CloverDX Server User's Guide



CloverDX Server: User's Guide

This User's Guide refers to CloverDX Server 5.1.x release.

Copyright © 2010-2019 CloverDX by Javlin, a.s. All rights reserved.

Do not copy or distribute without express permission of CloverDX.

www.cloverdx.com

Feedback welcome:

If you have any comments or suggestions for this documentation, please send them by email to support@cloverdx.com.

Consider How to speed up communication with CloverCARE support before contacting the support.

Table of Contents

I. Introduction 1				
1. What is CloverDX Server?	. 2			
2. CloverDX Server Architecture	3			
3. CloverDX Core	4			
4. CloverDX Worker	5			
5. CloverDX Cluster	7			
II. Installation	8			
6. System Requirements	9			
7. Installing	. 11			
Evaluation Server	. 13			
Installation	13			
Production Server	. 15			
Apache Tomcat	. 17			
IBM WebSphere	21			
JBoss Enterprise Application Platform	25			
Oracle WebLogic Server	29			
Activation	32			
CloverDX Server Activation Alternatives	34			
IBM InfoSphere MDM Plugin Installation	35			
Installation into Server	35			
8. Postinstallation Configuration	37			
Memory Settings	. 37			
Maximum Number of Open Files	. 39			
Maximum Number of Processes or Threads	39			
Firewall Exceptions	39			
Garbage Collector for Worker	40			
9. Optional Installation Steps	. 41			
Adding Libraries to the Worker's Classpath	41			
Worker Support for SMB 2.x and 3.x	41			
10. Troubleshooting	42			
III. Configuration	47			
11. Introduction	48			
12. Configuration Sources	. 50			
Configuration File on Specified Location	. 50			
Alternative Configuration Sources	50			
Priorities of Configuration Sources	51			
Specifying the Path to the Configuration File	52			
13. Setup	54			
Before You Start	. 54			
Using Setup	55			
Configuration File	56			
License	. 57			
Database	57			
Worker	58			
Sandboxes	. 58			
Encryption	59			
E-Mail	. 59			
LDAP	60			
Cluster	. 61			
14. System Database Configuration	63			
Embedded Apache Derby	65			
MySQL	66			
Creating Database	66			
CloverDX Server Setup	66			
DB2	. 67			

Creating Database	67
CloverDX Server Setup	. 67
DB2 on AS/400	69
Oracle	70
Creating Database	70
CloverDX Server Setup	. 70
Microsoft SQL Server	. 71
Creating Database	71
CloverDX Server Setup	. 71
PostgreSQL	. 73
Creating Database	73
CloverDX Server Setup	. 73
JNDI Configuration and Encryption	. 75
JNDI DB Datasource	75
JNDI Datasource Troubleshooting	. 76
Encrypted JNDI	76
15. List of Configuration Properties	. 82
General Configuration Properties	82
Worker - Configuration Properties	. 90
Worker - JNDI Properties	. 92
JDBC Datasources	. 92
JMS Connections	93
Worker - SSL Properties	95
Properties on Worker's Command Line	95
In Execution Properties	98
List of all properties	98
16 Secure Configuration Properties	101
17 Logoing	104
IV Administration	107
18 Monitoring	108
Standalone Server Detail	108
Cluster Overview	113
Node Detail	114
Server Logs	115
Using the Monitoring	116
Restarting the Worker	116
Showing Worker's Command Line Arguments	116
Suspending the Server	116
Resuming the Server	116
Displaying List of Threads of the Server Core	116
10 Temp Space Management	117
19. Temp Space Management	117
Management	118
20 Socure Decementers	120
20. Secure 1 diameters	120
I DAD Authentication	124
LDAF Authentication	120
Oscis	127
User Leakout	1/2
22 Sandhoved Server Side Job Files	145
22. Saluboxes - Server Side Job Files	145
Sandhay Contant Security and Dermissione	14/ 1/0
Sandbox Content security and Permissions	14ð
Dependence of ZD	148
Download sandbox as ZIP	149
Upload ZIP to sandbox	149
	149
Delete Sandbox	150
Job Config Properties	151

WebDAV Access to Sandboxes	154
WebDAV Clients	154
WebDAV Authentication/Authorization	154
23. Server Configuration Migration	156
Server Configuration Export	157
Server Configuration Import	158
24 Ungrading Server to Newer Version	160
25. Diagnostics	163
26. Troubleshooting Worker	165
20. Houbleshooting worker	165
V. Using Graphs	10/
27. Graph/Jobilow Parameters	168
Parameters by Execution Type	169
Adding Another Graph Parameters	171
28. Tasks	172
Send an Email	173
Placeholders	174
Execute Shell Command	176
Start a Graph	179
Start a Jobflow	182
Start a Profiler Job	184
Abort job	185
Archive Records	186
Send a IMS Message	188
Execute Groovy Code	190
29 Manual Task Execution	192
30 Scheduling	103
Timatahla Satting	104
Allocations of Scheduled Tests on Nodes	107
Allocations of Scheduled Task of Nodes	197
Scheduning the Lasks - Examples	190
Start a graph at specific time	198
Start a Jobflow once an hour	198
Complex Scheduling	198
31. Viewing Job Runs - Execution History	199
Filtering and ordering	199
Tracking	201
Log File	202
32. Listeners	204
Graph Event Listeners	206
Graph Events	206
Listener	207
Tasks	207
Use Cases	207
Jobflow Event Listeners	212
Jobflow Events	212
Listener	213
Tasks	213
IMS Message Listeners	213
Universal Event Listeners	214
Figure Criteria	217 210
Evaluation Chiefia	219
File Event Listeners (remote and local)	221
Cluster environment	222
Supported filesystems and protocols	223
Ubserved file	224
File Events	225
Check Interval, Task and Use Cases	226
Howtos	226
Task Failure Listeners	229
Task Choice	229

Task Failed E-mail Template	230
33. Recommendations for Transformations Developers	231
34. Extensibility - CloverDX Engine Plugins	232
35. Troubleshooting	233
VI. API	234
36. Simple HTTP API	235
37. JMX mBean	246
JMX Configuration	246
Operations	248
38. SOAP WebService API	249
SOAP WS Client	249
SOAP WS API Authentication/Authorization	249
39. Launch Services	250
Launch Services Authentication	255
Sending the Data to Launch Service	255
Results of the Graph Execution	255
40. Data Services	257
Overview	257
User Interface	258
List of Data Services	258
Detail	259
Testing and Documentation	260
State and History	261
Alerts and Notification	261
Configuration	264
Catalog of Services	264
Built-in Data Service Examples	265
HTTPS Connectors	266
Using Data Services	269
Deploving Data Service	269
Publishing and Unpublishing Data Service from Sandbox	270
Publishing Data Service Examples	270
Changing Data Service to Anonymous	270
Running Data Service on HTTPS	271
Running Data Service on HTTPS on Cluster	273
Monitoring Data Service	273
Testing Data Service	273
Performance Tuning	273
Exporting Data Service Configuration	273
Importing Data Service Configuration	273
Avoiding Premature Marking of Data Service as Failing	274
Looking up Particular Data Service	274
Resetting State of Failing Data Service Endpoint	274
Custom HTTP Headers	275
Data Services on Cluster	275
VII. Cluster	276
41. Sandboxes in Cluster	277
Using a Sandbox Resource as a Component Data Source	279
Remote Edges	280
42. Cluster Configuration	281
Mandatory Cluster Properties	282
Optional Cluster Properties	283
Example of 2 Node Cluster Configuration	286
Basic 2-nodes Cluster Configuration	286
2-nodes Cluster with Proxied Access to Database	287
2-nodes Cluster with Load Balancer	288
Example of 3 Node Cluster Configuration	290
1 0	•

	Jobs Load Balancing Properties	292
	Running More Clusters	293
	43. Recommendations for Cluster Deployment	294
	44. Troubleshooting	295
	NodeA Cannot Establish HTTP Connection to NodeB	295
	NodeA Cannot Establish TCP Connection (Port 7800 by Default) to NodeB	296
	NodeB is Killed or It Cannot Connect to the Database	296
	Auto-Resuming in Unreliable Network	297
	Long-Term Network Malfunction May Cause Jobs to Hang on	297
VIII.	Security	299
	45. Security Recommendations for CloverDX Server	300

Part I. Introduction

Chapter 1. What is CloverDX Server?

CloverDX Server is an enterprise runtime, monitoring and automation platform for the **CloverDX** data integration suite. It is a Java application built to J2EE standards with HTTP and SOAP Web Services APIs providing an additional automation control for integration into existing application portfolios and processes.

CloverDX Server provides necessary tools to deploy, monitor, schedule, integrate and automate data integration processes in large scale and complex projects. **CloverDX Server** supports a wide range of application servers: Apache Tomcat, IBM WebSphere, JBoss EAP and Oracle WebLogic Server.

CloverDX Server simplifies the process of:

- *Operation* CloverDX Server allows you to <u>set up (p. 54)</u> and <u>monitor (p. 108)</u> the status of the Server and jobs (p. 199) and <u>notify you via an email (p. 173)</u> if the job fails;
- *Automation* It allows you to efficiently handle jobflow events via <u>listeners (p. 204)</u> and <u>schedule (p. 193)</u> tasks to be triggered as one-time events or repeatedly, as required;
- Administration It helps you manage <u>users (p. 129)</u> and <u>groups (p. 131)</u> and their privileges, create and <u>configure sandboxes (p. 145)</u> and <u>export the configuration (p. 156)</u> to another instance of the Server. Furthermore, the Server provides <u>API (p. 234)</u> and allows you to create it via <u>Data Services (p. 257)</u>;
- *Security* For better control over **CloverDX Server**, you can set up <u>user lockout (p. 143)</u> and <u>encrypt sensitive</u> <u>data (p. 120)</u>.

To learn more about the architecture of **CloverDX Server**, see <u>Chapter 2</u>, <u>CloverDX Server</u> <u>Architecture (p. 3)</u>.



Figure 1.1. CloverDX Server User Interface

Chapter 2. CloverDX Server Architecture

CloverDX Server is a Java application distributed as a web application archive (.war) for an easy deployment on various application servers. It is compatible with Windows and Unix-like operating systems.

CloverDX Server requires Java Development Kit (JDK) to run. We **do not recommend** using Java Runtime Environment (JRE) only, since compilation of some transformations requires JDK to function properly.

The Server requires some space on the file system to store persistent data (transformation graphs) and temporary data (temporary files, debugging data, etc.). It also requires an external relational database to save run records, permission, users' data, etc.



Figure 2.1. System Architecture

The CloverDX Server architecture consists of Core and Worker.

CloverDX Core

CloverDX Server's Core manages <u>users and groups</u> (p. 124) checks permissions, <u>schedules</u> (p. 193) execution and provides <u>management</u> (p. 54) and <u>monitoring</u> (p. 108) UI. It provides APIs for other applications: <u>Data Service API</u>(p. 257), <u>HTTP API</u> (p. 235) and <u>Web Service API</u>(p. 249) For more information, see Chapter 3, <u>CloverDX Core</u> (p. 4).

CloverDX Worker

Worker is a separate process that executes jobs: graphs, jobflows and profiler jobs. The purpose of Worker is to provide a sandboxed execution environment. For more information, see Chapter 4, <u>CloverDX Worker</u> (p. 5).

Dependencies on External Services

The Server requires a database to store its configuration, user accounts, execution history, etc. It comes bundled with an Apache Derby database to ease the evaluation. To use CloverDX Server in production environment, a relational database (p. 10) is needed.

The Server needs a connection to an <u>SMTP server</u> (p. 59) to be able to send you notification emails.

Users and groups' data can be stored in the database or read from an LDAP server (p. 125).

Server Core - Worker Communication

Server Core receives Worker's stdout and stderr. The processes communicate via TCP connections.

Chapter 3. CloverDX Core

CloverDX Core is the central point of **CloverDX Server** It manages and monitors (p. 108)Workers (p. 5) that run the jobs. CloverDX Core launches scheduled (p. 193)jobs, checks user permissions (p. 131) sends notifications (p. 173) and forwards communication between Designer and Worker.

CloverDX Core is the visible part of the Server with a web-based user interface.

CloverDX Core connects to the system database (p. 63) and stores its configuration and service records in it. The system database is required. If it is configured, the Core connects to an SMTP server (p. 59) to send notification emails or to an LDAP server (p. 60) to authenticate users against an existing LDAP database.

Chapter 4. CloverDX Worker

Worker is a standalone JVM running separately from the Server Core. This provides an isolation of the Server Core from executed jobs (e.g. graphs, jobflows, etc.). Therefore, an issue caused by a job in Worker will not affect the Server Core.

Worker does not require any additional installation - it is started and managed by the Server. Worker runs on the same host as the Server Core, i.e. it is not used for parallel or distributed processes. In case of a Cluster, **each** Cluster node has its **own** Worker.

Worker is a relatively light-weight and simple executor of jobs. It handles job execution requests from the Server Core, but does not perform any high-level job management or scheduling. It communicates with the Server Core via an API for more complex activities, e.g. to request execution of other jobs, check file permissions, etc.

Configuration

General configuration

Worker is started by the Server Core as a standalone JVM process. The default configuration of Worker can be changed in the <u>Setup</u> (p. 58):

- Heap memory limits
- Port ranges
- Additional command line arguments (e.g. to tweak garbage collector (p. 40) settings)

The settings are stored in the usual Server configuration file. Worker is configured via special <u>configuration</u> <u>properties</u> (p. 90).

A full command line (p. 116) of Worker is available in the Monitoring section.

There are several properties that are propagated or otherwise put on the Worker's command line. For a full list, see the <u>Properties on Worker's command line</u> (p. 95).

Cluster specific configuration

Cluster should use a single <u>portRange</u> (p. 90) all nodes should have identical value of portRange. That is the preferred configuration, although different ranges for individual nodes are possible.

Management

The Server manages the runtime of Worker, i.e. it is able to start, stop, restart Worker, etc. Users don't need to manually install and start Worker.

Status of Worker and actions are available in the Monitoring Worker (p. 109) section.

In case of problems with Worker, see Chapter 26, Troubleshooting Worker (p. 165).

Job Execution

By default, all jobs are executed in Worker; yet the Server Core still keeps the capability to execute jobs. It is possible to set specific jobs or whole sandboxes to run in the Server Core via the worker_execution (p. 153) property on the job or sandbox. It is also possible to disable Worker completely (p. 90), in which case all jobs will be executed in the Server Core.

Executing jobs in the Server Core should be an exception. To see where the job was executed, look in the run details in **Execution History** - in the Executor field. Jobs started in Worker also log a message in their log, e.g. Job is executed on Worker:[worker0@node01:10500].

Job Configuration

The following areas of Worker configuration affect job execution:

• JNDI

Graphs running in Worker cannot use JNDI as defined in the application container of the Server Core, because Worker is a separate JVM process. Worker provides its own JNDI configuration (p. 92).

• Classpath

The classpath is not shared between the Server Core and Worker. If you need to add a library to the Worker classpath, e.g. a JDBC driver, follow the instructions in <u>Adding Libraries to the Worker's Classpath</u> (p. 41).

Chapter 5. CloverDX Cluster

CloverDX Cluster allows multiple instances of **CloverDX Server** to run on different hardware nodes and form a computer cluster. In this distributed environment, data transfer between **CloverDX Server** instances is performed by <u>Remote Edges</u> (p. 280).

CloverDX Cluster offers several advantages for Big Data processing:

• **High Availability** - All nodes are virtually equal; therefore, almost all request can be processed by any cluster node. This means that if one node is disabled, another node can substitute it.

To achieve high availability, it is recommended to use an independent HTTP load balancer. See the configuration in the <u>2-nodes Cluster with Load Balancer</u> (p. 288) example.

• Scalability - Allows for increased performance by adding more nodes.

There are two independent levels of scalability implemented. Scalability of transformation requests and data scalability.

For general information about CloverDX Cluster, see the CloverDX Designer documentation.

CloverDX Cluster is configured by a set of specific <u>Mandatory Cluster Properties</u> (p. 282) and <u>Optional Cluster</u> <u>Properties</u> (p. 283).

In Cluster environment, you can use several types of sandboxes, see Chapter 41, Sandboxes in Cluster (p. 277).



Note

Note: CloverDX Cluster requires a special license.

Part II. Installation

Chapter 6. System Requirements

Hardware Requirements

The following table shows hardware requirements for both Basic and Corporate **CloverDX Server** edition and for running **CloverDX Server** in a cluster.

Table 6.1.	Hardware	requirements	of CloverDX Se	rver
------------	----------	--------------	----------------	------

Hardware	Basic Edition	Corporate Edition	Cluster
RAM	4 GB (recommended 16 GB)	8 GB (recommended 64 GB)	8 GB (recommended 64 GB)
Processors	up to 4 cores	16 cores	8 cores ^a
Disk space (installation)	1 GB	1 GB	1 GB
Disk space (tempspace)	> 25 GB ^b	> 25 GB ^b	> 25 GB ^b
Disk space (data)	> 50 GB ^b	> 50 GB ^b	> 50 GB ^b
Disk space (shared) ^c	-	-	> 50 GB ^b

^aThis may vary depending on total number of nodes and cores in license.

^bMinimum value, the disk space depends on data.

^cDisk space for shared sandboxes is required only for **CloverDX Cluster**.

Software Requirements

Operating system

CloverDX Server is compatible with Windows and Unix-based systems, as well as with other systems supporting Java (Mac OS X, IBM System, etc.).

Java Virtual Machine

- Oracle JDK 8 or 9 64 bit
- IBM SDK 8 (for IBM WebSphere only)

Since it is a Java application, it requires Java Development Kit (JDK) to run. We **do not recommend** using Java Runtime Environment (JRE) only, since compilation of some CloverDX Server's transformations requires JDK to function properly.

Application Servers Compatible with CloverDX 5.0

- <u>Apache Tomcat 8.5 or 9</u> (p. 17)
- <u>IBM WebSphere 9</u> (p. 21)
- <u>JBoss EAP 7</u> (p. 25)
- Oracle WebLogic Server 12cR2 (12.2.1) 32/64 bit (p. 29)

Table 6.2. Compatibility Matrix

	CloverET	L 4.1 to 4.7	CloverET	L 4.8 to 4.9	Clover and 1	DX 5.0 newer
Application Server	Java 7	Java 8	Java 7	Java 8	Java 8	Java 9
Tomcat 6	~	~	×	×	×	×
Tomcat 7	~	~	~	~	×	×
Tomcat 8	~	~	~	~	×	×
Tomcat 8.5	×	×	✓ ^a	✓ ^a	~	~
Tomcat 9	×	×	×	~	~	~
Pivotal tc Server Standard (3.1.9, Tomcat 8)	×	~	×	~	×	×
Pivotal tc Server Standard (3.2.9, Tomcat 8.5)	×	×	×	<mark>√</mark> a	~	*
Jetty 9	~	~	~	~	×	×
WebLogic Server 11gR1 (10.3.6)	*	×	4.8 ✓ / 4.9 ×	×	×	×
WebLogic Server 12cR1 (12.1.2)	~	×	~	×	×	×
WebLogic Server 12cR1 (12.1.3)	~	~	~	~	×	×
WebLogic Server 12cR2 (12.2.1)	×	×	×	×	~	×
JBoss AS 6	~	×	~	×	×	×
JBoss AS 7	✓ ^b	~ ^c	✓ ^b	 ✓^c 	×	×
JBoss EAP 7	×	×	×	×	~	×
Glassfish 3	~	×	~	×	×	×
WebSphere 8.5	~	×	~	×	×	×
WebSphere 9	×	×	×	×	~	×

^aSince 4.8.2 ^bEAP 6.2 ^cEAP 6.4



Note

We support Java 8 and 9 on particular supported application server only if the application server itself officially supports it.

Database servers

We support the following database servers. The officially supported versions, we are testing against, are in parentheses.

- <u>MySQL (5.6.12)</u> (p. 66)
- <u>DB2 (10.5.1)</u> (p. 67)
- <u>Oracle (11.2.0.2.0)</u> (p. 70)
- <u>MS SQL Server 2014 (12.0.5557.0)</u> (p. 71)
- <u>PostgreSQL (9.2.4)</u> (p. 73)

The evaluation version uses an embedded Apache Derby database for evaluation purposes. We **do not support** using the Derby database in production environment. Instead you can choose one of several supported database servers.

Chapter 7. Installing

This chapter describes two different Server installations - <u>Evaluation Server</u> (p. 13) and <u>Production</u> <u>Server</u> (p. 15) - and provide instructions on installing the **CloverDX Server** License.

Evaluation Server

The <u>Evaluation Server</u> (p. 13) consists of **CloverDX Server** bundled with the Tomcat application container. The Server performs basic configuration during the first startup and requires no additional database server. This option is **recommended only for basic evaluation** of **CloverDX Server**'s functions.

However with further configuration, it is possible to evaluate other CloverDX Server features and even make the Evaluation Server **ready for production environment**. This process requires a connection to an external, dedicated database and subsequent configuration of services (e.g. SMTP, LDAP, etc.).



Important

The Apache Derby DB, bundled with the Evaluation Server, is **not** supported for production environment. Please use one of the supported database systems.

Production Server

In case of <u>Production Server</u> (p. 15), the **CloverDX Server** is installed on one of the several compatible application containers. This process requires additional configuration (e.g. memory allocation, database connection, etc.) but allows you to choose an application container and external database according to your preference.

Installation and Configuration Procedure

To create a fully working instance of Production CloverDX Server, you should:

• Install an application server

CloverDX Server is compatible with several application containers. Following subsections offer detailed instructions on installation of the respective application servers and their subsequent configuration.

