can reach a big size, so create the working directory in a file system partition with enough space (at least the size of the database). `> mkdir /path/to/WORKING_DIR`
3.	root	Assign read, write and execution permissions to all users on working directory created at Step 3 of this procedure. `> chmod 777 /path/to/WORKING_DIR`
4.	root	Change the login session's owner and access as `postgres` user. `> su - postgres`

5.	postgres	In the bash shell just opened, access the working directory and create a dump of the interested PostgreSQL database.
		`> cd /path/to/WORKING_DIR`
		`> pg_dump -Fc DATABASE_NAME -f DUMP_FILE_NAME`
		Please note that:
		• `DATABASE_NAME` is the name of the interested database in the PostgreSQL instance;
		• `DUMP_FILE_NAME` is the name of the dump file (it will be a binary file and can have a general extension, e.g. `.file`).

6.	root	Access the machine where the PostgreSQL instance in which import the exported database is present. If not running, start PostgreSQL service. (Please note that the exact command depends upon your Operating System.)
7.	Standard user	In PostgreSQL, create a new empty database in which import the interested database previously exported. This action can be performed using a management tool for PostgreSQL as pgAdmin. Please refer to Section 3.2.1 for the procedure needed to the database creation on pgAdmin.
8.	root	Change the login session's owner and access as `postgres` user. `> su - postgres`

9.	postgres	In the bash shell just opened, access the working directory where the database dump file is present and perform a restore command towards the newly created PostgreSQL database.
		`> cd /path/to/WORKING_DIR`
		`> pg_restore --clean --no-acl --no-owner -d DATABASE_NAME -U DATABASE_USER DUMP_FILE_NAME`
		Please note that:
		• `DATABASE_NAME` is the name of the empty database, created at Step 7 of this procedure, in the PostgreSQL instance in which import the exported database;
		• `DATABASE_USER` is the user name to connect the database. The owner on the

		imported database will be this user. • `DUMP_FILE_NAME` is the name of the dump file (it will be a binary file and can have a general extension, e.g. `.file`). • The option `--no-owner` shall ensure the removal of original database's ownership.

10.	Standard user	Using a management tool for PostgreSQL (e.g. pgAdmin [RD-4]), verify that the database has been imported correctly.
		This action can be performed executing SQL queries in order to retrieve the total number of products or to check that product's information are as expected.
		Please find below two example of SQL queries:
		`> SELECT COUNT(*) from products;`
		`> SELECT * FROM products WHERE uuid='PRODUCT_UUID_IDENTIFIER';`

11.		The PostgreSQL Database Export/Import action is completed; the imported database can now be used by DHuS software configuring accordingly the dhus.xml file (refer to Section 3.2.4).

3.4.2 Export/Import of external Solr

This procedure assumes that a non-empty Solr Core/Collection is already existing on a Solr instance.

The aim of this procedure is to export and re-import a Solr indexes from an original Solr instance to another one.

Step ID	User	Tasks and Commands
1.	Standard user	Access the machine where the Solr service containing the indexes to be exported is present.

2.	Standard user	If running, stop Solr and Zookeeper services.
		`> cd /<SOLR_INST_DIR>/solr-5.5.4`
		`> bin/solr stop`
		`> cd <Zookeeper_INST_DIR>/zookeeper-3.4.10/bin`
		`> ./zkServer.sh stop`

3.	Standard user	Create a working directory on the machine to host the `data` folder containing the exported Solr indexes. Please note that the working directory should be reachable from all the machines hosting the Solr services used for the Export/Import of the interested database. `> mkdir /path/to/WORKING_DIR`

4.	Standard user	Access the directory containing the interested indexes; the path of this directory changes depending on Solr installation mode (Standalone and Cloud Mode). See the Solr documentation [RD-3] for details about the two modes.
		`[Standalone Mode]    > cd /<SOLR_INST_DIR>/solr-5.5.4/server/solr/CORE_NAME`
		`[Cloud Mode]    > cd /<SOLR_INST_DIR>/solr-5.5.4/server/solr/COLLECTION_NAME`

5.	Standard user	Copy the `data` folder. `> cp -r data /path/to/WORKING_DIR/data`

6.	Standard user	Access the machine where the Solr service in which imports the exported indexes is present.
		If not running, start Zookeeper and Solr services.
		`> cd <Zookeeper_INST_DIR>/zookeeper-3.4.10/bin`
		`> ./zkServer.sh start`
		`> cd /<SOLR_INST_DIR>/solr-5.5.4`
		`> bin/solr start`

7.	Standard user	Access Solr Graphical User Interface and create a new Core/Collection in which import the interested indexes previously exported. Please refer to Section 3.2.3 for the procedure needed to the Core/Collection creation on Solr.

8.	Standard user	Stop Zookeeper and Solr services.
		`> cd /<SOLR_INST_DIR>/solr-5.5.4`
		`> bin/solr stop`
		`> cd <Zookeeper_INST_DIR>/zookeeper-3.4.10/bin`
		`> ./zkServer.sh stop`

9.	Standard user	Access the Core/Collection directory selected to host the interested indexes; as already stated, the path of this directory changes depending on Solr installation mode (Standalone and Cloud Mode). See the Solr documentation [RD-3] for details about the two modes.
		`[Standalone Mode]    > cd /<SOLR_INST_DIR>/solr-5.5.4/server/solr/CORE_NAME`
		`     [Cloud Mode]    >  cd /<SOLR_INST_DIR>/solr-5.5.4/server/solr/COLLECTION_NAME`

10.	Standard user	Copy the `data` folder containing the exported indexes in the Core/Collection directory. A `data` folder shall be already present and a backup of it should be performed in order to not lose information contained in this folder.
		`> mv data data_empty`
		`> cp -r /path/to/WORKING_DIR/data .`

11.	Standard user	Start Zookeeper and Solr services.
		`> cd <Zookeeper_INST_DIR>/zookeeper-3.4.10/bin`
		`> ./zkServer.sh start`
		`> cd /<SOLR_INST_DIR>/solr-5.5.4`
		`> bin/solr start`

12.	Standard user	Using the Solr Graphical User Interface, verify that indexes have been imported correctly. This action can be performed executing queries in order to retrieve the total number of products or to check that product's metadata are as expected.
13.	Standard user	The PostgreSQL Database Export/Import action is completed; the imported indexes can now be used by DHuS software configuring accordingly the dhus.xml file (refer to Section

		3.2.4).

<END of DOCUMENT>