• Set up limits on a number of opened files, memory allocation and firewall exceptions

CloverDX Server's graph transformations and evaluations may require more memory than the default limit set in the database as well as higher number of simultaneously opened files. Moreover, some components require firewall exceptions to be set. These instructions provide recommendation on adjusting both the <u>Memory Settings</u> (p. 37) and the <u>Maximum Number of Open Files</u>(p. 39) as well as <u>Firewall</u> <u>Exceptions</u> (p. 39).

• Install CloverDX Server into the application server

CloverDX Server is provided as a web archive (.war) file for an easy deployment.

Create a database dedicated to CloverDX Server

Unlike the Evaluation Server, the Production Server requires that you have created a dedicated database for **CloverDX Server**. In the configuration phase of this manual, you will be guided to Chapter 14, <u>System Database</u> <u>Configuration</u> (p. 63) with instructions on how to properly configure the properties file of various databases.

• Set up a connection to the database

You can set up the connection to the database via Server's GUI in **Configuration** \rightarrow **Setup** \rightarrow **Database**. Alternatively, you can set up the connection in the properties file before starting **CloverDX Server**.

• Install a license

To be able to execute graphs, you need to install a valid license, see Activation (p. 32).

Perform additional Server configuration

• Set up a master password for secure parameters

When handling sensitive information (e.g. passwords), it is advised to define secure graph parameters. This action requires a master password (see Chapter 20, <u>Secure Parameters</u> (p. 120)).

• Set up SMTP server connection

CloverDX Server lets you configure an SMTP (p. 59) connection for reporting events on the Server via emails.

• Configure temp space

CloverDX Server works with temporary directories and files. To ensure components work correctly, you should configure the Temp space location on the file system. For details, see Chapter 19, <u>Temp Space Management</u> (p. 117).

• Configure sandboxes

Lastly, you should set the content security and user's permissions for sandboxes. For details and instructions, see Chapter 22, <u>Sandboxes - Server Side Job Files</u> (p. 145).

Evaluation Server

The default installation of **CloverDX Server** uses embedded Apache Derby DB; therefore, it does not require any external database server or subsequent configuration, as **CloverDX Server** configures itself during the first startup. Database tables and some necessary records are automatically created on the first startup with an empty database.

By performing a subsequent configuration, you can evaluate other **CloverDX Server** features (e.g. sending emails (p. 173) LDAP authentication (p. 125) clustering (p. 277) etc.). This way, you can also prepare the Evaluation Server for production environment. However, note that the embedded Apache Derby database is **not** supported for production environment. Therefore, before the subsequent configuration, choose one of the supported external dedicated databases.

If the **CloverDX Server** must be evaluated on application containers other than Tomcat, or you prefer a different database system, proceed with a common installation of <u>Production Server</u> (p. 15)



Note

Default login credentials for CloverDX Server Console are:

Username: clover

Password: clover

Installation

1. Make sure you have a compatible Java version:

Oracle JDK or JRE v. 1.8.x or higher is required. We recommend JDK 1.8.x.

2. Download and extract the CloverDX Evaluation Server.

2.aLog into your CloverDX account and download the Evaluation Server Bundle.

2.bExtract the .zip archive. The name of the file is CloverDX.<version>.Tomcat-<version>.zip.



Note

It is recommended to place the extracted content on a path that does not contain space character(s).

C:\Program Files or /home/user/some dir ¥

C:\Users\Username or /home/user/some_dir *

- 3. Set the JAVA_HOME or JRE_HOME Environment Variables:
 - Unix-like systems:
 - Open the /bin/setenv.sh file and define the path at the beginning of the file:

export JAVA_HOME=/opt/jdk1.8.0_121

```
export JAVA HOME=/opt/jdk1.8.0_121
if [ -n "$JRE_HOME" -a -n "$JAVA_HOME" ]; then
        echo "Using JAVA_HOME instead of JRE_HOME"
        unset JRE_HOME
fi
```

Figure 7.1. setenv.sh edited in Linux.

• Windows system:

• Open the /bin/setenv.bat file and define the path at the beginning of the file:

```
set "JAVA_HOME=C:\java\jdk1.8.0"

1 %echo off
2 set "JAVA_HOME=C:\java\jdk1.8.0"
3 IF DEFINED JAVA_HOME IF DEFINED JAVA_HOME (
4 echo "Using JAVA_HOME Instead of JRE_HOME"
5 set JRE_HOME=%JAVA_HOME%
```

Figure 7.2. setenv.bat edited in Windows.

- 4. Run Tomcat.
 - Unix-like systems: run /bin/startup.sh.
 - Windows system: run \bin\startup.bat.
- 5. Log in CloverDX Server.

5.aType http://localhost:8083/clover/ in your browser.

- 5.bActivate (p. 32) the CloverDX Server.
- 5.cUse the default administrator credentials to access the web GUI:

Username: clover

Password: clover

6. **CloverDX Server** is now installed and prepared for basic evaluation. There are couple of sandboxes with various demo transformations installed.



Note

To safely stop the server, run /bin/shutdown.sh or \bin\shutdown.bat on Unix-like or Windows system respectively.

Production Server

This section describes in detail the installation of **CloverDX Server** on various application containers and its subsequent configuration required for production environment. For simple evaluation of **CloverDX Server** features, use Evaluation Server (p. 13) (note that CloverDX Evaluation Server can also be configured for production use).

CloverDX Server for production environment is shipped as a *Web application archive* (WAR file) and uses an external, dedicated database, so standard methods for deploying a web application on your application server may be used. However, each application server has specific behavior and features. Detailed information about their installation and configuration can be found in the following chapters.

List of Suitable Containers

- Apache Tomcat (p. 17)
- IBM WebSphere (p. 21)
- <u>JBoss Enterprise Application Platform</u> (p. 25)
- Oracle WebLogic Server (p. 29)

In case of problems during the installation see Chapter 10, <u>Troubleshooting</u> (p. 42).



Important

Oracle JDK or JRE v. 1.8.x or 1.9.x is required.

Installation and Configuration Procedure

To create a fully working instance of Production CloverDX Server, you should:

1. Install an application server

CloverDX Server is compatible with several application containers. Following subsections offer detailed instructions on installation of the respective application servers and their subsequent configuration.

2. Set up limits on a number of opened files, memory allocation and firewall exceptions

CloverDX Server's graph transformations and evaluations may require more memory than the default limit set in the database as well as higher number of simultaneously opened files. Moreover, some components require firewall exceptions to be set. These instructions provide recommendation on adjusting both the <u>Memory Settings</u> (p. 37) and the <u>Maximum Number of Open Files</u>(p. 39) as well as <u>Firewall</u> Exceptions (p. 39).

3. Install CloverDX Server into the application server

CloverDX Server is provided as a web archive (.war) file for an easy deployment.

4. Create a database dedicated to CloverDX Server

Unlike the Evaluation Server, the Production Server requires that you have created a dedicated database for **CloverDX Server**. In the configuration phase of this manual, you will be guided to Chapter 14, <u>System Database Configuration</u> (p. 63) with instructions on how to properly configure the properties file of various databases.

5. Set up a connection to the database

You can set up the connection to the database via Server's GUI in Configuration \rightarrow Setup \rightarrow Database. Alternatively, you can set up the connection in the properties file before starting CloverDX Server.

6. Install a license

To be able to execute graphs, you need to install a valid license. There are three options for <u>Activation</u> (p. 32).

7. Perform additional Server configuration

• Set up a master password for secure parameters

When handling sensitive information (e.g. passwords), it is advised to define secure graph parameters. This action requires a master password (see Chapter 20, <u>Secure Parameters</u> (p. 120)).

• Set up SMTP server connection

CloverDX Server lets you configure an SMTP (p. 59) connection for reporting events on the Server via emails.

• Configure temp space

CloverDX Server works with temporary directories and files. To ensure components work correctly, you should configure the Temp space location on the file system. For details, see Chapter 19, <u>Temp Space</u> <u>Management</u> (p. 117).

• Configure sandboxes

Lastly, you should set the content security and user's permissions for sandboxes. For details and instructions, see Chapter 22, <u>Sandboxes - Server Side Job Files</u> (p. 145).

Apache Tomcat

Installation of Apache Tomcat (p. 17) Apache Tomcat as a Windows Service (p. 17) Apache Tomcat on IBM AS/400 (iSeries) (p. 18) Installation of CloverDX Server (p. 19) Configuration of CloverDX Server on Apache Tomcat (p. 20)



Important

Before installation, check the software requirements, currently supported Apache Tomcat version and required Java version in the <u>Software Requirements</u> (p. 9) section.

If you encounter any problems during the installation, see Chapter 10, <u>Troubleshooting</u> (p. 42) for a possible solution.

Installation of Apache Tomcat

- 1. Download the binary distribution: <u>Tomcat 8.5</u> or <u>Tomcat 9</u>.
- 2. Extract the downloaded archive (zip or tar.gz).
- 3. Set up JAVA_HOME to point to the correct Java version:
 - Unix-like systems:

Setup the path in /etc/profile or /etc/bash.bashrc:

export JAVA_HOME=/path/to/JDK

• Windows system:

Under **System Variables** in **Advanced System Settings**, create a new variable named JAVA_HOME. The value should contain the path to the JDK installation directory.

- 4. Run Tomcat:
 - Unix-like systems:

Run the [Tomcat_home]/bin/startup.sh file.

• Windows system:

Run the [Tomcat_home]\bin\startup.bat file.

- 5. Check whether Tomcat is running.
 - Open a new tab in your browser and type http://localhost:8080/ in the address bar.

Continue with: Installation of CloverDX Server (p. 19).

Apache Tomcat as a Windows Service

- 1. Download the **32-bit/64-bit Windows Service Installer** file in the **Binary Distributions** section on the <u>Tomcat</u> <u>8.5</u> or <u>Tomcat 9</u> download page.
- 2. Use the standard installation wizard to install Apache Tomcat.
- 3. When Tomcat is installed as a Windows service, **CloverDX** is configured by one of the following options:

a. Graphical configuration utility

- Run the [Tomcat_home]\bin\Tomcat9w.exe file.
- In the **Apache Tomcat Properties** dialog box, select the **Java** tab and set the initial and maximum heap size in **Initial memory pool** and **Maximum memory pool** fields to 512MB and 1024MB respectively. Other configuration parameters can be defined in **Java Options** field, being separated by new line.
- Click on Apply and restart the service.



Note

The **Java** tab allows you to use alternative Java virtual machine by setup of path to jvm.dll file.

b. Command Prompt tool

- Run the [Tomcat_home]\bin\Tomcat9.exe file.
- If Tomcat is running, navigate to [Tomcat_home]\bin and stop the service by typing:

.\Tomcat9.exe //SS//Tomcat9

in the Command Prompt. (When using different version of Tomcat, change the number in the command to reflect the installed version.)

• Configure the service by typing the command:

```
.\Tomcat9.exe //US//Tomcat9 --JvmMs=512 --
JvmMx=1024 --JvmOptions=-Dclover.config.file=C:\path\to\clover-
config.properties#-XX:MaxMetaspaceSize=256m
```

The parameter JvmMs is the initial and JvmMx is the maximum heap size in MB; JvmOptions are separated by '#' or ';'.

• Start the service from Windows administration console or by typing the following command in the Command Prompt:

.\Tomcat9.exe //TS//Tomcat9



Tip

By default, when Apache Tomcat is run as a Windows service, it is **not available** for Java process monitoring tools (e.g., **JConsole** or **JVisualVM**). However, these tools can still connect to the process via **JMX**. In order to expose Tomcat's Java process via JMX, add the following options to the service settings:

```
-Dcom.sun.management.jmxremote.port=3333
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.authenticate=false
```

Once the service is run with these options, you can connect to **port 3333** using JMX and monitor the server.

More information about running Java applications as Windows Service can be found at Apache Commons.

Continue with: Installation of CloverDX Server (p. 19).

Apache Tomcat on IBM AS/400 (iSeries)

Additional settings are required to run CloverDX Server on the iSeries platform:

- 1. Declare you are using Java 8.0 32-bit.
- 2. Run Java with parameter -Djava.awt.headless=true.

To configure the settings, modify (or create) the [Tomcat_home]/bin/setenv.sh file to contain:

```
JAVA_HOME=/QOpenSys/QIBM/ProdData/JavaVM/jdk70/32bit
JAVA_OPTS="$JAVA_OPTS -Djava.awt.headless=true"
```

Continue with: Installation of CloverDX Server (p. 19)

Installation of CloverDX Server

1. Check if you meet the prerequisites:

- Oracle JDK or JRE is installed (See Java Virtual Machine (p. 9) for required Java version.)
- JAVA_HOME or JRE_HOME environment variable is set (see <u>Setting up JAVA_HOME</u> (p. 17)).
- A supported version (p. 10) of Apache Tomcat is installed.
- 2. It is strongly recommended to adjust the default limits for **Memory allocation** (see the <u>Memory Settings</u> (p. 37) section).

You can set the initial and maximum memory heap size by adjusting the Xms and Xmx JVM parameters:

Unix-like systems:

- Create the [Tomcat_home]/bin/setenv.sh file.
- Type or paste in the following lines:

```
export CATALINA_OPTS="$CATALINA_OPTS -XX:MaxMetaspaceSize=512m -Xms128m -Xmx1024m"
export CATALINA_OPTS="$CATALINA_OPTS -Dderby.system.home=$CATALINA_HOME/temp -server"
echo "Using CATALINA_OPTS: $CATALINA_OPTS"
```

Windows systems:

- Create the [Tomcat_home]\bin\setenv.bat file.
- Type or paste in the following lines:

```
set "CATALINA_OPTS=%CATALINA_OPTS% -XX:MaxMetaspaceSize=512m -Xms128m -Xmx1024m"
set "CATALINA_OPTS=%CATALINA_OPTS% -Dderby.system.home=%CATALINA_HOME%/temp -server"
echo "Using CATALINA_OPTS: %CATALINA_OPTS%"
```



Tip

For performance reasons, it is recommended to run the container in the "server mode" by setting the -server switch, as seen in the settings above.¹

Note that on a 64-bit capable JDK, only the Java Hotspot Server VM is supported so the -server option is implicit.

- 3. Go to the download section of your CloverDX account and download the clover.war (web archive) file containing CloverDX Server for Apache Tomcat.
- 4. Copy clover.war to the [Tomcat_home]/webapps directory.

- 5. Tomcat should automatically detect and deploy the clover.war file.
- 6. Check whether **CloverDX Server** is running:
 - Run Tomcat.
 - Open a new tab in your browser and type <u>http://localhost:8080/clover/</u> in the address bar.
 - Use the default administrator credentials to access the web GUI: username: clover, password: clover.

Continue with: <u>Configuration of CloverDX Server on Apache Tomcat</u> (p. 20)

Configuration of CloverDX Server on Apache Tomcat



Tip

Default installation (without any configuration) is only recommended for evaluation purposes. For production use, at least a dedicated, system database and SMTP server configuration is recommended.

For easy configuration of **CloverDX Server**, use Setup GUI (p. 54) in which you can configure various properties, including the connection to the database, username and password, path to the license file, private properties, clusters and much more (see Chapter 15, <u>List of Configuration Properties</u>(p. 82) and Chapter 42, <u>Cluster Configuration</u> (p. 281). We recommend you place the file in a specified (p. 50)location and define the path to the file with a system property.

The content of such a file (an example with a PostgreSQL database):

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://127.0.0.1/clover_db?charSet=UTF-8
jdbc.username=yourUsername
jdbc.password=yourPassword
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
```

Properties File in Specified Location

The properties file is loaded from a location specified by a system property or by an environment variable clover_config_file or clover.config.file.

- 1. Create the cloverServer.properties file in a directory readable by Apache Tomcat. (If you need an example of connection to any of the supported database systems, see Chapter 14, <u>System Database</u> <u>Configuration</u> (p. 63).)
- 2. Edit the [Tomcat_home]/bin/setenv.sh file (if it does not exist, you can create it).
- 3. Set the system property by adding the following line into the file:

JAVA_OPTS="\$JAVA_OPTS -Dclover_config_file=/path/to/cloverServer.properties"



Note

Continue with: Chapter 8, <u>Postinstallation Configuration</u> (p. 37)

IBM WebSphere

Installation of CloverDX Server on IBM WebSphere (p. 21) Configuration of CloverDX Server on IBM WebSphere (p. 23)



Important

Before installation, check the software requirements, currently supported IBM WebSphere version and required Java version in the <u>Software Requirements</u> (p. 9) section.

If you encounter any problems during the installation, see Chapter 10, <u>Troubleshooting</u> (p. 42) for a possible solution.

Installation of CloverDX Server on IBM WebSphere

1. Download and install IBM WebSphere from the official download page.

Note: During installation, make sure the IBM WebSphere profile name does not contain the keyword **clover**, otherwise the **CloverDX Server** won't start properly.

2. It is strongly recommended to adjust the default limits for **Memory allocation** (see the <u>Memory Settings</u> (p. 37) section).

You can set the limits in IBM WebSphere's **Integrated Solutions Console** (default URL: <u>http://localhost:9060/</u> <u>ibm/console/</u>).

- a. Go to Servers →Server Types →WebSphere application servers →[Server_Name] (default name: server1) →Java and Process Management →Process definition →Java Virtual Machine
- b. Change the value in the **Maximum heap size** field to 2048 MB. The default value (256 MB) is insufficient for ETL transformations.

Week All tasks Mechan Mechan	WebSphere, software	Welcom	e Help Logout IBM.
• Useda Useda bases to configure advanced 2ava(TM) virtual machine settings. • General Properties Additional Properties • Vaciphere sequences Additional Properties • Vaciphere sequences Additional Properties • Additional Properties • Cution • Boot Classpath • Cution • Socice Cution • Verboos dass loading	View: All tasks	Application servers > server1 > Process definition > Java Virtual Machine	^
Bodde Activities Configuration Runtime Server Types Additional Properties Additional Properties WedSphere application servers Education Configuration Application Types Configuration Consecution Production Types Configuration Consecution Consecution Production Types Server Types Configuration Consecution Consecution Production Types Server Serv	Welcome	Use this page to configure advanced Java(TM) virtual machine settings.	
Serverses Brownerses Severses Severses Webspress Webspress Additional Properties Additional Properties Additional Properties Severses Severses Additional Properties Additional Properties Severses Severses </td <td>Guided Activities</td> <td>Configuration Runtime</td> <td></td>	Guided Activities	Configuration Runtime	
Service	Servers		
• WdSphere splication saver: Additional Properties • WdSphere splication saver: • Casenal Properties • WdSphere splication: • Casenal Properties • New Application: • Casenal Properties • New Application: • Casenal Properties • WdSphere streptica splication: • Casenal Properties • Service: • Service: • Service: • Service: • Service: • Verbose class loading • Verbose data loading • Verbose class loading • Verbose data loading • Verbose splita splita • Verbose data loading • Verbose splita • Verbose splita security • MB • Daba • Verbose splita • Service: • Verbose splita • Daba • Verbose splita • Daba • Verbose splita • Debug Mode • Debug Mode • Debug Mode • Debug splita • Ourt-Minutta • Dourt-Minutta • Dourt-M	Server Types		
WabSphare MQ searers Additional Properties WabSphare MQ searers Castom PApPlications Castom PApPlications	WebSphere application servers	Caparal Properties	
Application Custom Application Custom Nex Application Custom Application Selication Application Selication Subines-level applications Selication Asser Selication Service clents Selication Service clents Verbose class loading Verbose class loading Verbose class loading Service clents Waimum heap size Service clents MA Dobc Service clents Waimum heap size MA Scholalors Na Obbc Na Service clents Ma Dobc Service clents Waimum heap size Ma Scholalors Na Obbc Service clents Baseurce datapers Ma Baseurce datapers Ma Baseurce datapers Ma Baseurce datapers Ma Baseurce datapers Service clents Baseurce datapers Ma Baseurce datapers Service clents Baseurce	WebSphere MQ servers	deletal Propercies	Additional Properties
Applications • New Applications • Applications • Applications • Applications • Applications • Applications • Beasiness-Ind applications • Chabit deployment settings • Service providers • Service providers • Service classs • Service classs • Service classs • Policy sets • Policy sets • Neadowne settings • Object color massages • Object color massages • Object color massages • Applications beans • Cache instances • Mail • Util • Security cache • Security cache • Object color massages • Object color massages • Security cache • Object color massages • Applications beans • Cache instances • Mail • Util • Security cache • Object color massages • Dable Mode • Debug Mode • Debug Mode • Security cache and parsa • Security cache and pars	Web servers	Classpath	Custom
New Application Application Types Applications Bootics Bootics Bootics Service Service providers Service class Bootics Service class Bootics Service class Service class Bootic service Service class Bootics Pristaervice Service class Service class Bootic service Service class	Applications		properties
Application Types Modeling applications Modeling applications Application Types Advectors Active	New Application		
WebSphere enterprise applications Business-Invel applications Assets Global deployment settings Services Service providers Service clients Service clients Corbose dass loading Service rotations Orbose dass loading Verbose dass loading Verbose garbage collection Verbose dass loading	Application Types		
Busines-levia applications Assets Global deployment settings Services Service providers Service cleans Policy sets Service cleans Service cleans Policy sets Service providers Service cleans Service cleans Policy sets Service providers Service providers<	WebSphere enterprise applications		
- Asses Global deployment settings - Global deployment settings - Service providers - Service cleants - Service cleants - Trast service - Security - Global security - Security - Global security - Global security - Global security - Security - Security - Security - Basses - Basses - Debug arguments - Debug interment - Debug intermen	Business-level applications		
Cickel deployment settings Services Services Service providers Service providers Service Cients Policy sets Trust service Security cache Resources Object pool managers Mail Object pool managers E Cache instances Mail With Resource Adapters Cache instances Mail With B Resource Environment Debug Mode Debug	Assets	Boot Classpath	
 Service providers Service class is class	Global deployment settings		
Service cherss Service cherss Service cherss Security cache Reliable messaging state Verbose class loading Verbose garbage collection Verbose garbage collection Verbose 3 webse state Verbose 3 webse 3 webse state Verbose 3 webse 3 webs	Services		
 Service clears Policy sets Trust service Security cache Resources Verbose garbage collection Verbose garbage collection Verbose JNI Verbose JNI Object pol managers JMS JDBC Resource Adapters Asynchronous baans Cache instances Ruinum heap size Zoda8 Ruinum heap size Dabug Mode Debug Marents -Doron.ibm.xml.xkp.jaxb.opti.level=3 	Service providers		-
Policy sets Trust service Trust service Sacurity cache Resource Adaters Advintative Authorization Groups Security S	Service clients		-
Becurity carbo Security carbo Resource carbo Verbose class loading Verbose garbage collection Verbose of parbages V	Policy sets		
Scurity cache Resources Schadulers Object pool managers Mail DBC Resource Adapters Cache instances Mail URL Resource Environment Debug Mode Debug Mode </td <td>Trust service</td> <td></td> <td></td>	Trust service		
Reliable messaging state Resources Schedulers Object pool managers MS JDBC Resource Adapters Asynchronous beans Cache instances Mail URL Resource Environment Object you management Security Object you management Security Object you management Security Bus security Bus security	Security cache	Verbose class loading	
Resources Schedulers Object pool managers MS DBoC B Resource Adapters Administrative Authorization Groups Security Object area and sky management Debug ments Debug ments </td <td>Reliable messaging state</td> <td>Verbose garbage collection</td> <td></td>	Reliable messaging state	Verbose garbage collection	
Schedulers Initial heap size Ghyter topi managers MB JMS Initial heap size B JDBC MB Resource Adapters Zo48 Cache instances Run HProf Mail HProf Arguments B Clobal security Debug Mode Security Debug arguments SScurity Geheric JVM arguments SScurity Generic JVM arguments SScurity Generic JVM arguments SScurity Generic JVM arguments Sscurity working Generic JVM arguments Sscurity Corn.ibm.xml.xkp.jaxb.opti.level=3	Resources	Verbose INI	
Object pool managers Initial heap size JNS Maximum heap size JOBC ZoAe instances Asynchronous beans Run HProf Mail Maximum heap size URL Debug Mode Security Debug arguments Security chains Security during arguments Security Generic JVM arguments Security Generic JVM arguments Security Security during and they management Security Security	Schedulers		
B/MS MB B/DBC MB B/DBC Maximum heap size 2048 MB 2048 MB Cache instances Run HProf Mail HProf Arguments UkL Debug Mode Security Debug figuments Security Gehalt security Security Generic JVM arguments Security Orom.ibm.xml.xkp.jaxb.opti.level=3	Object pool managers	Initial heap size	
BOBC Resource Adapters Aynchronous baans Cache instances Aynchronous baans Cache instances Mail Mail Wall Bussurce Environment Clobal security Security Comains Security Comains Security Comains Security Comains Security Comains Security Comains Security Security Comains Security Comains Security Comains Security Comains Security Security Comains Security	± JMS	MB	
	± JDBC	Maximum heap size	
Administrative Authorization Groups Security Security Security Security comains SSuctricate and key management Sscurity auding Bus security	Resource Adapters	2048 MB	
Cache instances Mail WRI URL Resource Environment Debug Mode Debug arguments Security domains Security domains Security domains Security adding Bus security Debug arguments -Dcom.ibm.xml.xkp.jaxb.opti.level=3			
Image: Mail will will will will will will will w	Cache instances	Run HProf	
Image: Security Debug Mode Security Debug arguments Security domains - Debug arguments Security arguments - Debug arguments Bus security - Debug arguments - Debug arguments - Debug arguments - Deb	+ Mail	HProf Arguments	
	H URL		
Security Debug Mode Global sacurity Debug arguments Sacurity domains -gentilbij/buge=transport=dt_socket.server=y.suspend=n.address=4444 Sacurity auditing Generic JVM arguments Sacurity auditing -Dcom.ibm.xml.xkp.jaxb.opti.level=3 Bus security -Dcom.ibm.xml.xkp.jaxb.opti.level=3	Resource Environment		
□ Security Debug arguments □ Global security -agentlib ijdwp=transport=dt_socket.server=y.suspend=n.address=4444 □ Security domains Generic JVM arguments □ SEL certificate and key management -Dcom.ibm.xml.xkxp.jaxb.opti.level=3 □ Security -Dcom.ibm.xml.xkxp.jaxb.opti.level=3		Debug Mode	
Global security -agentlib.ijdwp=transport=dt_socket.server=y.suspend=n.address=4444 Sacurity domains Generic JVM arguments SSL certificate and key management -Dcom.ibm.xml.xkp.jaxb.opti.level=3 Bus security Bus security	E Security	Debug arguments	
Security domains Generic JVM arguments SSL certificate and key management -Dcom.ibm.xml.xkxp.jaxb.opti.level=3 Bus security Bus security	Global security	-agentlib:jdwp=transport=dt_socket,server=y,suspend=n,address=4444	
Administrative Authorization Groups Generic JVM arguments SSL certificate and key management -Dcom.ibm.xml.xbxp.jaxb.opti.level=3 Bus security -Discom.ibm.xml.xbxp.jaxb.opti.level=3	Security domains		
Society and the security	Administrative Authorization Groups	Generic JVM arguments	
Bus security	 SSL certificate and key management Security audities 	-Dcom.ibm.xml.xlxp.jaxb.opti.level=3	
	Buc cocurity		
	ous security		

Figure 7.3. Adjusting Maximum heap size limit

- c. Add the following parameters into the **Generic JVM arguments** field to set the perm space limit and direct memory limit:
 - -XX:MaxPermSize=512M

-XX:MaxDirectMemorySize=512M

d. Java runtime settings:

Go to Servers \rightarrow Server Types \rightarrow WebSphere application servers \rightarrow [Server_Name] (default name: server1) \rightarrow Java SDKs and select version 1.8 as the default SDK.

- e. Save the changes to configuration and restart the server so that they take effect.
- 3. Go to the download section of your CloverDX account and download the clover.war (web archive) file containing CloverDX Server for WebSphere.
- 4. Deploy the clover.war file.
 - a. Go to Integrated Solutions Console (default URL: http://localhost:9060/ibm/console/).
 - b. Go to Applications →New Application →New Enterprise Application, select the CloverDX Server WAR archive and deploy it to the application server, but do not start it yet.
- 5. Configure application class loading.

Go to WebSphere Enterprise Applications \rightarrow clover_war (or other name of the Clover application) \rightarrow Manage Modules \rightarrow CloverDX and under Class loader order select Classes loaded with local class loader first (parent last).

6. Save the changes to the Server configuration and start the clover_war application.

7. Log in CloverDX Server.

a. Provided you set clover.war as the application running with "clover" context path, use the following URL (notice the port number has changed):

http://localhost:9080/clover

- b. Activate (p. 32) the CloverDX Server.
- c. Use the default administrator credentials to access the web GUI:

Username: clover

Password: clover



Note

Please note that some **CloverDX** features using third party libraries do not work properly on IBM WebSphere.

- Hadoop is guaranteed to run only on Oracle Java 1.6+, but Hadoop developers do make an effort to remove any Oracle/Sun-specific code. See <u>Hadoop Java Versions</u> on Hadoop Wiki.
- AddressDoctor5 on IBM WebSphere requires additional JVM parameter -Xmso2048k to prevent AddressDoctor from crashing JVM. See documentation on AddressDoctor component.

Note that the parameter must be set for Worker, as well. Use the <u>worker.jvmOptions</u> (p. 91) property.

Configuration of CloverDX Server on IBM WebSphere



Tip

Default installation (without any configuration) is only recommended for evaluation purposes. For production use, at least a dedicated, system database and SMTP server configuration is recommended.

For easy configuration of **CloverDX Server**, use a Setup GUI (p. 54) in which you can configure various properties, including the connection to the database, username and password, path to the license file, private properties, clusters and much more (see Chapter 15, <u>List of Configuration Properties</u>(p. 82) and Chapter 42, <u>Cluster Configuration</u> (p. 281). We recommend you place the file in a specified (p. 50)location and specify the path to the file with a system property.

The content of such a file (an example with a PostgreSQL database):

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://127.0.0.1/clover_db?charSet=UTF-8
jdbc.username=yourUsername
jdbc.password=yourPassword
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
```

Properties File in Specified Location

The properties file is loaded from a location which is specified by the environment/system property clover_config_file or clover.config.file.

- 1. Create the cloverServer.properties file in a directory readable by IBM WebSphere. (If you need an example of connection to any of the supported database systems, see Chapter 14, <u>System Database</u> <u>Configuration</u> (p. 63).)
- 2. Set a system property (or environment variable) clover_config_file pointing to the properties file.

- a. Go to Integrated Solutions Console (default URL: http://localhost:9060/ibm/console/).
- b. Go to Servers →WebSphere application servers →[Server_name] →Java and Process Management →Process Definition →Java Virtual Machine →Custom Properties.
- c. Create a system property named clover_config_file whose value is a full path to the properties file (e.g. cloverServer.properties) on your file system.
- 3. Restart IBM WebSphere for changes to take effect.



JBoss Enterprise Application Platform

<u>Installation of CloverDX Server on JBoss EAP</u> (p. 25) <u>Configuration of CloverDX Server on JBoss EAP</u> (p. 27)



Important

Before installation, check the software requirements, currently supported JBoss EAP versions and required Java versions in the <u>Software Requirements</u> (p. 9) section.

If you encounter any problems during the installation, see Chapter 10, <u>Troubleshooting</u> (p. 42) for a possible solution.

Installation of CloverDX Server on JBoss EAP

- 1. Download and install JBoss EAP from the official download page.
- 2. It is strongly recommended to adjust the default limits for **Memory allocation** (see the <u>Memory</u> <u>Settings</u> (p. 37) section).

You can set the **initial** and **maximum memory heap size** by adjusting the "Xms" and "Xmx" JVM parameters and **classloaders memory limit** by adjusting the "XX:MaxMetaspaceSize" parameter:

For JBoss EAP standalone mode, follow these steps:

- Unix-like systems:
 - Edit the [JBoss_EAP_home]/bin/standalone.conf file.
 - Add the following line:

JAVA_OPTS="\$JAVA_OPTS -XX:MaxMetaspaceSize=512m -Xms128m -Xmx2048m"

- Windows systems:
 - Edit the [JBoss_EAP_home]\bin\standalone.conf.bat file.

JAVA_OPTS="\$JAVA_OPTS -XX:MaxMetaspaceSize=512m -Xms128m -Xmx2048m"

- 3. Go to the download section of your CloverDX account and download the clover.war (web archive) file containing **CloverDX Server** for JBoss EAP.
- 4. Configure the database connection.

By default, **CloverDX Server** uses an embedded Derby database; however, such setup is not supported for production use.

You can use the database connection provided by JNDI-bound datasource deployed by JBoss EAP. In order to define the datasource, edit the file:

[JBoss_EAP_home]/standalone/configuration/standalone.xml

and add the definition of the datasource into the section <subsystem xmlns="urn:jboss:domain:datasources:1.1"> under the element <datasources>. Here is an example of datasource connecting to a MySQL database:

```
<datasource jndi-name="java:jboss/datasources/CloverDX"</pre>
    pool-name="CloverDX-Pool" enabled="true">
    <connection-url>jdbc:mysql://localhost:3307/cloverServerDB</connection-url>
    <driver>com.mysgl</driver>
    <transaction-isolation>TRANSACTION_READ_COMMITTED</transaction-isolation>
    <pool>
        <min-pool-size>5</min-pool-size>
        <max-pool-size>50</max-pool-size>
        <prefill>true</prefill>
    </pool>
    <security>
        <user-name>clover</user-name>
        <password>SecretPassword</password>
    </security>
    <statement>
        <prepared-statement-cache-size>32</prepared-statement-cache-size>
        <share-prepared-statements>true</share-prepared-statements>
    </statement>
</datasource>
<drivers>
    <driver name="com.mysql" module="mysql.driver">
        <driver-class>com.mysgl.jdbc.Driver</driver-class>
    </driver>
</drivers>
```

5. The datasource definition references a module (mysql.driver) with the MySQL JDBC driver. Take the following steps to add the module:



Note

Under JBoss EAP, there are more options to set up **CloverDX Server**'s database: along with JNDI-bound data source, it is possible to use the embedded Derby database or other supported database system specified in the **CloverDX** configuration file.

In order to be able to connect to the database, you need to define a global module so that the driver is available for the **CloverDX** web application - copying the driver to the lib/ext directory of the Server will **not** work. Such module is created and deployed in few steps (the example is for MySQL and module's name is mysql.driver):

- a. Create a directory [JBoss_EAP_home]/modules/mysql/driver/main (note that the directory path corresponds to module name mysql.driver)
- b. Copy the driver mysql-connector-java-5.1.5-bin.jar to the directory and create a file module.xml there with the following content:

c. Add the module to global server modules: in case of the standalone JBoss EAP server they are defined in [JBoss_EAP_home]/standalone/configuration/standalone.xml. Add the module to the EE domain subsystem section:

```
<subsystem xmlns="urn:jboss:domain:ee:1.1">
<global-modules>
<module name="mysql.driver" slot="main" />
</global-modules>
<spec-descriptor-property-replacement>false</spec-descriptor-property-replacement>
```

<jboss-descriptor-property-replacement>true</jboss-descriptor-property-replacement>
</subsystem>

- 6. Configure CloverDX Server according to a description in the next section (p. 27).
- 7. Deploy WAR file.

Copy the clover.war file to [JBoss_EAP_home]/standalone/deployments.

- 8. To start the JBoss platform:
 - Unix-like systems:

Run [JBoss_EAP_home]/bin/standalone.sh.

• Windows system:

Run [JBoss_EAP_home]\bin\standalone.bat.

It may take a couple of minutes for all applications to start.

9. Log in CloverDX Server.

- a. Type http://localhost:8080/clover in the browser.
- b. Activate (p. 32) the CloverDX Server.
- c. Use the **default administrator credentials** to access the web GUI:

Username: clover

Password: clover

Configuration of CloverDX Server on JBoss EAP



Tip

The default installation (without any configuration) is only recommended for evaluation purposes. For production use, at least a dedicated, system database and SMTP server configuration is recommended.

For an easy configuration of **CloverDX Server**, use a Setup GUI (p. 54) in which you can configure various properties, including the connection to the database, username and password, path to the license file, private properties, clusters and much more (see Chapter 15, <u>List of Configuration Properties</u>(p. 82) and Chapter 42, <u>Cluster Configuration</u> (p. 281). We recommend you place the file in a specified (p. 50)location and specify the path to the file with a system property.

Properties File in Specified Location

The properties file is loaded from a location which is specified by the environment/system property clover_config_file or clover.config.file.

1. • Create the cloverServer.properties file in a directory readable by JBoss EAP. (If you need an example of connection to any of the supported database systems, see Chapter 14, <u>System Database</u> <u>Configuration</u> (p. 63).):

```
datasource.type=JNDI
datasource.jndiName=java:jboss/datasources/CloverDXServerDS
jdbc.dialect=org.hibernate.dialect.MySQLDialect
license.file=/home/clover/config/license.dat
```

Do not forget to set a correct JDBC dialect according to your database server (Part III, "<u>Configuration</u>" (p. 47)). You can set the path to the license file, too.

• Alternatively, you can set "JDBC" datasource.type and configure the database connection to be managed directly by **CloverDX Server** (provided that you have deployed proper JDBC driver module to the Server):

```
datasource.type=JDBC
jdbc.url=jdbc:mysql://localhost:3306/cloverServerDB
jdbc.dialect=org.hibernate.dialect.MySQLDialect
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.username=clover
jdbc.password=SecretPassword
license.file=/home/clover/config/license.dat
```

2. Set the clover.config.file system property (or environment variable).

It should contain the full path to the cloverServer.properties file created in the previous step.

The simplest way to set the system property is to edit the configuration file [JBoss_EAP_home]/ standalone/configuration/standalone.xml, and to add the following snippet just under the <extensions> section:

```
<system-properties>
<property name="clover.config.file" value="C:/jboss-eap-6.2/cloverServer.properties" />
</system-properties>
```

- 3. Restart JBoss EAP for the changes to take effect.
- 4. Check the CloverDX Server application is running:

By default, the Server's console is accessible at http://localhost:8080/clover.



Note

By default, JBoss EAP has enabled HTTP session replication. This requires session serialization that is not supported by **CloverDX Server** and produces lots of harmless errors in JBoss's console.

```
To eliminate these errors, disable the session replication. Edit [jboss-home]/standalone/
configuration/standalone.xml and comment out the whole <cache-container
name="web" aliases="standard-session-cache"> block under the <subsystem
xmlns="urn:jboss:domain:infinispan:1.5"> section.
```



Note

Continue with: Chapter 8, <u>Postinstallation Configuration</u> (p. 37)

Oracle WebLogic Server

Installation of CloverDX Server on Oracle WebLogic (p. 29) Configuration of CloverDX Server on Oracle WebLogic (p. 30)



Important

Before installation, check the software requirements, currently supported Oracle WebLogic versions and required Java version in the <u>Software Requirements</u> (p. 9) section.

If you encounter any problems during the installation, see Chapter 10, <u>Troubleshooting</u> (p. 42) for a possible solution.

Installation of CloverDX Server on Oracle WebLogic

1. Download and install Oracle WebLogic from the official download page.

WebLogic has to be running and a domain has to be configured. You can check it by connecting to Administration Console: http://localhost:7001/console/. Username and password are specified during installation.

2. It is strongly recommended to adjust the default limits for **Memory allocation** (see the <u>Memory Settings</u> (p. 37) section).

You can set the **initial** and **maximum memory heap size** by adjusting the "Xms" and "Xmx" JVM parameters and **classloaders memory limit** by adjusting the "XX:MaxMetaspaceSize" parameter:

• Unix-like systems:

Edit the start script and add:

export JAVA_OPTIONS='\$JAVA_OPTIONS -Xms512m -Xmx2048m -XX:MaxMetaspaceSize=512m'

• Windows system:

See <u>WebLogic Server Performance and Tuning</u>.

- 3. Go to the download section of your CloverDX account and download the clover.war (web archive) file containing **CloverDX Server** for Oracle WebLogic Server.
- 4. Change HTTP Basic Authentication configuration
 - When WebLogic finds an "Authentication" header in an HTTP request, it tries to find a user in its own realm. This behavior has to be disabled so **CloverDX** could authenticate users itself.
 - Edit the configuration file [domainHome]/config/config.xml and add:

<enforce-valid-basic-auth-credentials>false</enforce-valid-basic-auth-credentials>

into <security-configuration> element (just before the end tag).

5. Deploy clover.war (or an application directory).

Use the **WebLogic Server Administration Console**. See the <u>Oracle Fusion Middleware Administrator's Guide</u> for details.

6. Configure a license and other properties. See <u>Configuration of CloverDX Server on Oracle</u> <u>WebLogic</u> (p. 30) for details.
7. Log in CloverDX Server.

- a. Type http://localhost:7001/clover in the browser.
- b. Activate (p. 32) the CloverDX Server.
- c. Use the default administrator credentials to access the web GUI:

Username: clover

Password: clover

Configuration of CloverDX Server on Oracle WebLogic



Tip

The default installation (without any configuration) is only recommended for evaluation purposes. For production use, at least a dedicated, system database and SMTP server configuration is recommended.

For an easy configuration of **CloverDX Server**, use a <u>Setup GUI (p. 54)</u> in which you can configure various properties, including the connection to the database, username and password, path to the license file, private properties, clusters and much more (see Chapter 15, <u>List of Configuration Properties</u>(p. 82) and Chapter 42, <u>Cluster Configuration</u> (p. 281). We recommend you place the file in a <u>specified (p. 50)</u>location and specify the path to the file with a system property.

The content of such a file (example with a PostgreSQL database):

```
datasource.type=JDBC
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://127.0.0.1/clover_db?charSet=UTF-8
jdbc.username=yourUsername
jdbc.password=yourPassword
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
```

Properties File in Specified Location

1. Create the cloverServer.properties file in a directory readable by WebLogic. (If you need an example of connection to any of the supported database systems, see Chapter 14, <u>System Database</u> <u>Configuration</u> (p. 63).)

The configuration file should contain DB datasource configuration, SMTP connection configuration, etc. For details, see Part III, "<u>Configuration</u>" (p. 47).

- 2. Set the clover_config_file system property (or environment variable) pointing to the configuration properties file.
 - Set the JAVA_OPTIONS variable in the WebLogic domain start script [domainHome]/ startWebLogic.sh

JAVA_OPTIONS="\${JAVA_OPTIONS} -Dclover_config_file=/path/to/clover-config.properties

3. Restart WebLogic for changes to take effect.

Note: When **CloverDX Server** is deployed on WebLogic and JNDI Datasource pointing to Oracle DB is used, there must be an extra configuration property in the configuration file:

 ${\tt quartz.driverDelegateClass} \texttt{-} \texttt{org.quartz.impl.jdbcjobstore.oracle.weblogic.WebLogicOracleDelegateClass} \texttt{-} \texttt{org.quartz.impl.jdbcjobstore.oracle.weblogicOracleDelegateClass} \texttt{org.quartz.impl.jdbcjobstore.oracleDelegate} \texttt{org.quartz.impl.jdbcjobstore.$



Activation

To be able to execute graphs, **CloverDX Server** requires a valid license. You can install and run **CloverDX Server** without any license, but no graph will be executed.

There are three ways of installing the license. They work on all application servers and can be used at the same time, but **only the most recent valid license is used**.

We recommend using the first and easiest option (for other options, see <u>CloverDX Server Activation</u> <u>Alternatives</u> (p. 34)):

CloverDX Server Activation using Web Form

If the **CloverDX Server** has been started without assigning any license, click the **Activate server** link on the welcome page. You will be redirected to the **Add New License** form where you can upload the license file using the **Browse** button, or simply copy the license from the file and paste it into the **License key text** field.

Clover DX	Documentation Support CloverDX Home
Server is not activated. <u>Activate server</u>	
You are already logged in.	
Enter server console	
Logout	
	Data Profiler Reporting Console

Figure 7.4. Login page of CloverDX Server without license

After clicking the **Update** button, the license is validated and saved to the database table *clover_licenses*. If the license is valid, a table with license's description appears. To proceed to **CloverDX Server** console click **Continue to server console**.

You can skip adding a license by clicking the Close button.

Add New License		×
License key in a file: License key text:	+ Upload file	
		// License terms
		Add Cancel

Figure 7.5. Add new license form

Updating CloverDX Server License in the Configuration Section

If the license has been already installed, you can still change it by using form in the Server web GUI.

- Go to server web GUI \rightarrow Configuration \rightarrow Setup \rightarrow License
- Click Update license.

You can paste a license text into a **License key** text area or use the **Browse** button to search for a license file in the filesystem. To skip adding a license, click the **Close** button.

After clicking the **Update** button, the license is saved to the database table *clover_licenses* and reloaded.

Jpdate License		×
License key in License key tex	a file: + Upload file :t:	
		License terms
	Update	e Cancel

Figure 7.6. Update license form



Tip

The **CloverDX** license can be **changed** at any time by replacing license.dat file. Afterwards, you have to let **CloverDX Server** know the license has changed.

- Go to server web GUI \rightarrow Configuration \rightarrow Setup \rightarrow License
- Click Reload license.
- Alternatively, you can restart the CloverDX Server application.



Note

The license in the database is common for all nodes in the cluster. Reloading of the license occurs on each node in the cluster.

Continue with: Chapter 13, <u>Setup</u> (p. 54)

CloverDX Server Activation Alternatives

If, for any reason, you decide to not use the recommended way of installing the server license, you can choose one of the following options:

Activation Using license.file Property

- 1. Download the license.dat file.
- 2. Set the **CloverDX Server** license.file parameter to the full path of the license.dat file.

For list of properties, see Chapter 15, List of Configuration Properties (p. 82).

3. Restart the application server.

Separate License WAR

A simple approach, but it may be used only for a standalone Server running on Apache Tomcat.

- 1. Download the clover-license.war web archive file.
- 2. Copy clover-license.war to the [tomcat_home]/webapps directory.
- 3. The war file should be detected and deployed automatically without restarting Tomcat.
- 4. Check whether the license web-app is running on:

http://[host]:[port]/clover-license/ (Note: clover-license contextPath is mandatory
and cannot by changed)

Note →Continue with: Chapter 13, Setup (p. 54)

IBM InfoSphere MDM Plugin Installation

Downloading

IBM InfoSphere MDM Components for **CloverDX Server** are downloaded as a ZIP file containing the extension. The ZIP file is available for download under your account on www.cloverdx.com in **CloverDX Server** download area, under the **Utilities** section as the ibm-mdm-connectors.\${version}.zip file.

Requirements

Requirements of IBM InfoSphere MDM Components:

- supported OS are Microsoft Windows 32 bit, Microsoft Windows 64 bit, Linux 64 bit, and Mac OS X Cocoa
- at least 512MB of RAM
- installed CloverDX Server

The support for 32 bit Linux was removed in 4.5.0.

Installation into Server

The following steps are needed to install IBM InfoSphere MDM Components into CloverDX Server:

- 1. Install CloverDX Server, see its documentation for details.
- 2. Download the ZIP file with **IBM InfoSphere MDM Components** for the Server and store it on the system where **CloverDX Server** is installed. For the download instructions, see <u>Downloading</u> (p. 35).

3. The ZIP file contains a **CloverDX** plugin. Your Server installation needs to be configured to find and load the plugin from the ZIP file. This is done by setting the engine.plugins.additional.src Server configuration property to the absolute path of the ZIP file, e.g. engine.plugins.additional.src=c:/Server/ibm-mdm-connectors. \${version}.zip (in case the Server is configured via property file).

Details for setting the configuration property depend on your Server installation specifics, application server used, etc. See **CloverDX Server** documentation for details. Typically the property would be set similarly to how you set-up the properties for connection to the Server's database. Updating the configuration property usually requires restart of the Server.

4. To verify that the plugin was loaded successfully, log into the Server's **Reporting Console** and look in the **Configuration** \rightarrow **CloverDX Info** \rightarrow **Plugins** page. In the list of plugins, you should see cloverdx.engine.initiate.

Troubleshooting

If you get an Unknown component or Unknown connection error when running a graph with IBM InfoSphere MDM components, it means that the **IBM InfoSphere MDM Components** plugin was not loaded by the Server successfully. Please check the above steps to install the plugin, especially the path to the ZIP file.

Chapter 8. Postinstallation Configuration

Memory Settings (p. 37) Maximum Number of Open Files (p. 39) Maximum Number of Processes or Threads (p. 39) Firewall Exceptions (p. 39)

Memory Settings

Current implementation of Java Virtual Machine allows only a global configuration of memory for the JVM system process. Thus the whole application server, together with WARs and EARs running on it, share one memory space.

Default JVM memory settings is **too low** for running an application container with **CloverDX Server**. Some application servers, like IBM WebSphere, increase JVM defaults themselves, however they **still may be too low**.

The **optimal memory limits** depend on many conditions, i.e. transformations which **CloverDX** should execute. Please note that the maximum limit isn't the amount of permanently allocated memory, but limit which can't be exceeded. If the limit is exhausted, the OutOfMemoryError is raised.

JVM Memory Areas

JVM memory consists of several areas: **heap memory**, **PermGen space**, **direct memory** and **stack memory**. Since JVM memory is not just HEAP memory, you should not set the HEAP limit too high; in case it consumes whole RAM, JVM won't be able to allocate direct memory and stack for new threads.

Туре	Description
Heap memory	Heap is an area of memory used by JVM for dynamic memory allocation. Required heap memory size depends on various factors (e.g. complexity of graphs, number of graphs running in parallel, type of component, etc.), see the respective server container's installation guide in this documentation. (Note that current heap memory usage can be observed in CloverDX Server Console (p. 108).)
PermGen Space	Permanent Generation - separate memory space containing class definitions and related metadata. (PermGen was removed from Java 8.)
Direct Memory	Memory used by graph edges and buffers for I/O operations.
Stack Memory	Stack Memory contains local, method specific variables and references to other objects in the method. Each thread has its own stack; therefore, the memory usage depends on the number of components running in parallel.

Table 8.1. JVM Memory Structure

Configuring Memory

You can set the minimum and maximum memory heap size by adjusting the "Xms" and "Xmx" JVM parameters. There are more ways to change the settings depending on the used application container.

Recommended Server Core and Worker Heap Memory Configuration

Optimal distribution of main memory between Server Core and Worker depends on nature of executed tasks. The recommended defaults of Server Core heap size and Worker heap size for different RAM sizes are in the table below.

Heap limit is *not* a limit of the full memory used by JVM. JVM uses memory in addition to the heap size for other memory spaces, e.g. direct memory. We recommend to set the heap limit to no more than 80% of system memory size, to leave space for the operating system and other JVM memory spaces.

Chapter 8. Postinstallation Configuration

Table 8.2. Recommended Heap Memory

RAM Size	Server Core Heap	Worker Heap	Remaining for OS (estimated)
4 GB	1 GB	1 GB	1 GB
8 GB	2-3 GB	2-3 GB	2 GB
16 GB	4 - 8 GB	4 - 8 GB	4 GB
32 GB	4 - 8 GB	16 - 20 GB	8 GB
64 GB	4 - 8 GB	42 - 52 GB	8 GB

Memory Configuration in Java 8

In Java 8, the memory space for loading classes (so called "Metaspace") is separated from heap, and can be set by the JVM parameter -XX:MaxMetaspaceSize. The default maximum Metaspace size is unlimited.

Please see the specific container section for details on memory settings.

Metaspace

We recommend you to put limit on metaspace memory. Add -XX:MaxMetaspaceSize=size to command line parameters of Server Core or Worker. Replace size with a suitably high limit. 512MB should be enough.

See https://docs.oracle.com/javase/8/docs/technotes/tools/unix/java.html

Direct memory

To avoid excessive usage of direct memory, we add -Djdk.nio.maxCachedBufferSize=262144 to command line of Worker. We recommend you to add this system property also to the command line of Server Core. This system property is available since java 1.8.0_102.

In Apache Tomcat, add this system property to JAVA_OPTS environment variable, which is configured in bin/setenv.sh file.

Codecache Size

Some **CloverDX Server** installations can occasionally run into performance issue: JVM is running more than hundred times slower. The issue can be caused by a full code cache (Java SE Embedded: Developer's Guide - Codecache Tuning). The reserved code cache size is platform dependent and can be too small for **CloverDX Server**. It is highly recommended to increase the code cache size using the following JVM argument:

-XX:ReservedCodeCacheSize=256m

Maximum Number of Open Files

When using resource-demanding components, such as **FastSort**, or when running a large number of graphs concurrently, you may reach the system limit on simultaneously open files. This is usually indicated by the java.io.IOException: Too many open files exception.

The default limit is fairly low in many Linux distributions (e.g. 4096 in Ubuntu). Such a limit can be easily exceeded, considering that one FastSort component can open up to 1,000 files when sorting 10 million records. Furthermore, some application containers recommend increasing the limit themselves (8,192 for IBM WebSphere).

Therefore, it is recommended to increase the limit for production systems. Reasonable limits vary from 10,000 to about 100,000 depending on the expected load of **CloverDX Server** and the complexity of your graphs.

The current limit can be displayed in most UNIX-like systems using the **ulimit -Sn** command.

The exact way of increasing the limit is OS-specific and is beyond the scope of this manual.

Maximum Number of Processes or Threads

If you run graphs with many subgraphs containing many components, you may reach the limit on number of threads per user or per system. In this case, you can find java.lang.OutOfMemoryError: unable to create new native thread in graph's log.

The current limit on number of processes/threads per user can be displayed in most UNIX-like systems using the **ulimit -Su** command. Note that the documentation on **ulimit** may not distinguish between processes and threads. The limit on number of threads per system can be displayed using the **sysctl kernel.threads-max** command. The exact way of increasing the limit is OS-specific and is beyond the scope of this manual.

Firewall Exceptions

In order to function properly, **CloverDX Server** requires an outside communication. The table below describes both incoming and outgoing communication of **CloverDX Server**. Please, configure your firewall exceptions accordingly.

Traffic	Communi	Description & Components
Incoming	HTTP(S)	Communication between Designer and Server
	JMX	Tracking and debugging information
Outgoing	JDBC	Connection to databases (DBInputTable, DBOutputTable, DBExecute)
(dependin on an	g MX	Receiving and sending JMS messages (JMSReader, JMSWriter, JMS Listener)
actual usage)	HTTP(S)	Requesting and receiving responses from servers (Readers, WebserviceClient, HTTPConnector)
	SMTP	Sending data converted into emails (EmailSender)
	IMAP/ POP3	Receiving emails (EmailReader)
	FTP/ SFTP/ FTPS:	Remote file reading and writing (readers, writers)

Table 8.3. Firewall Exceptions

Garbage Collector for Worker

We recommend using the G1 garbage collector for Worker, as it behaves better on huge heaps and causes shorter full stops of the Java Virtual Machine. G1 is the default garbage collector in Java 9 or newer; however, for Java 8 the Parallel garbage collector is the default and G1 is just optional. Because of that, **CloverDX** automatically enables G1 garbage collector for Worker on Java 8.

If <u>worker.javaExecutable</u> (p. 92) is not specified and no specific garbage collector is selected by the <u>worker.jvmOptions</u> (p. 91) property, **CloverDX** uses the G1 garbage collector on Worker by default on Java 8. So if you modify the JVM used by Worker, we don't set G1 by default. Also, if you specify a different garbage collector, we don't override this setting.

Selecting Garbage Collector

Java 8 uses the Parallel garbage collector by default, which we override for Worker to use the G1 garbage collector. If you wish to use a different garbage collector than G1 for Worker, then it must be specified by adding the -XX:+UseParallelGC (for Parallel GC) command line option for the Worker's JVM using the worker.jvmOptions (p. 91) property.

2	Note	
\odot	Continue with:	Chapter 14, <u>System Database Configuration</u> (p. 63)

Chapter 9. Optional Installation Steps

This chapter describes optional installation steps for items not specified in the previous sections.

Adding Libraries to the Worker's Classpath

Worker may need additional libraries, e.g. a JDBC driver library or the Bouncy Castle cryptographic library. There are two ways to add the libraries.

Using default worker classpath directory

Create a worker-lib directory in the \${clover.home} directory and place the libraries there. The path set by <u>clover.home</u> (p. 82) can be found in **Server GUI** under **Configuration** —**CloverDX Info** —**Server Properties**.

Configuration Property

Create a directory containing the libraries and set the worker.classpath configuration property to the path to this directory.

Worker Support for SMB 2.x and 3.x

Worker supports the SMB 2.x and 3.x protocol. It utilizes the SMBJ library dependent on Bouncy Castle.

Before you start using SMB 2.x/3.x & Bouncy Castle:

- 1. Go to the official Latest Bouncy Castle Java Releases page.
- 2. Locate the section "SIGNED JAR FILES" and download the latest release.

Download a required . jar file. The filename consists of the name bcprov-jdk15on, followed by a version number, for example:

bcprov-jdk15on-158.jar

3. a. **Recommended** - after you download the .jar file, you can point to the directory containing the file by adding the following line:

-Dcloveretl.smb2.bouncycastle.jar.file=path/to/bcprov-jdk15on.jar

to the JVM arguments field in the Worker (p. 58) tab of Setup GUI.

- b. Alternatively create a worker-lib directory in the \${user.data.home}/CloverDX directory and move the jar file there;
- 4. Now in case of using the recommended method, you only need to restart Worker. When using the alternative method, restart **CloverDX Server** for the changes to take effect.

Chapter 10. Troubleshooting

Since **CloverDX Server** is considered a universal JEE application running on various application servers, databases and JVM implementations, problems may occur during the installation. These can be solved with a proper configuration of the Server environment. This section contains tips for the configuration.

Memory Issues on Derby (p. 42) JAVA_HOME or JRE_HOME Environment Variables Are Not Defined (p. 42) Apache Tomcat Context Parameters Do Not Have Any Effect (p. 43) Tomcat Log File catalina.out Is Missing on Windows (p. 43) clover.war as Default Context on WebSphere (Windows OS) (p. 43) Derby.system.home Cannot be Accessed (p. 44) Environment Variables and More than one CloverDX Server Instances Running on Single Machine (p. 44) Special Characters and Slashes in Path (p. 44) File System Permissions (p. 45) JMS API and JMS Third-Party Libraries (p. 45) Using an Unsupported JDBC Connector for MySQL (p. 45) illegal reflective access warning (p. 45)

Memory Issues on Derby

If your Server suddenly starts consuming too much resources (CPU, memory) despite having been working well before, it might be caused by a running internal Derby DB. Typically, causes are incorrect/incomplete shutdown of Apache Tomcat and parallel (re)start of Apache Tomcat.

Solution: move to a standard (standalone) database.

How to fix this? Redeploy CloverDX Server:

- 1. Stop Apache Tomcat and verify there are no other instances running. If so, kill them.
- 2. Backup the configuration file, if you configured any.
- 3. Delete the webapps/clover directory.
- 4. Start the Apache Tomcat server. It will automatically redeploy CloverDX Server.
- 5. Verify you can connect from Designer and from web.
- 6. Shutdown Apache Tomcat.
- 7. Restore the configuration file and point it to your regular database.
- 8. Start Apache Tomcat.

JAVA_HOME or JRE_HOME Environment Variables Are Not Defined

If you are getting this error message during an attempt to start your application server (mostly Tomcat), perform the following actions.

Linux:

This command will help you set a path to the variable on the server. [clover@server /] export JAVA_HOME=/usr/local/jdk1.x.x

As a final step, restart the application server.

Windows OS:

Set JAVA_HOME to your JDK installation directory, e.g. C:\Program Files\java\jdk1.8.0.



Important

Some **CloverDX** functions requires JDK to work correctly, therefore we do not recommend having only JRE installed.

Apache Tomcat Context Parameters Do Not Have Any Effect

Tomcat may sometimes ignore some context parameters. It may cause strange **CloverDX Server** behavior, since it appears as configured, but only partially. Some parameters are accepted, some are ignored. This issue is rare, however it may occur in some environments. Such behavior is consistent, so restart has no effect. It's possibly related to Tomcat issues: <u>Bug #47516</u> and <u>Bug #50700</u> To avoid this, please use a properties file instead of context parameters to configure **CloverDX Server**.

Tomcat Log File catalina.out Is Missing on Windows

Tomcat start batch files for Windows aren't configured to create the catalina.out file which contains the standard output of the application. The catalina.out file may be vital when Tomcat isn't started in the console and an issue occurs. Or even when Tomcat is executed in the console, it may be closed automatically just after the error message appears in it.

Please follow these steps to enable catalina.out creation:

• Modify [Tomcat_home]/bin/catalina.bat and add a parameter /B to the lines where the _EXECJAVA variable is set. There should be two such lines:

set _EXECJAVA=start /B [the rest of the line]

Parameter /B causes, that "start" command doesn't open a new console window, but runs the command in its own console window.

• Create a new startup file, e.g. [Tomcat_home]/bin/startupLog.bat, containing a single line:

catalina.bat start > ..\logs\catalina.out 2<&1</pre>

It executes Tomcat in the usual way, but the standard output isn't put to the console, but to the catalina.out file.

Then use the new startup file instead of [Tomcat_home]/bin/startup.bat.

clover.war as Default Context on WebSphere (Windows OS)

If you are deploying clover.war on the IBM WebSphere server without the context path specified, be sure to check whether it is the only application running in the context root. If you cannot start **CloverDX Server** on WebSphere, check the log and look for a following message:

```
com.ibm.ws.webcontainer.exception.WebAppNotLoadedException:
Failed to load webapp: Failed to load webapp: Context root /* is already bound.
Cannot start application CloverDX
```

The easiest way to fix the issue is to stop all other (sample) applications and leave only clover.war running on the server. That should guarantee the server will be available in the context root from now on (e.g. http://localhost:9080/).

WebSphere. software				Welcome		gout <u>IBM.</u>
View: All tasks	<u> </u>		Profile=AppSrv01			Close page
	En	nterprise	Applications			2 -
Welcome						
Guided Activities		Enterp	ise Applications			
Servers		Use this	s page to manage installed applications. A single application	can be deployed onto multiple	e servers.	
Server Types		± Pref	erences			
WebSphere application servers		Star	t Stop Install Uninstall Update Rollout Upda	te Remove File Export	Export DDL	Export File
WebSphere MQ servers						
Web servers						
Applications		Select	Name 🗘	Application Status 👲		
New Application	=	You ca	in administer the following resources:			
Application Types			DefaultApplication	8		
WebSphere enterprise applications				-		
Business-level applications Assets			clover war	47		
Global deployment settings			ivtApp	8		
Services			guery	8		
Service providers		Total 4	1			
Service clients						
Policy sets						
Trust service						
Security cache						
Reliable messaging state						

Figure 10.1. CloverDX Server as the only running application on IBM WebSphere

Derby.system.home Cannot be Accessed

If the Server cannot start and the following message is in the log:

java.sql.SQLException: Failed to start database 'databases/cloverserver'

then see the next exception for details. After that, check settings of the derby.system.home system property. It may point to an unaccessible directory, or files may be locked by another process. We suggest you set a specific directory as the system property.

Environment Variables and More than one CloverDX Server Instances Running on Single Machine

If you are setting environment variables like clover_license_file or clover_config_file, remember you should not be running more than one **CloverDX Server**. Therefore, if you ever need to run more instances at once, use other ways of setting parameters (see Part III, "<u>Configuration</u>" (p. 47) for description of all possibilities). The reason is the environment variables are shared by all applications in use causing them to share configurations and fail unexpectedly. Instead of the environment variables, you can use system properties (passed to the application container process using parameter with -D prefix: -Dclover_config_file).

Special Characters and Slashes in Path

When working with servers, be sure to follow the folder naming rules. Do not use any special characters in the server path, e.g. spaces, accents, diacritics are not recommended. It can produce issues which are hard to find. If you are experiencing weird errors and cannot trace the source of them, install the application server in a safe destination like:

C:\JBoss6\

Similarly, use slashes but never backslashes in paths inside the *.properties files, e.g. when pointing to the **CloverDX Server** license file. If you incorrectly use a backslash, it will be considered an escape character and the server may not work properly. This is an example of a correct path:

license.file=C:/CloverDX/Server/license.dat

File System Permissions

The application server must be executed by an OS user with proper read/write permissions on file system. Problem may occur, if app-server is executed by a root user for the first time, so log and other temp files are created by root user. When the same app-server is executed by another user, it will fail because it cannot write to root's files.

JMS API and JMS Third-Party Libraries

Missing JMS libraries do not cause failure of the server startup, but it is an issue of deployment on an application server, thus it is still related to this chapter.

clover.war itself does not contain jms.jar, so it has to be on an application server's classpath. Most of the application servers have jms.jar by default, but Tomcat, for example, does not. So if the JMS features are needed, the jms.jar has to be added explicitly.

If the "JMS Task" feature is used, there must be third-party libraries on a Server's classpath as well. The same approach is recommended for JMS Reader/Writer components, even if these components allow to specify external libraries. It is due to common memory leak in these libraries which causes "OutOfMemoryError: PermGen space".

Using an Unsupported JDBC Connector for MySQL

CloverDX Server requires MySQL 5 up to version 5.5 included. Using an unsupported JDBC connector for MySQL might cause an exception, for example:

```
could not execute query
You have an error in your SQL syntax; check the manual that corresponds to your MySQL s
to use near 'OPTION SQL_SELECT_LIMIT=DEFAULT' at line 1
```

Illegal Reflective Access Warning

When running **CloverDX Server** on Java 9 and later, an illegal reflective access warning may occur. For example, below is the warning which appears in the log when Worker is starting:

2018-10-09 15:54:14,356[threadPool-1] WorkerProcess ERROR [worker0@node01]106

The warning is a result of changes made in Java 9. (see <u>Understanding Runtime Access Warnings</u>) and has no effect on the function of **CloverDX** applications.

Note: The solution below applies only to the **CloverDX Server** + **Tomcat 9** bundle. The procedure may differ per application server, so if you run another supported application server and want to suppress the warnings, a combination of the flags (both Tomcat 9 default and those added manually) may be required. The changes must be done by your server administrator.

In CloverDX Server bundled with Tomcat 9, the warning has been disabled by adding the following flags:

to JDK_JAVA_OPTS in setenv.sh/setenv.bat:

```
--add-opens=java.base/java.util=ALL-UNNAMED
--add-exports=java.xml/com.sun.org.apache.xerces.internal.parsers=ALL-UNNAMED
```

on Worker's command line:

```
--add-opens=java.base/java.lang=ALL-UNNAMED
--add-opens=java.base/java.util=ALL-UNNAMED
--add-exports=java.xml/com.sun.org.apache.xerces.internal.parsers=ALL-UNNAMED
--add-opens=java.rmi/sun.rmi.transport=ALL-UNNAMED
```

Tomcat 9 default flags in catalina.sh/catalina.bat:

JDK_JAVA_OPTIONS="\$JDK_JAVA_OPTIONS --add-opens=java.base/java.lang=ALL-UNNAMED" JDK_JAVA_OPTIONS="\$JDK_JAVA_OPTIONS --add-opens=java.rmi/sun.rmi.transport=ALL-UNNAMED" export JDK_JAVA_OPTIONS

Part III. Configuration

Chapter 11. Introduction

This part describes in detail the configuration options for **CloverDX Server** used in a production environment. In the following chapters, you will find information on setting required properties and parameters, description of **CloverDX Server**'s Setup GUI elements, parameters for specific database configuration, list of properties used in general configuration, instructions on encrypting confidential properties and log files setting.



Note

We recommend the default installation (without any configuration) only for evaluation purposes. For production use, you should configure a dedicated, system database and set up an SMTP server for sending notifications.

CloverDX Server Configuration Procedure

For initial configuration after the installation of **CloverDX Server**, follow these steps:

• Choose a configuration source

Choose a source of configuration data for **CloverDX Server**. There are several options for configuration sources; however, we recommend using a property file on specified location. For more information, see Chapter 12, <u>Configuration Sources</u> (p. 50).

• Set up a database dedicated to CloverDX Server

Now, you should set up a **CloverDX Server**'s database and configure a connection to the database. Choose a supported <u>database system (p. 10)</u> and read Chapter 14, <u>System Database Configuration</u>(p. 63) for information and examples on how to create a database, add user/role for Clover, grant it required rights/ privileges, etc.

Once you have set up the database, configure **CloverDX Server**'s connection to the database using the <u>Setup</u> <u>GUI (p. 54)</u>.

• Activate the server with a license

After you have set up the database, configured the connection to it and specified the source for configuration data, you can activate the Server with your license. While it is possible to activate the Server immediately after installation, we do not recommend this, since after the activation, the license information is stored in the database. To activate the Server, follow the information in the <u>Activation</u> (p. 32) section.

• Configure the server

Finally, you can configure the server features. The Setup with a user friendly GUI allows you to configure the basic, most important features including, encryption of sensitive data, SMTP for email notifications, etc. For more information, see Chapter 13, <u>Setup</u> (p. 54).

Configure Worker

Worker is the executor of jobs, all jobs run in the Worker by default. It runs in a separate process (JVM), so it requires configuration in addition to the Server Core.

Basic configuration of the Worker can be done in the <u>Worker</u> (p. 58) page of the Setup (p. 54) For all configuration properties of Worker see <u>Worker - Configuration Properties</u>(p. 90) See the Chapter 26, <u>Troubleshooting Worker</u> (p. 165) section for useful tips on solving issues.

The following are the typical areas that need to be configured for Worker:

• Heap memory size - required heap size for Server Core and Worker depends on the nature of executed jobs. In general, Worker should have higher heap allocated, as it runs the jobs which represent the bulk of memory consumption.

See our recommendations (p. 37) for heap sizes of Worker and Server Core.

Heap size of Worker can be easily configured in the Worker (p. 58)tab of Setup (p. 54) or via the worker.maxHeapSize (p. 91) configuration property.

• **Classpath** - the Worker's classpath is separate from Server Core (i.e. application container classpath). Any libraries needed by jobs executed on Worker need to be added on the Worker's classpath.

See the worker.classpath (p. 90) configuration property for more details.

• **Command line options** - Worker is started as a separate process with its own JVM. If you need to set JVM command line options, e.g. for garbage collector (p. 40) tweaking, better diagnostics, etc., then you need to set them on the Worker's JVM. See Additional Diagnostic Tools (p. 164) section for useful options for troubleshooting and debugging Worker.

Command line options of Worker can be easily customized in the Worker (p. 58)tab of Setup (p. 54) or via the worker.jvmOptions (p. 91) configuration property.

• JNDI - Worker has its own JNDI pool separate from the application container JNDI pool. If your jobs use JNDI resources (to obtain JDBC or JMS connections), you have to configure the Worker's JNDI pool and its resources.

See JNDI in worker (p. 92) section for more details.

• Encrypt the configuration

As the last step, we strongly recommend you to encrypt the configuration file to protect your sensitive data.

For more information, see the Encryption (p. 59) section.



Figure 11.1. CloverDX Server's System Database Configuration

Chapter 12. Configuration Sources

<u>Configuration File on Specified Location</u> (p. 50) <u>Alternative Configuration Sources</u> (p. 50) <u>Priorities of Configuration Sources</u> (p. 51) <u>Specifying the Path to the Configuration File</u> (p. 52)

Once installed, **CloverDX Server** requires the configuration of essential (database connection, license, sandboxes) and optional (SMTP, LDAP, etc.) features. The configuration is done by specifying <u>configuration</u> <u>properties (p. 82)</u> in a [property-key]=[property-value] format.



Important

The configuration might contain sensitive data; therefore, **CloverDX Server** enables you to encrypt the configuration properties (for more information, see Chapter 16, <u>Secure Configuration</u> <u>Properties</u> (p. 101)).

CloverDX can load the configuration properties from several sources. We recommend the easiest, most convenient way:

Configuration File on Specified Location

The cloverServer.properties configuration file is a text file which contains all **CloverDX** settings. You can edit the file either manually or using a much **simpler and intuitive** <u>Setup GUI (p. 54)</u>.

Example of configuration file's content for PostgreSQL database:

```
# Modify the url, username and password for your environment.
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://hostname/clover_db?charSet=UTF-8
jdbc.username=user
jdbc.password=pass
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
```

The path to the file can be specified by system property, context parameter or environment variable.

We recommend specifying the path with the clover.config.file system property.

Example for Apache Tomcat:

```
Editbin/setenv.sh(orbin\setenv.bat) and add -Dclover.config.file=/absolute/path/to/cloverServer.properties to CATALINA_OPTS.
```

For more information and examples on the supported application containers, see <u>Specifying the Path to the</u> <u>Configuration File</u> (p. 52).

Alternative Configuration Sources

There are other sources of configuration properties, as well. Each source containing the configuration data has a different priority. If a property isn't set, application's default setting is used.



Warning

Combining configuration sources could lead to a confusing configuration which would make maintenance much more difficult.

• Environment Variables

Environment variables are variables configured by means of your operating system (e.g. \$PATH is an environment variable).

Set an environment variable with the clover. prefix, i.e. (clover.config.file).

Some operating systems may not use a dot (.) character, so underlines (_) may be used instead of dots. So the clover_config_file name works as well.

• System Properties

System properties are configured by means of JVM, i.e. with the -D argument (-Dclover.config.file).

Set a system property with the clover. prefix, i.e. (clover.config.file).

Underlines (_) may be used instead of dots (.) so the clover_config_file name works as well.

• Configuration File on Default Location

A text file containing configured **CloverDX** properties. By default, **CloverDX** searches for the file on the [AppServerDir]/cloverServer.properties path.

Modification of Context Parameters in web.xml

This way **isn't recommended**, since it requires a modification of the WAR file, but it may be useful when none of the approaches above are possible.

Unzip clover.war and modify the WEB-INF/web.xml file. Add the following piece of code into the file:

```
<context-param>
<param-name>[property-name]</param-name>
<param-value>[property-value]</param-value>
</context-param>
```

• Context Parameters (Available on Apache Tomcat)

Some application servers allow you to set context parameters without modification of the WAR file.

This way of configuration is possible, but it is **not recommended**, as Apache Tomcat may ignore some context parameters in some environments. Using the configuration file is almost as convenient and much more reliable.

Example for Apache Tomcat:

On Tomcat, it is possible to specify context parameters in a context configuration file [Tomcat_home]/ conf/Catalina/localhost/clover.xml which is created automatically just after deployment of the CloverDX Server web application.

You can specify a property by adding this element:

<Parameter name="[propertyName]" value="[propertyValue]" override="false" />

(Note: by setting the override attribute to false), the context parameter does not override the default setting associated with the owning host.)

Priorities of Configuration Sources

Configuration sources have the following priorities (from the highest to lowest):

1. Context parameters

Context parameters are specified in an application server or directly in a web.xml file (not recommended).

2. External configuration file

The path to the external configuration file can be specified in several ways. CloverDX Server attempts to find the file in this order (only one of them is loaded):

- a. the path specified with a config.file context parameter;
- b. the path specified with a clover_config_file or clover.config.file system property (recommended);
- c. the path specified with a clover_config_file or clover.config.file environment variable;
- d. the default location ([AppServerDir]/cloverServer.properties).

3. System properties

- 4. Environment variables
- 5. Default values

Specifying the Path to the Configuration File

Setup (p. 54) uses the configuration file to save the Server's settings. The path to the file is specified by the clover.config.file system property. Each application server has a different way to configure the path:

• Apache Tomcat

Edit bin/setenv.sh (or bin/setenv.bat) and add -Dclover.config.file=/absolute/path/to/cloverServer.properties to CATALINA_OPTS.

See also Apache Tomcat (p. 17).

• JBoss Enterprise Application Platform

Edit the configuration file /standalone/configuration/standalone.xml and add the following snippet just under the <extensions> section:

```
<system-properties>
<property name="clover.config.file" value="C:/jboss-eap-6.2/cloverServer.properties" />
</system-properties>
```

See also <u>JBoss Enterprise Application Platform</u> (p. 25).

• IBM WebSphere

- 1. Go to Integrated Solutions Console (default URL: http://localhost:9060/ibm/console/).
- 2. Go to Servers →WebSphere application servers →[Server_name] →Java and Process Management →Process Definition →Java Virtual Machine →Custom Properties.
- 3. Create a system property named clover_config_file whose value is a full path to the properties file (e.g. cloverServer.properties) on your file system.

See also IBM WebSphere (p. 21).

• Weblogic

Set JAVA_OPTIONS variable in the WebLogic domain start script [domainHome]/startWebLogic.sh

JAVA_OPTIONS="\${JAVA_OPTIONS} -Dclover_config_file=/path/to/clover-config.properties

See also Oracle WebLogic Server (p. 29).

Note



Continue with: Chapter 14, <u>System Database Configuration</u> (p. 63)

Chapter 13. Setup

Before You Start (p. 54) Using Setup (p. 55)

While it is possible to configure **CloverDX Server** by modifying the configuration file (p. 56) in a text editor, the **Setup** with a user-friendly GUI offers a much easier way of configuring basic properties according to your preferences and requirements. Setup is accessible from **Server Console** under **Configuration** \rightarrow **Setup**.

It lets you configure:

- License (p. 57)
- Database Connection (p. 57)
- Worker (p. 58)
- Sandbox Paths (p. 58)
- Encryption (p. 59)
- E-mail (p. 59)
- LDAP Connection (p. 60)
- Cluster Configuration (p. 61)

Remember that you should create a database for **CloverDX Server** and add a user/role for **Clover** with appropriate rights **before** you set up the connection to the database in the server's Setup GUI.

CloverDX Server requires a working database connection for storing license information. Therefore, it allows you to access the Setup and configure the connection **prior** the Server **activation** - simply log in the Server Console and click the **Close** button. Otherwise, you would have to activate the server again, after switching from Derby to a new system database.



Important

To access the Setup section, you need the Server Setup permission (p. 142).



Tip

To keep your settings and data in case of a database migration (e.g. from evaluation to production environment), see Chapter 23, <u>Server Configuration Migration</u> (p. 156).

Before You Start

Before you start using the Setup, you have to specify the path to the configuration file where the Setup saves the settings, and add required libraries to the classpath.

1. Specify the Path to the Configuration File

Setup uses the configuration file to save the server's settings. The path to the file is specified by the clover.config.file system property. Each application server has a different way to configure it; for more information, see the <u>Specifying the Path to the Configuration File</u> (p. 52) section.

2. Add Libraries to the Classpath

As the next step, you should place the libraries required for further configuration on the application server's classpath (in most cases, this is done by placing the files into a specific directory). You usually need a JDBC driver for the connection to the database or a .jar file with an encryption provider.

For example, in case of Apache Tomcat, copy the libraries to the TOMCAT/lib/ folder. Then, restart the application server.

3. Configure Particular Items

Now, you can use the Setup to configure the rest of the Server's features (see the following section). The configuration is then saved into a file defined in the clover.config.file property.

If you wish to encrypt the sensitive data in you configuration file, set up the Encryption first.

4. Restart the Server (if needed)

Some changes require you to restart the Server. These changes are indicated by the z icon. Other changes (e.g. License, Sandboxes, E-mail, LDAP) are applied immediately and do not require a restart.

Using Setup

Each setup page consists of a menu with setup tabs on the left, a main configuration part in the middle and a configuration status and text on the right side.

The main configuration part contains several buttons:

- Save saves changes made to the configuration file. The changes in the configuration must be valid.
- Save Anyway saves the configuration even if it is invalid. For example, a database connection is considered invalid if a required library is missing.
- Validate validates the configuration on a current tab. If you see the Save button disabled, use Validate to validate the configuration first.
- Discard Changes discards unsaved changes and returns to currently used values.

The following icons can appear in the Server GUI:

- ✓ configured tab
- inactive tab
- 🔺 error
- 🔺 warning
- c restart required

pending changes which have not been saved yet

If an error/warning icon appears, a status message on the right side of the Setup GUI will provide relevant details.

If you start the Server without configuration, you will see decorators pointing to the Setup. The decorators mark problems which require your attention. The displayed number corresponds to the number of items.



Figure 13.1. Setup GUI with decorators

Configuration File

The **Configuration** tab displays the content of the configuration file. For basic setting, you do not have to edit the content of the file manually. Instead, use the particular Setup tab to configure the corresponding subsystem. For advanced setting, see Chapter 15, <u>List of Configuration Properties</u> (p. 82).

Configuration File

Configuration file of the CloverDX Server that stores basic settings. Most of the settings defined in the other Setup sections are stored in this file.



Figure 13.2. Example of the Server Configuration file

License

The License tab lets you add/update, show or reload the license. The type of the license determines what CloverDX Server's features are activated.



Important

The license is **stored in the database**; therefore, you should configure the database prior activating the Server with a license. Otherwise, you will have to specify the license again.

Similarly, if you change the database, you will be prompted to re-activate the Server.

_icense					
CloverDX Server license gr erver installation.	ants you th	e ability to use the	server and specifies	capabilitie	s of this
VAILABLE LICENSES					
License number Comp	any name	Products	License storage	Expirati(date	Active
Javlin	(internal)	Server Cluster(5.0.x); Data Quality(5.0.x)	DATABASE	22.1.201	0
 Show detail 				0-1	
License number	CLECKAR	v1,60396098EV	License r	name Sei Clu	ster
Info			Company add	lress	
Version	5.0.x		Days until expir	ation 85	
Max allowed CPUs	32		Plat	form	
Jobflow			Scheduling allo	owed 📀	
Graph event listener allowed	0		Jobflow event list allo	tener 🕑	
File event listener allowed	0		JMS message e listener alle	owed 🕑	
Groovy event listener allowed	0		Task event list allo	tener 🕑	
Data partitioning allowed	I		Clustering all	owed 🗸	
Max cluster nodes	4		Launch Services allo	s API	
Simple HTTP API allowed	0		JMX API allo	owed 📀	
Data Service allowed	v				
Update License St	low License	e Reload Lice	nse		

Figure 13.3. The License tab

Database

The Database tab lets you configure the connection to the database. You can connect via:

• JDBC

Under JDBC connection, choose your **Database** from the first drop-down list. This will enable you to choose a **Database URL** template from the second drop-down list. In this template (e.g. jdbc:postgresql:// host:5432/dbname for PostgreSQL), replace the *host* and *dbname* keywords with proper values.

Next, enter the User name and Password for your database (default User name/Password: clover/clover).



Note

An Apache Derby JDBC 4 compliant driver is bundled with **CloverDX Server**. If you use another database system, add a JDBC 4 compliant driver on the classpath .

• JNDI

With JNDI, you can access the Datasource on an application server level. Select your **Database** and choose a suitable item from a JNDI tree For more information, see <u>JNDI DB Datasource</u> (p. 75).

Database	Database
Database is used to store CloverDX Server's configuration (users, event listeners, etc.) and runtime state information (running jobs, task logs, etc.).	Database is used to store CloverDX Server's configuration (users, event listeners, etc.) and runtime state information (running jobs, task logs, etc.).
CONNECTION TYPE	CONNECTION TYPE
JDBC connection	JDBC connection
JNDI data source	JNDI data source
CONNECTION SPECIFICATION	CONNECTION SPECIFICATION
Database PostgreSQL 🔻	Database PostgreSQL -
Database URL	JNDI data source name Doubleclick on the data source in JNDI tree
User name	C Discard Changes B Save Validate
Password	
C Discard Changes Save Validate	

Figure 13.4. Database connection configuration for JDBC (left) and JNDI (right)

Worker

The Worker tab lets you configure Worker properties.

You can change **Initial** and **Maximum heap size** for Worker, include **JVM arguments** and set **Port range** for communication with Worker.

Changes in Worker require restart - click on the Actions button in the right pane and select Finish jobs and restart or Restart now.

his subsustant of Olaver	V Conver is reenensible fo	a munina inha	
nis subsystem of CloverL	DX Server is responsible to	r running jobs.	
EMORY SETTINGS			
Initial heap size (MB)	2048	0	
Maximum heap size (MB)	4096	0	
/M ARGUMENTS			
			0
ONNECTION			
Port range 10500	- 10600 (3)		

Figure 13.5. The Worker tab

Sandboxes

The **Sandboxes** tab lets you configure a path to shared (p. 277) local (p. 277) and partitioned (p. 278) sandboxes.

Sandboxes	Sandboxes		
Sandboxes store project's files such as graphs, metadata, input and output data, etc.	Sandboxes store project's files such as graphs, metadata, input and output data, etc		
SANDBOX LOCATION	SANDBOX LOCATION		
Sandboxes home C:\Users\clover\CloverDXServer.5.0.2.Tomcat-9.0.10/sandboxes 0	Shared sandboxes home	/home/javlin/sandboxes/cluster_shared	0
Resolved path: C:/Users/clover/CloverDXServer.5.0.2.Tomcat-9.0.10/sandboxes		Resolved path: /home/javlin/sandboxes/cluster_shared	
C Discard Changes D Save Validate	Local sandboxes home	\${clover.home}/sandboxes-local	0
		Resolved path: /home/javlin/CloverETL/sandboxes-local	
	Partitioned sandboxes	\${clover.home}/sandboxes-partitioned	0
	nome	Resolved path: /home/javlin/CloverETL/sandboxes-partitioned	

Figure 13.6. Sandbox path configuration with clustering disabled (left) and enabled (right)

Encry	ntion
	μισπ

To secure your sensitive data in the configuration file, you can use the Encryption feature. In the **Encryption** tab, choose a desired **Encryption provider** and **Encryption algorithm**; among the default algorithms, the *PBEWithSHA1AndDESede* is the strongest.

However, since the default algorithms are generally weaker, we **recommend** using <u>Bouncy Castle</u> - a free custom JCE (Java Cryptography Extension) provider offering a **higher strength of encryption**.

Note that in case you use a custom provider, the libraries have to be added on the classpath (in the same way as database libraries).

Encryption			
Passwords and other s	ensitive information can be	encrypted in the con	figuration file.
 Enable encryption 			
ENCRYPTION CONFIGURAT	10N		
Encryption provider	SunJCE	•	•
Encryption algorithm	PBEWithMD5AndDES	•	
C Discard Changes	🖬 Save 🖬 Save & I	Encrypt Validat	е

Figure 13.7. Encryption configuration

E-Mail

The **E-mail** tab lets you configure a connection to an SMTP server so the **CloverDX Server** can send an email (p. 173) reporting the server status / events on the server.

To make sure the configuration of **Outgoing SMTP Server** is correct, you can check that the SMTP server can be reached by sending a **test email** from the dialog.

Email			
CloverDX Server can send	d email notifications about	various events, such	as a failed graph.
 Enable SMTP connect 	ion		
OUTGOING SMTP SERVER			
Mail protocol	smtn	•	
SMTP boot	omp		
SWIP nost	25		
IP port	25		
Network timeout (ms)	5000		
User name			
Password			
Additional properties	Name	Value	0
TEST EMAIL			
Ŧ			
10			
From	Administrator		
Subject	CloverDX Setup SMTP Te	st	
Message text	This is a testing message Server Setup.	from CloverDX	
	Send Email		
C Discard Changes	Save Validate		

Figure 13.8. E-mail configuration

LDAP

The LDAP tab lets you use an existing LDAP database for user authentication.

Specify a connection to the LDAP server and define a pattern for user DN. **Login Test** allows you to validate the login using any user matching the pattern.

For more information, see LDAP Authentication (p. 125).

LDAP Setup		
CloverDX Server can authenticate users against an LDAP directory.		
✓ Enable LDAP authentication		
AUTHENTICATION POLICY		
Use LDAP for user authentication only		
Use LDAP for user authentication and user synchronization		
CONNECTION SPECIFICATION		
Context factory com.sun.jndi.ldap.LdapCtxFactory		
LDAP host Idap.forumsys.com		
IP port 389		
Use encryption (SSL)		
Referral processing ignore		
USER AUTHENTICATION		
User DN pattern uid=\${username},dc=example,dc=com		
Example: uid=\${username},ou=employees,dc=company,dc=com or just \${username}		
LOGIN TEST		
User name		
Password		
Validate Login		
C Discard Changes Save		

Figure 13.9. LDAP configuration

Cluster

The **Cluster** tab lets you configure clustering features.

In case your license does not allow clustering, the Enable clustering checkbox is grayed out and the note The license does not allow clustering. appears at the top.

For more information, see Part VII, "<u>Cluster</u>" (p. 276).

Cluster			
CloverDX Server can dist reliability.	ibute workload within a c	luster of servers for high	er performance and
 Enable clustering 			
CLUSTER IDENTIFICATION -			
Cluster group name	cloverCluster	0	
NODE IDENTIFICATION			
Cluster node ID	node01	0	
This node URL			- 0
Bind address	127.0.0.1	• 0	
IP port	7800		

Figure 13.10. Cluster configuration

Chapter 14. System Database Configuration

The **CloverDX Server** license, as well as user's information, event listeners and other services, are saved in a database. For stability and performance reasons, the default Apache Derby database is **not** supported for production environment; therefore, you should choose one of the supported DB systems.



Important

Since **CloverDX Server** stores important data in a database, you should create a system database and set up a working connection **before** you activate the Server with license and configure it.

For details on how to set up a connection to an external system database, see the list of examples below. The examples contain details on creating databases in DB systems supported by **CloverDX Server** and configuring a working connection between the database and the Server.

It is possible to specify common **JDBC** DB connection properties (see below) or a **JNDI** location of DB Datasource.

Clustered Deployment

In a clustered deployment, at least one node in the cluster must have a DB connection configured. Other nodes may have their own direct connection (to the same DB), or may use another node as a proxy for persistent operations; however, the scheduler is active only on nodes with a direct connection. For more information about the feature, see Part VII, "<u>Cluster</u>" (p. 276).

Setting up a CloverDX Server's System Database

1. Create a database

• Choose one of the supported database systems and create a database dedicated to **CloverDX Server**. Add a user/role for Clover and grant it required rights/privileges.

2. Configure common JDBC connection properties

• Some JDBC connection properties are common for all supported database systems. If you use a **properties file** for configuration, specify these properties:

jdbc.driv	erClassName Class name for JDBC driver name.	
jdbc.url	JDBC URL used by CloverDX Server to store data.	
jdbc.user	name JDBC database username.	
jdbc.pass	jdbc.password JDBC database password.	
jdbc.dial	ect Hibernate dialect to use in Object-relational mapping (ORM).	

3. Add a JDBC 4 compliant driver on the classpath.

• As the last step, add a JDBC 4 compliant driver on the classpath. A JDBC Driver which doesn't meet JDBC 4 won't work properly.

Below is a list of examples of individual database systems configurations.

Examples of Database Configurations

- <u>Embedded Apache Derby</u> (p. 65)
- <u>MySQL</u> (p. 66)

- <u>DB2</u> (p. 67)
 <u>Oracle</u> (p. 70)
 <u>Microsoft SQL Server</u> (p. 71)
 <u>PostgreSQL</u> (p. 73)
- JNDI DB Datasource (p. 75)

For officially supported versions of particular database systems, see <u>Database servers</u> (p. 10).

Embedded Apache Derby

The Apache Derby embedded DB is used with a default CloverDX Server installation. It uses the working directory as a storage for data persistence by default. This may be a problem on some systems. In case of any problems with connecting to Derby DB, we recommend you configure a connection to external DB or at least specify the Derby home directory:

Configure the derby.system.home system property to set path which is accessible for application server. You can specify this system property with this JVM execution parameter:

-Dderby.system.home=[derby_DB_files_root]

Example of a properties file configuration:

```
jdbc.driverClassName=org.apache.derby.jdbc.EmbeddedDriver
jdbc.url=jdbc:derby:databases/cloverDb;create=true
jdbc.username=user
jdbc.password=password
jdbc.dialect=com.cloveretl.server.dbschema.DerbyTableIdDialect
```

Take a closer look at the jdbc.url parameter. The databases/cloverDb part means a subdirectory for DB data. This subdirectory will be created in the directory which is set as derby.system.home (or in the working directory if derby.system.home is not set). You may change the default value databases/cloverDb.

A Derby JDBC 4 compliant driver is bundled with CloverDX Server, thus there is no need to add it on the classpath.



Continue with: Encrypted JNDI (p. 76) or Activation (p. 32)
MySQL

```
<u>Creating Database</u> (p. 66)
<u>CloverDX Server Setup</u> (p. 66)
```

CloverDX Server supports MySQL 5, up to version 5.6 included.

Creating Database

The following steps will create a clover_db database and the clover user with clover password.

1. Create database clover_db, set charset and collate.

CREATE SCHEMA clover_db CHARACTER SET utf8 COLLATE utf8_unicode_ci;

2. Use clover_db as the current database.

```
USE clover_db;
```

3. Create a new user with password and host.

CREATE USER 'clover'@'%' IDENTIFIED BY 'clover';

4. Add all privileges to user 'clover' in DB clover_db.

GRANT ALL ON clover_db.* TO 'clover'@'%';

5. Reload privileges.

FLUSH privileges;

CloverDX Server Setup

Example of a properties file configuration:

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://127.0.0.1:3306/clover_db?useUnicode=true&characterEncoding=utf8
jdbc.username=clover
jdbc.password=clover
jdbc.dialect=org.hibernate.dialect.MySQLDialect
```

Add a JDBC 4 compliant driver on the classpath. A JDBC Driver which doesn't meet JDBC 4 won't work properly.



DB2

<u>Creating Database</u> (p. 67) <u>CloverDX Server Setup</u> (p. 67) <u>Troubleshooting</u> (p. 68) <u>DB2 on AS/400</u> (p. 69)

Creating Database

1. Create a dedicated user for the CloverDX database and set a password (UNIX/Linux).

useradd clover

passwd clover

2. Create a new database.

db2 "CREATE DATABASE cloverdb PAGESIZE 32768 RESTRICTIVE"

3. Activate the database.

db2 activate db cloverdb

4. Connect to the database.

db2 connect to cloverdb

5. Grant the user DBADM authority (DBADM authority is an administrative authority for a specific database. The database administrator possesses the privileges that are required to create objects and issue database commands. By default, DATAACCESS and ACCESSCTRL authority are also granted).

db2 "GRANT DBADM ON DATABASE TO USER clover"

6. Disconnect from database

db2 connect reset

CloverDX Server Setup

Example of a properties file configuration:

```
jdbc.driverClassName=com.ibm.db2.jcc.DB2Driver
jdbc.url= jdbc:db2://localhost:50000/clover
jdbc.username=clover
jdbc.password=clover
jdbc.dialect=org.hibernate.dialect.DB2Dialect
```

Add a JDBC 4 compliant driver on the classpath. A JDBC driver which doesn't meet JDBC 4 specifications won't work properly.

Troubleshooting

Wrong pagesize

The *cloverdb* database has to be created with suitable PAGESIZE. DB2 has several possible values for this property: 4096, 8192, 16384 or 32768.

CloverDX Server should work on DB with PAGESIZE set to 16384 or 32768. If the PAGESIZE value is not set properly, there should be an error message in the log file after failed **CloverDX Server** startup.

The error indicating wrong pagesize:

```
ERROR:
DB2 SQL Error: SQLCODE=-286, SQLSTATE=42727, SQLERRMC=16384;
ROOT, DRIVER=3.50.152
```

SQLERRMC contains suitable value for PAGESIZE.

Solution:

You can create a database with a proper page size using the PAGESIZE command, e.g.:

CREATE DB clover PAGESIZE 32768;

The table is in the reorg pending state

On rare occasions, the ALTER TABLE commands may cause tables to remain in "reorg pending state". This behavior is specific for DB2. The ALTER TABLE DDL commands are executed only during the first start of a new **CloverDX Server** version.

The issue may return the following error messages:

```
Operation not allowed for reason code "7" on table "DB2INST2.RUN_RECORD"..
SQLCODE=-668, SQLSTATE=57016
```

DB2 SQL Error: SQLCODE=-668, SQLSTATE=57016, SQLERRMC=7;DB2INST2.RUN_RECORD, DRIVER=3.5

In this case, the "RUN_RECORD" table is in the "reorg pending state" and "DB2INST2" is the DB instance name.

Solution:

Go to DB2 console and execute the following command (for table run_record):

reorg table run_record

DB2 console output should look like this:

```
db2 => connect to clover1
Database Connection Information
Database server = DB2/LINUX 9.7.0
SQL authorization ID = DB2INST2
Local database alias = CLOVER1
db2 => reorg table run_record
DB20000I The REORG command completed successfully.
db2 => disconnect clover1
DB20000I The SQL DISCONNECT command completed successfully.
```

"clover1" is DB name

DB2 does not allow ALTER TABLE which trims DB column length.

This problem depends on the DB2 configuration and has been experienced only on some AS400s, so far. **CloverDX Server** applies a set of DP patches during the first installation after the application upgrade. Some of these patches may apply column modifications which trim the length of the text columns. These changes never truncate any data, however DB2 does not allow this since it "may" truncate some data. DB2 refuses these changes even in empty DB table.

Solution:

Disable the DB2 warning for data truncation, restart **CloverDX Server** which applies patches, then enable DB2 warning again.

DB2 on AS/400

The connection on AS/400 might be slightly different.

Example of a properties file configuration:

```
jdbc.driverClassName=com.ibm.as400.access.AS400JDBCDriver
jdbc.username=user
jdbc.password=password
jdbc.url=jdbc:as400://host/cloversrv;libraries=cloversrv;date format=iso
jdbc.dialect=org.hibernate.dialect.DB2400Dialect
```

Use credentials of your OS user for jdbc.username and jdbc.password.

cloversrv in jdbc.url above is the name of the DB schema.

You can create the schema in AS/400 console:

- 1. execute command STRSQL (SQL console)
- 2. execute command

CREATE COLLECTION cloversrv IN ASP 1

cloversrv is the name of the DB schema and it may be at most 10 characters long

Proper JDBC driver must be in the application server classpath.

Use jt400ntv.jar JDBC driver found in /QIBM/ProdData/Java400 on the server.

Add a JDBC 4 compliant driver on the classpath. A JDBC driver which doesn't meet JDBC 4 specifications won't work properly.



Note

Continue with: <u>Encrypted JNDI</u> (p. 76) or <u>Activation</u> (p. 32)

Oracle

<u>Creating Database</u> (p. 70) <u>CloverDX Server Setup</u> (p. 70)

Creating Database

Run the following script to create a role (cloverRole), user (cloverUser) with password (cloverPassword) and tablespace for **CloverDX Server**:

```
-- Create a new role and grant it privileges.
CREATE ROLE cloverRole NOT IDENTIFIED;
GRANT CREATE SESSION TO cloverRole;
GRANT CREATE TABLE TO cloverRole;
GRANT CREATE TRIGGER TO cloverRole;
GRANT CREATE TRIGGER TO cloverRole;
-- Create a new database user with password.
CREATE USER cloverUser IDENTIFIED BY cloverPassword;
-- Set quota on tablespace.
GRANT UNLIMITED TABLESPACE TO cloverUser;
-- Connect a new role to a new user.
GRANT cloverRole TO cloverUser;
```

CloverDX Server Setup

Example of a properties file configuration:

```
jdbc.driverClassName=oracle.jdbc.OracleDriver
jdbc.url=jdbc:oracle:thin:@host:1521:db
jdbc.username=cloverUser
jdbc.password=cloverPassword
jdbc.dialect=org.hibernate.dialect.Oracle10gDialect
```

Add a JDBC 4 compliant driver on the classpath. A JDBC driver which doesn't meet the JDBC 4 specifications won't work properly.

These are the privileges which have to be granted to a schema used by CloverDX Server:





Microsoft SQL Server

<u>Creating Database</u> (p. 71) <u>CloverDX Server Setup</u> (p. 71)

Creating Database

It is advised to use SQL Server Authentication instead of Windows Authentication. To enable it, select the server instance in Microsoft SQL Server Management Studio: go to **Properties** \rightarrow **Security** \rightarrow **Server authentication** and select the SQL Server and Windows Authentication mode. The server instance needs to be restarted.



Note

Make sure you have:

- TCP/IP Enabled in SQL Server Network Configuration → Protocols
- TCP Port set to **1433** in TCP/IP **Properties** →**IP** Addresses →**IP**All
- 1. Create a new database

CREATE DATABASE clover_db;

2. Enable Read Committed Snapshot Isolation on the new database

ALTER DATABASE clover_db SET READ_COMMITTED_SNAPSHOT ON;

3. Create a new login role.

```
CREATE LOGIN clover WITH PASSWORD = 'clover', DEFAULT_DATABASE = clover_db;
```

4. Connect to the database.

USE clover_db;

5. Create a new database user.

CREATE USER clover FOR LOGIN clover;

6. Add a database role membership db_owner (Members of the db_owner fixed database role can perform all configuration and maintenance activities on the database, and can also drop the database).

EXEC sp_addrolemember 'db_owner','clover';

CloverDX Server Setup

Using MS SQL requires configuration of the database server:

1. Run Microsoft SQL Server Management Studio tool;

- 2. Create a new user under Security/Logins;
- 3. Under **Databases** create a new database (e.g. 'clover') for **CloverDX Server**, select the user from the previous step as owner of the database;
- 4. Under database Options, set the Is Read Committed Snapshot On option to True.

Example of a properties file configuration:

```
jdbc.driverClassName=com.microsoft.sqlserver.jdbc.SQLServerDriver
jdbc.url=jdbc:sqlserver://localhost:1433;instance=SQLSERVERINSTANCE;database=clover_db
jdbc.username=clover
jdbc.password=clover
jdbc.dialect=org.hibernate.dialect.SQLServerDialect
```

Add a JDBC 4 compliant driver on the application server classpath. A JDBC driver that does not meet the JDBC 4 specifications will not work properly.



PostgreSQL

<u>Creating Database</u> (p. 73) <u>CloverDX Server Setup</u> (p. 73)

Creating Database

Advanced users can create their own table space

We are going to create a database for **CloverDX** to use a 'user group' role which will own the database and a user role which we will add to the user group. This user role will be then used by the Server to access the database.

Database name: clover_db

UserGroup: cloverdx

User: clover

Password: clover

- 1. Optionally, you can create a new tablespace
- 2. Connect as postgres (default admin) to the default DB postgres and execute the following commands:

```
CREATE ROLE cloverdx NOSUPERUSER NOCREATEDB NOCREATEROLE NOINHERIT NOLOGIN;
CREATE ROLE clover NOSUPERUSER NOCREATEDB NOCREATEROLE INHERIT LOGIN ENCRYPTED PASSWORD 'clover';
GRANT cloverdx TO clover;
CREATE DATABASE clover_db;
GRANT ALL ON DATABASE clover_db TO cloverdx;
REVOKE ALL ON DATABASE clover_db FROM public;
```

To separate the database into its own tablespace, create a tablespace before creating the database.

and use the following command to create the database:

CREATE DATABASE clover_db WITH OWNER cloverdx TABLESPACE tablespace_name;

For more information, see the **PostgreSQL** documentation.

CloverDX Server Setup

Example of a properties file configuration:

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://localhost/clover_db?charSet=UTF-8
jdbc.username=clover
jdbc.password=clover
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
```

Add a JDBC 4 compliant driver on the classpath. A JDBC driver which doesn't meet the JDBC 4 specifications won't work properly.

The JDBC driver for PostgreSQL can be downloaded from the official PostgreSQL page.

Example for Apache Tomcat: place the libraries into the TOMCAT/lib directory.

Chapter 14. System Database Configuration

See also PostgreSQL documentation on jdbc URL.



JNDI Configuration and Encryption

<u>JNDI DB Datasource</u> (p. 75) <u>JNDI Datasource Troubleshooting</u> (p. 76) <u>Encrypted JNDI</u> (p. 76)

JNDI DB Datasource

CloverDX Server can connect to a database using JNDI Datasource which is configured in an application server.



Worker JNDI Configuration

Note that the following configuration applies to <u>Server Core</u> (p. 4) only. Graphs running in <u>Worker</u> (p. 5) cannot use JNDI as defined in the application container of the Server Core, because Worker is a separate JVM process. Worker provides its own <u>JNDI configuration</u> (p. 92).

Example for Apache Tomcat and PostgreSQL database:

• JNDI Datasource Definition

First you need to **define a JNDI Datasource** in an application server. The following context resource configuration may be added to the [Tomcat_home]/conf/server.xml file to the <Host> element.

Note: Do not put the code into the <GlobalNamingResources> element, since the resource would not be visible by the **CloverDX** webapp.

```
<Context path="/clover">

<Resource name="jdbc/clover_server"

auth="Container"

type="javax.sql.DataSource"

factory="org.apache.tomcat.jdbc.pool.DataSourceFactory"

driverClassName="org.postgresql.Driver"

url="jdbc:postgresql://127.0.0.1:5432/clover_db"

username="clover"

password=""

maxTotal="20"

maxIdle="10"

maxWaitMillis="-1"/>

</Context>
```

• JNDI Connection Configuration

Now that the Datasource is defined, you should configure the connection.

The following parameters may be set in the same way as other parameters (in the properties file or the Tomcat context file). You can also set the parameters in the <u>Database (p. 57)</u> tab of the <u>Setup (p. 54)</u> GUI.

```
      datasource.type=JNDI
      # type of Datasource; must be set, because the default value

      datasource.jndiName=jdbc/clover_server
      # JNDI location of DB Datasource; the default value is java:c

      jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
      # Set the dialect according to DB which DataSource is connect

      # The correct dialect can be found in the examples of DB conf
```

Since the DB connection contains sensitive information (e.g. username, password, etc.), **CloverDX** provides the JNDI Encryption (p. 76) feature.



Tip

The resource configuration may also be added to the context file [Tomcat_home]/conf/Catalina/localhost/clover.xml.



Important

Special characters typed in the context file have to be specified as XML entities, e.g. ampersand "&" as "&", etc.

For a detailed list of parameters which can be set up in the configuration file, see Chapter 15, <u>List of Configuration</u> <u>Properties</u> (p. 82).

JNDI Datasource Troubleshooting

JNDI Datasource in Oracle WebLogic

CloverDX's default Quartz configuration does not work with default JNDI Datasource from WebLogic. Proceed with one of the following options:

- 1. Configure the Quartz JDBC delegate manually **before** the server is started;
- 2. Disable JDBC type wrapping in the WebLogic's Datasource configuration.

Apache Tomcat's DBCP JNDI pool

The default JNDI pool **DBCP** in **Apache Tomcat** does not handle connections efficiently. With the **DBCP** JNDI pool, low performance can be seen if DBOutputTable with returning statement is used.

Therefore,tomcat-jdbc-poolisusedinsteadbyaddingthefactory="org.apache.tomcat.jdbc.pool.DataSourceFactory"attribute to the definition ofthe JNDI resource. See The Tomcat JDBC Connection Pool

Encrypted JNDI

The encryption feature allows you to protect your sensitive data defined in the Datasource definition (e.g. username, password, etc.), which are by default stored in plain text. The configuration differs between particular application servers.

Encrypted JNDI on Tomcat (p. 77) Encrypted JNDI on JBoss 7 (p. 78) Encrypted JNDI on WebSphere 8.5.5.0 (p. 80) Encrypted JNDI on WebLogic (p. 81)

Encrypted JNDI on Tomcat

You need secure-cfg-tool to encrypt the passwords. Use the version of secure-cfg-tool corresponding to the version of **CloverDX Server**. Usage of the tool is described in Chapter 16, <u>Secure Configuration Properties</u> (p. 101).

Use encrypt.shorencrypt.bat for password encryption. Place the encrypted password into a configuration file, and put cloverdx-secure-jndi-resource-{version}.jar and jasypt-1.9.0.jar files on the classpath of the application server. The .jar files can be found in the tomcat-secure-jndi-resource directory packed in secure-cfg-tool.

The tomcat-secure-jndi-resource directory contains a useful README file with further details on encrypted JNDI.

Example of encrypted JNDI connection for PostgreSQL

Encrypt the password:

- 1. ./encrypt.sh -a PBEWithSHA1AndDESede
- 2. The configuration is placed in ${CATALINA_HOME}/conf/context.xml$. Note that the encryption algorithm PBEWithSHA1AndDESede is not default.

```
<Resource name="jdbc/clover_server"
auth="Container"
factory="com.cloveretl.secure.tomcatresource.Tomcat8SecureDataSourceFactory"
secureAlgorithm="PBEWithSHA1AndDESede"
type="javax.sql.DataSource"
driverClassName="org.postgresql.Driver"
url="jdbc:postgresql://127.0.0.1:5432/clover_db?charSet=UTF-8"
username="conf#rPz5Foo7HPn4dFTRV5Ourg=="
password="conf#4KlNp8/FVDR+rTWX0dEgWA=="
maxTotal="20"
maxIdle="10"
maxWaitMillis="-1"/>
```

If you use other JCE (e.g. Bouncy Castle), it has to be added to the classpath of the application server (\${CATALINA_HOME}/lib). The encrypt command requires the path to directory with JCE, too.

```
./encrypt.sh -l ~/lib/ -c
org.bouncycastle.jce.provider.BouncyCastleProvider -a
PBEWITHSHA256AND256BITAES-CBC-BC
```

```
<Resource name="jdbc/clover_server"
auth="Container"
factory="com.cloveretl.secure.tomcatresource.Tomcat8SecureDataSourceFactory"
secureProvider="org.bouncycastle.jce.provider.BouncyCastleProvider"
secureAlgorithm="PBEWITHSHA256AND256BITAES-CBC-BC"
type="javax.sql.DataSource"
driverClassName="org.postgresql.Driver"
url="jdbc:postgresql://127.0.0.1:5432/clover_db?charSet=UTF-8"
username="conf#Ws9IuHKo9h7hMjPllr31VxdI1A9LKIaYfGEUmLet9rA="
password="conf#Cj1v59Z5nCBHaktn6Ubgst4Iz69JLQ/q6/32Xwr/IEE="
maxTotal="20" maxIdle="10"
maxWaitMillis="-1"/>
```

Encrypted JNDI on JBoss 7

JBoss 7 - JBoss EAP 6.2.0.GA - AS 7.3.0.Final-redhat-14



Note

For details, see Using Encrypted DataSource Password in JBoss AS7.

Configuration steps are similar to configuring of JBoss 6.

The configuration takes place in a single configuration file, e.g. for standalone profile JBOSS_HOME/ standalone/configuration/standalone.xml.

Original data source:

```
<datasources>
    <datasource jndi-name="java:/MysqlDS" pool-name="MySQLPool">
        <connection-url>jdbc:mysql://localhost:3306/clover</connection-url>
        <driver>mysql</driver>
        <pool>
           <max-pool-size>30</max-pool-size>
        </pool>
        <security>
           <user-name>user</user-name>
           <password>password</password>
        </security>
   </datasource>
    <drivers>
        <driver name="mysql" module="com.cloveretl.jdbc">
           <driver-class>com.mysql.jdbc.Driver</driver-class>
        </driver>
    </drivers>
<datasources>
```

1. In JBOSS_HOME directory run the cli command:

java -cp modules/system/layers/base/org/picketbox/main/picketbox-4.0.19.SP2-redhat-1.jar:client/jboss-logging.jar

The command will return an encrypted password, e.g. 5dfc52b51bd35553df8592078de921bc.

2. Add a new security-domain to security-domains, the password value is a result of the command from the previous step.

3. Replace user and password with a reference to the security domain.

```
<datasources>

<datasource jndi-name="java:/MysqlDS" pool-name="MysqlPool" enabled="true" use-java-context="true">

<connection-url>jdbc:mysql://localhost:3306/clover</connection-url>

<driver>mysql</driver>

<pool>

<max-pool-size>30</max-pool-size>

</pool>

<security>
```

It is possible that the same mechanism can also be used for JMS.

Encrypted JNDI on WebSphere 8.5.5.0

In WebSphere, user credentials aren't saved in plain text, but as J2C authentication data. (see <u>How to Create a</u> <u>WAS JDBC Provider, J2C Authentication Alias, and Data Source for the IBM i</u>).

The same mechanism can also be used for JMS connection (see IBM's instructions on <u>Configuring an external</u> <u>JMS provider</u>).

Encrypted JNDI on WebLogic

Password in a JNDI datasource file is encrypted by default when created by admin's web console (Service/Datasource).

Example of datasource file (located in the DOMAIN/config/jdbc/ directory):



The same mechanism is also used for encrypting password in the JMS connection (see Oracle's instructions on <u>Configuring an external JMS provider</u>).



General Configuration Properties (p. 82) Worker - Configuration Properties (p. 90) Worker - JNDI Properties (p. 92) SSL properties for Worker (p. 95) Properties on Worker's command line (p. 95) Job Execution Properties (p. 98)

Below you can find the *configuration properties* available in **CloverDX Server**. The essential properties can be configured using the <u>Setup GUI (p. 54)</u>. Other properties serve to tweak various features of **CloverDX Server**. However, these properties have to be configured manually, e.g. by editing the configuration file.

In **CloverDX Server** UI, you can view the properties and their values in Configuration > CloverDX Info > Server Properties.

Additional properties used for cluster configuration can be found in Chapter 41, Sandboxes in Cluster (p. 277).



Important

Configuration property and system property are not the same. Configuration properties can be configured in **Setup** section or in cloverdx.properties file. System properties serve to configure the JVM. E.g. in Apache Tomcat, they are configured in bin/setenv.[bat|sh] file using -D prefix.

General Configuration Properties

Configuration file (p. 82) License (p. 83) Engine (p. 83) Sandboxes (p. 83) Database connection (p. 83) Security (p. 84) SMTP (p. 85) Logging (p. 85) Thread Manager (p. 86) Archivator (p. 87) Properties resolver (p. 87) Data Services (p. 87) API (p. 88) JVM (p. 88) Misc (p. 88)

Table	15.1.	General	confiau	ration

Кеу	Description	Default Value	
Configuration			
clover.config.file	Absolute path to location of a CloverDX Server configuration file	/absolute/ path/to/ cloverServ	er.propertie
clover.home	By default, this property is commented out and has a dynamically computed value: path containing CloverETL value for current (pre 5.0) installations and CloverDX for new (5.0 and newer) installations. If defined by the user, value has a higher priority. The property can be overridden using:	\${user.dat CloverETI or \${user.dat CloverDX	a.home}/ a.home}/

Key	Description	Default Value	
	environment variable clover.clover.home		
	context parameter <parameter name="clover.home"></parameter>		
	or system property -Dclover.clover.home=		
License			
license.file	Absolute path to location of a CloverDX Server license file (license.dat)		
license.context_name	sA comma-separated list of web-app contexts which may contain license. Each of them has to start with a slash! Works only on Apache Tomcat.	/clover- license,/ clover_licens	
Engine			
engine.config.file	location of a CloverDX engine configuration properties file	properties file packed with CloverDX	
engine.plugins.additio	This property may contain an absolute path to some "source" of additional ColorerDX engine plugins. These plugins are not a substitute for plugins packed in WAR. "Source" may be a directory or a zip file. Both, a directory and a zip, must contain a subdirectory for each plugin. Changes in the directory or the ZIP file apply only when the server is restarted. For details see Chapter 34, <u>Extensibility - CloverDX Engine Plugins</u> (p. 232).	empty	
Sandboxes			
sandboxes.home	This property is primarily intended to be used as a placeholder in the sandbox root path specification. So the sandbox path is specified with the placeholder and it's resolved to the real path just before it's used. The sandbox path may still be specified as an absolute path, but placeholder has some significant advantages:	\${clover.hom sandboxes	
	* sandbox definition may be exported/imported to another environment with a different directory structure		
	* user creating sandboxes doesn't have to care about physical location on the filesystem		
	* each node in cluster environment may have a different "sandboxes.home" value, so the directory structure doesn't have to be identical		
	For backward compatibility, the default value uses the content of the <u>clover.home</u> (p. 82) configuration property.		
sandboxes.access.che	ck.boundaries.enabled true false If it is set to false, then the path relative to a sandbox root may point out of the sandbox. No file/folder outside of the sandbox is accessible by the relative path otherwise.	true	
Database connection			
datasource.type	Set this explicitly to JNDI if you need CloverDX Server to connect to a DB using JNDI datasource. In such case, "datasource.jndiName" and "jdbc.dialect" parameters must be set properly. Possible values: JNDI JDBC	JDBC	

Key	Description	Default Value	
datasource.jndiName	JNDI location of a DB DataSource. It is applied only if "datasource.type" is set to "JNDI".	java:comp/ env/jdbc/ clover_serve	er
jdbc.driverClassNam	class name for JDBC driver name		
jdbc.url	JDBC URL used by CloverDX Server to store data		
jdbc.username	JDBC database user name		
jdbc.password	JDBC database password		
jdbc.dialect	hibernate dialect to use in ORM		
quartz.driverDelegate	SQL dialect for quartz. Value is automatically derived from "jdbc.dialect"		
Security		, 	
private.properties	List of server properties which are used only by the CloverDX Server code. So these properties are not accessible outside of the ServerFacade. By default, there are all properties which may contain password in the list, so their values are not visible for web GUI users. The values are replaced by a single star "*". Changes in this list may cause unexpected behavior of some server API.	jdbc.passwo executor.pas security.ldaj clover.smtp	ord, ssword, p.password .password
security.session.valid	Spession validity in milliseconds. When the request of logged-in user/client is detected, validity is automatically prolonged.	14400000	
security.session.exch	aligterival tfor exchange of invalid tokens in milliseconds.	360000	
security.default_dom	abomain in which all new users are included. Stored in user's record in the database. Shouldn't be changed unless the "clover" must be white-labelled.	clover	
security.basic_authen	tication.features_list List of features which are accessible using HTTP and which should be protected by Basic HTTP Authentication. The list has form of semicolon separated items; Each feature is specified by its servlet path.	/ request_pro simpleHttp/ launch;/ launchIt;/ downloadSt / downloadFi uploadSand downloadLo webdav	cessor;/ Api;/ orage; le;/ boxFile;/ og;/
security.basic_authen	tication.realm	CloverDX	
	Realm string for HTTP Basic Authentication.		
security.digest_authe	ntication.features_list List of features which are accessible using HTTP and which should be protected by HTTP Digest Authentication. The list has form of semi-colon separated items. Each feature is specified by its servlet path.		
	Please keep in mind that HTTP Digest Authentication is feature added to the version 3.1. If you upgraded your older CloverDX Server distribution, users created before the upgrade cannot use the HTTP Digest Authentication until they reset their passwords. So when they reset their passwords (or the admin does it for them), they can use Digest Authentication as well as new users.		
security.digest_auther	ntication.storeA1.enabled	false	

Key	Description	Default Value
	Switch whether the A1 Digest for HTTP Digest Authentication should be generated and stored or not. Since there is no CloverDX Server API using the HTTP Digest Authentication by default, it's recommended to keep it disabled. This option is not automatically enabled when any feature is specified in the security.digest_authentication.features_list property.	
security.digest_auther	ntication.realm Realm string for HTTP Digest Authentication. If it is changed, all users have to reset their passwords, otherwise they won't be able to access the server features protected by HTTP digest Authentication.	CloverDX Server
security.digest_auther	tication.nonce_validity Interval of validity for HTTP Digest Authentication specified in seconds. When the interval passes, server requires new authentication from the client. Most of the HTTP clients do it automatically.	300
security.lockout.login	Athermpts of failed login attempts after which a next failed login attempt will lock the user. Set the value to 0 to disable the function. Since 4.8.0M1.	50
security.lockout.reset	Rerived of time in seconds during which the failed login attempts are counted. Since 4.8.0M1.	60
security.lockout.unloc	Perirido f time in seconds after which a successful login attempt will unlock the previously locked user. Since 4.8.0M1.	300
security.csrf.protectio	Finallik /disable protection of Simple HTTP API against CSRF attacks, enabled by default. The CSRF protection requires presence of the X- Requested-By header in the requests.	true
	For more details, see the section called " <u>CSRF Protection</u> " (p. 235).	
SMTP		1
clover.smtp.transport.	pSM ddbkerver protocol. Possible values are "smtp" or "smtps".	smtp
clover.smtp.host	SMTP server hostname or IP address	
clover.smtp.port	SMTP server port	
clover.smtp.authentic	ation/false If it is false, username and password are ignored.	
clover.smtp.username	SMTP server username	
clover.smtp.password	SMTP server password	
clover.smtp.additiona	Properties with a "clover.smtp.additional." prefix are automatically added (without the prefix) to the Properties instance passed to the Mailer. May be useful for some protocol specific parameters. The prefix is removed.	
Logging		
logging.project_name	Used in log messages where it is necessary to name the product name.	CloverDX
logging.default_subdi	rName of a default subdirectory for all server logs; it is relative to the path specified by system property "java.io.tmpdir". Don't specify as an absolute path, use properties which are intended for absolute path.	cloverlogs
logging.logger.server	audit.enabled Enables logging of operations called on ServerFacade and JDBC proxy interfaces. The name of the output file is "server-audit.log". It is stored in the same directory as other CloverDX Server log files by default. The default logging level is DEBUG so it logs all operations which may process any change.	false

Key	Description	Default Value	
logging.logger.server	integration.enabled Enables logging of Designer-Server calls. The name of the output file is "server-integration.log". It is stored in the same directory as other CloverDX Server log files by default. The default logging level is INFO. Username is logged, if available. JDBC and CTL debugging is not logged.	true	
launch.log.dir	Location, where server should store launch requests logs. See Chapter 39, Launch Services (p. 250) for details.	\${java.io.tmpdi [logging. default_subdir]/ launch where \${java.io.tmpdi is system property	r}/ /
graph.logs_path	Location, where server should store Graph run logs. See Chapter 17, Logging (p. 104) for details.	\${java.io.tmpdi [logging. default_subdir]/ graph where \${java.io.tmpdi is system property	r}/ /
logging.appender.job	s.pattern_layout Pattern of the jobs' log messages	%d %-5p %-3X{runId} [%t] %m %n	
logging.appender.job	Encoding of the jobs' log files	UTF-8	
logging.mem_append	er.WORKER.pattern_layout		
	Format of log that can be seen in Monitoring > Logs > Worker .		
logging.mem_append	er.WORKER.size_limit		
	Size of log that can be seen in Monitoring > Logs > Worker .		
Thread Manager			
threadManager.pool.c	orePoolSize	4	
	Number of threads which are always active (running or idling). Related to a thread pool for processing server events.		
threadManager.pool.c	ueueCapacity	0	
	Max size of the queue (FIFO) which contains tasks waiting for an available thread. Related to a thread pool for processing server events. For queueCapacity=0, there are no waiting tasks, each task is immediately executed in an available thread or in a new thread.		
threadManager.pool.n	naxPoolSize	8192	
	Max number of active threads. If no thread from a core pool is available, the pool creates new threads up to "maxPoolSize" threads. If there are more concurrent tasks then maxPoolSize, thread manager refuses to execute it.		
threadManager.pool.a	llowCoreThreadTimeOut	false	

Key	Description	Default Value	
	Switch for idling threads timeout. If true, the "corePoolSize" is ignored so all idling threads may be time-outed		
threadManager.pool.l	ceepAliveSeconds	20	
	timeout for idling threads in seconds		
Archivator			
task.archivator.batch_	Max number of records deleted in one batch. It is used for deleting of archived run records.	50	
task.archivator.archiv	e_file_prefix Prefix of archive files created by the archivator.	cloverArc	nive_
Properties resolver]
properties_resolver.re	solve_server_props.server_props_list_additional A list of properties from a subset of properties, which values are resolved. The properties' values may use system properties or environment variables as placeholders. The values are resolved during the server startup. If the system property is changed later, the resolved CloverDX Server property value doesn't change. Users may use this property, if some property they need to resolve is missing in the property: properties_resolver.resolve_server_props_server_props_list_default. If the property to resolve is already specified by the property properties_resolver.resolve_server_props_server_props_list_default, don't add it to this property.		
properties_resolver.re	solve_server_props.server_props_list_default A list of properties from a subset of properties, which values are resolved. The properties' values may use system properties or environment variables as placeholders. Values are resolved during the server startup. If the system property is changed later, the resolved CloverDX Server property value doesn't change. Users are discouraged from modification of the property, unless it's necessary. Instead, users may add more properties by modifying property: properties_resolver.resolve_server_props.server_props_list_additional	clover.hor sandboxes sandboxes cluster.jgr cluster.jgr cluster.jgr cluster.jgr cluster.jgr cluster.jgr cluster.jgr	he, home, home.local, home.partitioned oups.bind_address oups.start_port, oups.external_add oups.external_por oups.tcpping.initis up.name, p.url
properties_resolver.p	aceholders.server_props_list_default A list of properties from a subset of properties, that may be used as placeholders and shall be resolved if used in paths. The properties can be used if you define a path to the root of a sandbox, or to locations of local or partitioned sandboxes, or path to a script, or path in archiver job. Users are strongly discouraged from modification of the property. The property name changed since CloverDX 4.2 , however the obsolete name is also still accepted to maintain backwards compatibility.	clover.hor sandboxes sandboxes sandboxes user.data.h	ne, .home, .home.local, .home.partitioned tome
Data Services			
dataservice.invocatio	It sets the maximal age in minutes before the record is removed from the database. The default is 1440 min = 24 h.	1440	
dataservice.failure.rat	io.min.record.count Used for Data Service failure indication. It represents the minimum number of invocations required to evaluate whether the percentage of failures is over the threshold. Ensures that during periods of low traffic the endpoint does not switch to failing state. 10 by default.	10	

Key	Description	Default Value
API	·	,
launch.http_header_p	refietix of HTTP headers added by launch services to the HTTP response.	X- cloveretl
http.api.enabled	Enables or disables simple HTTP API.	true
	If the HTTP API is disabled, there is no link to HTTP API operations in login page, the HTTP API tab in Launch Service that is accessible under Test button is not visible, and the HTTP API, the /clover/ httpapi.jsp and HTTP API servlet are not accessible.	
webDay method pror	find maxDenth	40
webDav.memod.prop	Maximum depth for webDAV method PROPFIND. When the depth is not specified, the default is supposed to be infinite (according to the rfc2518), however it's necessary to set some limit, otherwise the webDav client might overload the server filesystem. Also if the depth value specified by webDAV client in the request is higher than the pre-configured max depth, only the pre-configured maximum is	40
	used.	
JVM		
server.env.min_heap_	Setsoby required minimal heap memory threshold. If the configuration of CloverDX Server is set to less heap memory, a warning is displayed. Experienced users can change the default value to avoid the warning when running the server on a system with lower memory. The threshold is in megabytes.	900
server.env.min_nonh	eap_memory Sets the required minimal non-heap memory threshold. If the configuration of CloverDX Server is set to less non-heap memory, a warning is displayed. Experienced users can change the default value to avoid the warning when running the server on a system with lower memory. The threshold is in megabytes.	256
jvm.implementation.c	heck.enabled Displays warnings when unsupported Java implementation is used.	true
Miscellaneous		
temp.default_subdir	Name of a default subdirectory for server tmp files; it is relative to the path specified by system property "java.io.tmpdir".	clovertmp
graph.pass_event_par	ams_to_graph_in_old_style Since 3.0. It is a switch for backwards compatibility of passing parameters to the graph executed by a graph event. In versions prior to 3.0, all parameters are passed to executed graph. Since 3.0, just specified parameters are passed. Please see <u>Start a Graph</u> (p. 179) for details.	false
cluster.node.invocatio	n.record.info.interval Sets the interval for synchronization of the Data Services health state between the cluster nodes. The time is in milliseconds.	30000
clover.event.fileChec	MinInterval Interval of the timer, running file event listener checks (in milliseconds). See <u>File Event Listeners (remote and local)</u> (p. 221) for details.	1000
clover.event.groovyC	heckMinInterval	1000

Key	Description	Default Value
clover.inDevelopmen		1000
launchservices.visible	Display Launch Services (p. 250) in the main menu in Server UI. Can be true or false.	false

Worker - Configuration Properties

Table 15.2. Server - Worker configuration

Key	Description	Default Value
worker.initialWorker	Enable/disable the Worker. To enable Worker, set to 1 (this is the default). To disable Worker and run all jobs in Core Server, set to 0.	1
	Starting more than one Worker is currently (in 4.9.0) not supported.	
worker.portRange	Port range used for communication between Server Core and Worker and between Workers on different cluster nodes. Communication between Server Core and Worker is done on localhost. Workers on different cluster nodes communicate directly with each other over these ports - in Cluster setup, this port range should be open in firewall for other Cluster nodes. This property can be easily configured in the Worker (p. 58) tab	10500-10600
	of Setup (p. 54). worker.portRange should contain at least 5 ports for 1 node (depending on other options, a node takes at most 5 ports from the range). We recommend to use portRange of at least 10 ports to avoid possible problems with occupied ports after restart of Worker.	
	If more cluster nodes run on the same machine, make sure that there are enough free ports for Workers of all cluster nodes on the machine. The default configuration of worker.portRange is sufficient for that.	
worker.connectTimed	ultimeout for connection initialization between Worker and Server Core, in both directions. The timeout is in milliseconds.	5000
	This setting can be useful in case of handling communication issues between Server Core and Worker, typically under high load you might want to increase the timeout.	
worker.readTimeout	Read timeout for communication requests between Worker and Server Core, in both directions. If a request is not completely served before reaching this limit, the connection is terminated. The timeout is in milliseconds.	600000
	This setting can be useful in case of handling communication issues between Server Core and Worker, typically under high load you might want to increase the timeout.	
worker.classpath	A directory with additional .jar files to be added to the Worker's classpath. The .jar files would typically be libraries used by graphs (e.g. JDBC drivers for database connections) or JDBC drivers used in JNDI connections defined in Worker (see <u>Worker - JNDI Properties</u> (p. 92)).	\${clover.home}/ worker-lib
	The Worker's classpath is separate from Server Core (i.e. application container classpath). Any libraries needed by jobs executed on Worker need to be added on the Worker's classpath.	

Key	Description	Default Value
	For backward compatibility, the default value uses the content of the <u>clover.home</u> (p. 82) configuration property.	
	The property can contain paths to multiple directories. The separator between the directories can be a colon (on Linux and Mac) or semicolon (Linux, Mac and Windows), e.g.:	
	worker.classpath=/home/clover/worker-lib;/ opt/worker-lib-2	
	If a directory is added on the Worker's classpath, its subdirectories are automatically added too.	
	Some basic wildcards are supported: directory-* and directory-?.	
worker.maxHeapSize	The maximum Java heap size of Worker in MB, it will be translated to the -Xmx option for the Worker's JVM. Jobs executed in the Worker require heap memory based on their complexity, dataset size, etc.	0
	See our recommendations (p. 37) for heap sizes of Worker and Server Core.	
	This property can be easily configured in the Worker (p. 58) tab of Setup (p. 54).	
	Setting to 0 uses Java default heap size (automatically determined by Java). This setting is not recommended for production usage.	
worker.initHeapSize	The initial Java heap size of Worker in MB, it will be translated to the -Xms option for the Worker's JVM. We recommend to set this to the same value as worker.maxHeapSize	0
	This property can be easily configured in the Worker (p. 58) tab of Setup (p. 54).	
	Setting to 0 uses Java default initial heap size (automatically determined by Java). This setting is not recommended for production usage.	
worker.jvmOptions	Adds Java command line options for the Worker's JVM. This property is useful to tweak the configuration of the Worker's JVM, e.g. to tune garbage collector settings. These command line options override default options of the JVM.	
	For example to enable parallel garbage collector: -XX: +UseParallelGC.	
	See Additional Diagnostic Tools (p. 164) section for useful options for troubleshooting and debugging Worker.	
	This property can be easily configured in the Worker (p. 58) tab of Setup (p. 54).	
worker.enableDebug	Remote Java debugging of Worker, enables <u>JDWP</u> . Enabling this allows you to connect a Java debugger remotely to the running Worker process, to debug your Java transformations, investigate issues, etc. The port used by the debugger is determined	false

Key	Description	Default Value
	dynamically and can be seen in the Worker section of the Monitoring (p. 108) or Setup (p. 54) page.	
worker.inheritSystem	Properties	true
	Sets whether system Java properties are inherited from the Server Core process to the Worker process. We automatically inherit some system properties to simplify the Worker configuration. For the list of system Java properties inherited from the Server Core to Worker, see <u>properties passed from Server Core if</u> worker.inheritSystemProperties is true (p. 96). This functionality is enabled by default. Use this property to disable this behavior in case some of the inherited properties would cause issues.	
worker.javaExecutabl	eAbsolute path to the Java binary for Worker process, e.g. /	Value is
	Use this property if you need to use a specific Java binary for running the Worker.	determined based on \$JAVA_HOME environment variable.

Worker - JNDI Properties

The Worker has its own JNDI pool separate from the application container JNDI pool. If your jobs use JNDI resources (to obtain JDBC or JMS connections), you have to configure the Worker's JNDI pool and its resources.

The worker JNDI properties must be configured using the clover.properties configuration file. Libraries used by the JNDI resources must be added to the Worker's classpath, see worker.classpath (p. 90).

It is possible to define multiple datasources pointing to different databases or JMS queues, see examples below. The datasources are indexed in configuration, their properties have suffix [0], [1], etc. Even a single datasource must have the [0] index.

JDBC Datasources

Worker uses the Apache DBCP2 pool for its JNDI functionality. Any DBCP2 configuration attribute is supported, see <u>DBCP attributes</u>. The only mandatory properties are jndiName and url.

See table below for basic JNDI properties.

You can monitor the state of the datasources via JMX. See Additional Diagnostic Tools (p. 164) for details on how to enable JMX on Worker. Then you can connect to the Worker's JMX interface with tools like jconsole and monitor the JNDI datasources, e.g. for the number of currently open connections. The related MBeans are under the Tomcat/DataSource/localhost///javax.sql.DataSource path:

Overview Memory Threads Classes VM Summa	ry N	(Beans		4
👜	^	Attribute values		
Connector Connector Connector Coalhost Coalhost		Name maxIde maxOpenPrepared maxTotal maxWaitMills minEvictableIdleTim minIde modelerType	Value 8 8 -1	
⊕-⊕ Engine ⊕- Filter ⊕- GlobalRequestProcessor ⊕- Host Loader		numActive numIdle numTestsPerEvictio password	0 0 inRun 3 root	

Figure 15.1. MBean for a JNDI datasource in jconsole

Key	Description	Example
worker.jndi.datasource[0].j	ndiName The name of the JNDI datasource. Mandatory.	jdbc/database_name
worker.jndi.datasource[0].u	rl The JDBC connection URL. Mandatory.	jdbc:postgresql:// hostname:5432/ database_name
worker.jndi.datasource[0].u	sername The user name for a database connection.	clover
worker.jndi.datasource[0].p	assword The password for a database connection. The password value can be encrypted using the secure configuration tool, see Secure Configuration Properties (p. 101).	clover
worker.jndi.datasource[0].c	riverClassName The database driver classname. The database driver must be on the Worker classpath, see worker.classpath (p. 90).	org.postgresql.Driver
worker.jndi.datasource[0].r	naxIdle The maximum number of idle database connections in a pool. Set to -1 for no limit.	10
worker.jndi.datasource[0].n	naxTotal The maximum number of database connections in a pool. Set to -1 for no limit.	20
worker.jndi.datasource[0].r	naxWaitMillis The maximum time Worker waits for a database connection to become available. In milliseconds, set to -1 for no limit.	30000
worker.jndi.datasource[0].c	bcpAttribute Any DBCP2 attribute, e.g. worker.jndi.datasource[0].initSQL. See DBCP attributes.	

Table 15.3	Properties for	JDBC JNDI	Resources in	Worker
------------	----------------	-----------	--------------	--------

The following example shows configuration of two JDBC Datasources.

```
worker.jndi.datasource[0].jndiName=jdbc/postgresql_finance
worker.jndi.datasource[0].url=jdbc:postgresql://finance.example.com:5432/finance
worker.jndi.datasource[0].maxIdle=5
worker.jndi.datasource[0].maxTotal=10
worker.jndi.datasource[0].maxWaitMillis=-1
worker.jndi.datasource[0].username=finance_user
worker.jndi.datasource[0].password=conf#eCflGDlDtKSJjh9VyDlRh7IftAbI/vsH
worker.jndi.datasource[0].driverClassName=org.postgresql.Driver
worker.jndi.datasource[1].jndiName=jdbc/MysqlDB
worker.jndi.datasource[1].url=jdbc:mysql://marketing.example.com:3306/marketing?useUnicode=true&characterEncod
worker.jndi.datasource[1].maxIdle=10
worker.jndi.datasource[1].maxTotal=20
worker.jndi.datasource[1].maxWaitMillis=-1
worker.jndi.datasource[1].username=marketing_user
worker.jndi.datasource[1].password=conf#JWsMa2okg7Dq2gtLBM84sE==
worker.jndi.datasource[1].driverClassName=com.mysql.jdbc.Driver
```

JMS Connections

Worker can use any JMS broker to define JMS connections in JNDI. Any JMS broker configuration attribute is supported. The mandatory properties are jndiName,factoryClass,typeClass and typeInterface. See table below for basic JNDI properties for JMS resources.

Table	15.4.	Propertie	es for JM	IS JNDI F	Resources	in Worker
i ubio	10.1.	i roportio		0 01 101 1	100000100001	

Key	Description	Example	
worker.jndi.jms[0].jndiNar	ne The name of the JNDI JMS resource. Mandatory.	jms/jms_queue	
worker.jndi.jms[0].factory	Factory class for creating the JMS resource. This is JMS broker specific. Mandatory.	org.apache.act:	.vemq.jndi.J
worker.jndi.jms[0].type	Implementation class of the JMS resource. This is JMS broker specific. Mandatory.	org.apache.act:	vemq.comman
worker.jndi.jms[0].jmsProp	erty Configuration property for the JMS resource. Any configuration property supported by the JMS broker can be used.	worker.jndi.jms	[0].brokerU

The following example shows configuration of several JMS resources.

worker.jndi.jms[0].jndiName=jms/CloverConnectionFactory
worker.jndi.jms[0].type=org.apache.activemq.ActiveMQConnectionFactory
worker.jndi.jms[0].factory=org.apache.activemq.jndi.JNDIReferenceFactory
worker.jndi.jms[0].brokerUrl=tcp://localhost:61616?jms.prefetchPolicy.queuePrefetch=1
worker.jndi.jms[0].brokerName=LocalActiveMQBroker
worker.jndi.jms[1].jndiName=jms/CloverQueue
worker.jndi.jms[1].type=org.apache.activemq.command.ActiveMQQueue
worker.jndi.jms[1].factory=org.apache.activemq.jndi.JNDIReferenceFactory
worker.jndi.jms[1].factory=org.apache.activemq.jndi.JNDIReferenceFactory
worker.jndi.jms[1].epysicalName=TestQueue

Worker - SSL Properties

In Cluster, Workers of each node communicate with each other directly for increased performance. This communication is used to transport data of cluster remote edges in clustered jobs between the nodes. For increased security, it is possible to use SSL for the remote edge communication.

SSL communication between Workers needs to be enabled and configured separately from SSL of the application container that runs Server Core. The worker.ssl.enabled property is used to enable/disable SSL. If a Cluster node's "self" URL is using HTTPS, we automatically set the property to true. Configuration of SSL consists of setting paths and passwords of KeyStore and TrustStore, see the table below for details.

Note that if the standard SSL related system properties (javax.net.ssl.keyStore, javax.net.ssl.keyStorePassword, javax.net.ssl.keyAlias, javax.net.ssl.trustStore and javax.net.ssl.trustStorePassword) are used to configure KeyStore/TrustStore for the Server Core, they are propagated to Worker; therefore, their respective worker.ssl properties do not need to be configured.

Recommended steps to enable SSL for inter-worker communication are:

- Enable SSL for each cluster node, via the application container settings. Configure TrustStore and KeyStore via the standard javax.net.ssl.* properties.
- Set cluster.http.url for each node to point to its own HTTPS URL
- Check that communication between Cluster nodes over SSL works and that the nodes can correctly see each other. The Monitoring page of Server Console should show the whole cluster group and its nodes correctly.
- Worker should automatically inherit the above SSL configuration.
- Run a clustered job on Worker

Key	Description	Example				
worker.ssl.enabled	worker.ssl.enabled Enables or disables an SSL connection for Worker. Note that if the Server runs on HTTPS, SSL is enabled automatically; however, this property has a higher priority.					
worker.ssl.keyStore	Absolute path to the KeyStore file.	path/to/keyStore.file				
worker.ssl.keyStorePasswo	rd The KeyStore password.					
worker.ssl.keyAlias	The alias of the key in keyStore. Optional - the property does not have to be specified if there is only one key in the KeyStore.					
worker.ssl.port	The port for SSL communication with Worker. The property is configured automatically and the value is set from <u>worker.portRange</u> (p. 90).					
cluster.ssl.disableCertificat	e Midatherionalidation of certificates in HTTPS connections of remote edges. Disabling the validation affects jobs run on both Worker and Server Core.	true/false				

Properties on Worker's Command Line

To ensure proper functionality of CloverDX Worker, there is a number of parameters which can appear on its command line. These can be hard-coded, passed from the configuration file, propagated from Server Core or added manually for a specific purpose, see the list below.

Note that the list does not include internal system parameters not configurable by the user.

Table 15	5.6. List of	Properties on	Worker's	Command	Line
----------	--------------	---------------	----------	---------	------

Property	Note
Passed fro	om Server Core if <u>worker.inheritSystemProperties</u> (p. 92) is true
-Djavax.net.ssl. trustStorePassword	Standard SSL related properties.
-Djavax.net.ssl. keyStorePassword	Values of the properties containing passwords are encrypted.
-Djavax.net.ssl.keyStore	javax.net.ssl.keyStore is passed only if
-Djavax.net.ssl.trustStore	
-Djavax.net.ssl.keyAlias	See also <u>SSL properties for Worker</u> (p. 95).
-Djava.library.path	
-Djava.io.tmpdir	
-Dhttps.protocols	Standard Java properties
-XX:MaxMetaspaceSize	
- Djava.rmi.server.hostname	
-DsocksProxyHost	
-DsocksProxyPort	
-DsocksProxyVersion	
- Djava.net.socks.username	
- Djava.net.socks.password	Properties for proxy configuration.
- Djava.net.useSystemProxie	The properties with $*$ are passed for http, https and ftp.
-D*.proxyHost	
-D*.proxyPort	
-D*.proxyUser	
-D*.proxyPassword	
-D*.nonProxyHosts	
- Dcom.opensys.cloveretl.ad	dependent of the component. For more information, see the Designer documentation of the component.
Dcom.opensys.cloveretl.ad	dressdoctor.persistent
- Dcloveretl.smb2.bouncyca	Bouncy Castle for SMB2. stle.jar.file
	Hard-coded Properties
- Djdk.nio.maxCachedBuffe	Limits the memory used by the temporary buffer cache; prevents memory leaks. Size
-Dfile.encoding	The charset for file contents.
	Added for Java 8
-XX:+UseG1GC	Setting G1 garbage collector as default for Java 8. See <u>garbage collector for</u> <u>Worker</u> (p. 40).
	Added for Java 9 and Newer

Property	Note				
add-opens=java.base/ java.lang					
add-opens=java.base/ java.util					
add-exports=java.xml/ com.sun.org.apache.xerces	internal.parsers				
add-opens=java.rmi/ sun.rmi.transport					
Add	ed from config.properties/clover.properties				
-Xms*m	From the worker.initHeapSize property; * is the property value.				
-Xmx*m	From the worker.maxHeapSize property; * is the property value.				
- agentlib:jdwp=transport=dt	From the worker.enableDebug (p. 91) property; * is a dynamically determined port useek bysthe clebuggdre The point pandon seen in the Worker section of the Monitoring or Setup page.				
-Dworker.ssl.enabled					
-Dworker.ssl.keyStore	Worker encific SSL configuration properties				
-Dworker.ssl. keyStorePassword	Values of the properties containing passwords are encrypted.				
-Dworker.ssl.keyAlias	If the standard SSI properties are used (see shove) the Worker-specific properties				
-Dworker.ssl.trustStore	don't have to be configured. See also <u>SSL properties for Worker</u> (p. 95).				
- Dworker.ssl.trustStorePass	word				
- Dsecurity.config_propertie	Eneryptomprovider lans Nargorithm for secure parameters. See Secure parameters configuration (p. 122).				
Dsecurity.config_propertie	s.encryptor.algorithm				
	Added if <u>worker.ssl.enabled</u> (p. 95) is true				
-Dworker.ssl.port	The port for SSL communication with Worker is taken from the port range set by the <u>worker.portRange</u> (p. 90) property.				

Job Execution Properties

Table	15.7.	Defaults	for job	execution	configuration	ı - see	<u>Job</u>	Config	Properties	(р.	151)	for	details
-------	-------	----------	---------	-----------	---------------	---------	------------	--------	-------------------	-----	------	-----	---------

Key	Description	Default Value
executor.tracking_interval	An interval in milliseconds for scanning of a current status of a running graph. The shorter interval, the bigger log file.	2000
executor.log_level	Log level of graph runs. TRACE DEBUG INFO WARN ERROR	INFO
executor.max_job_tree_dep	th	32
	Defines maximal depth of the job execution tree, e.g. for recursive job it defines the maximal level of recursion (counting from root job).	
executor.max_running_con	currently Amount of graph instances which may exist (or run) concurrently. 0 means no limits.	0
executor.max_graph_instar	ice_age	0
	Specifies how long can a graph instance be idling before it is released from memory. Interval is in milliseconds. 0 means no caching.	
	This property has been renamed since 2.8. Original name was executor.maxGraphInstanceAge	
executor.classpath	Classpath for transformation/processor classes used in the graph. Directory [Sandbox_root]/trans/ does not have to be listed here, since it is automatically added to a graph run classpath.	
executor.skip_check_confi	Disables check of graph configuration. Increases performance of a graph execution; however, it may be useful during graph development.	true
executor.password	This property is deprecated. The password for decoding encoded DB connection passwords.	
executor.verbose_mode	If true, more descriptive logs of graph runs are generated.	true
executor.use_jmx	If true, the graph executor registers JMX mBean of the running graph.	true
executor.debug_mode	If true, edges with enabled debug store data into files in debug directory.	false

List of all properties

clover.event.fileCheckMinInterval (p. 88) clover.event.fileCheckMinInterval (p. 88) clover.inDevelopment (p. 89) clover.smtp.additional.* (p. 85) clover.smtp.authentication (p. 85) clover.smtp.host (p. 85) clover.smtp.password (p. 85) clover.smtp.port (p. 85) clover.smtp.transport.protocol (p. 85) clover.smtp.username (p. 85) clover.smtp.username (p. 85) config.file (p. 82) clover.home (p. 82) dataservice.invocation.record.max.age (p. 87) dataservice.failure.ratio.min.record.count (p. 87) datasource.type (p. 83) datasource.jndiName (p. 84) engine.config.file (p. 83) engine.plugins.additional.src (p. 83) executor.classpath (p. 98) executor.debug_mode (p. 98) executor.log_level (p. 98) executor.max_job_tree_depth (p. 98) executor.max running concurrently (p. 98) executor.max_graph_instance_age (p. 98) executor.password (p. 98) executor.skip_check_config (p. 98) executor.tracking.interval (p. 98) executor.use_jmx (p. 98) executor.verbose mode (p. 98) graph.logs path (p. 86) graph.pass event params to graph in old style (p. 88) http.api.enabled (p. 88) jdbc.dialect (p. 84) jdbc.driverClassName (p. 84) jdbc.password (p. 84) jdbc.url (p. 84) jdbc.username (p. 84) jvm.implementation.check.enabled (p. 88) launch.http header prefix (p. 88) launch.log.dir (p. 86) launchservices.visible (p. 89) license.context_names (p. 83) license.file (p. 83) logging.appender.jobs.encoding (p. 86) logging-appender-jobs-pattern layout (p. 86) logging.default_subdir (p. 85) logging.logger.server audit.enabled (p. 85) logging.logger.server integration.enabled (p. 86) logging.mem appender.WORKER.pattern layout (p. 86) logging.mem_appender.WORKER.size_limit (p. 86) logging.project_name (p. 85) private.properties (p. 84) properties resolver.placeholders.server props list default (p. 87) properties resolver.resolve_server_props_list_additional (p. 87) properties_resolver.resolve_server_props_list_default (p. 87) quartz.driverDelegateClass (p. 84) sandboxes.access.check.boundaries.enabled (p. 83) sandboxes.home (p. 83) security.basic authentication.features list (p. 84) security.basic authentication.realm (p. 84) security.default_domain (p. 84) security.digest_authentication.features_list (p. 84) security.digest authentication.nonce_validity (p. 85) security.digest_authentication.realm (p. 85) security.digest_authentication.storeA1.enabled (p. 84) security.lockout.login.attempts (p. 85) security.lockout.reset.period (p. 85) security.lockout.unlock.period (p. 85)

security.csrf.protection.enabled (p. 85) security.session.exchange.limit (p. 84) security.session.validity (p. 84) server.env.min_heap_memory (p. 88) server.env.min_nonheap_memory (p. 88) task.archivator.archive file prefix (p. 87) task.archivator.batch size (p. 87) temp.default subdir (p. 88) threadManager.pool.allowCoreThreadTimeOut (p. 86) threadManager.pool.corePoolSize (p. 86) threadManager.pool.keepAliveSeconds (p. 87) threadManager.pool.maxPoolSize (p. 86) threadManager.pool.queueCapacity (p. 86) webDay.method.propfind.maxDepth (p. 88) worker.additional.classpath (p. 90) worker.enableDebug (p. 91) worker.inheritSystemProperties (p. 92) worker.initHeapSize (p. 91) worker.javaExecutable (p. 92) worker.jvmOptions (p. 91) worker.maxHeapSize (p. 91) worker.portRange (p. 90) worker.connectTimeout (p. 90) worker.readTimeout (p. 90) worker.initialWorkers (p. 90) worker.jndi.datasource[0].url (p. 93) worker.jndi.datasource[0].maxIdle (p. 93) worker.jndi.datasource[0].maxTotal (p. 93) worker.jndi.datasource[0].maxWaitMillis (p. 93) worker.jndi.datasource[0].username (p. 93) worker.jndi.datasource[0].password (p. 93) worker.jndi.datasource[0].driverClassName (p. 93) worker.jndi.datasource[0].jndiName (p. 93) worker.jndi.jms[0].jndiName (p. 94) worker.jndi.jms[0].factory (p. 94) worker.jndi.jms[0].type (p. 94) worker.jndi.jms[0].jmsProperty (p. 94) worker.ssl.enabled (p. 95) worker.ssl.keyStore (p. 95) worker.ssl.keyStorePassword (p. 95) worker.ssl.keyAlias (p. 95) cluster.ssl.disableCertificateValidation (p. 95)

Chapter 16. Secure Configuration Properties

Some configuration properties can be confidential (e.g. a password to a database, mail client, etc.) and thus it's desirable to encrypt them. For this purpose, there is a command-line utility *secure-cfg-tool.jar*.

Basic Utility Usage (p. 101) Advanced Usage - Custom Settings (p. 102)

Basic Utility Usage

1. Download the utility archive file (secure-cfg-tool.zip) and unzip it.

The utility is available in the download section of your **CloverDX** account - at the same location as the download of **CloverDX Server**.

2. Execute the script given for your operating system, encrypt.bat for MS Windows, encrypt.sh for Linux. You will be asked for inserting a value of a configuration property intended to be encrypted.

Example:

C:\secure-cfg-tool>encrypt.bat

Enter text to encrypt: mypassword Text to encrypt: "mypassword" Encrypted text: conf#eCflGDlDtKSJjh9VyDlRh7IftAbI/vsH

C:\secure-cfg-tool>

If you want to configure the way the values are encrypted, see Advanced Usage - Custom Settings (p. 102)

3. The encrypted string has *conf#encrypted_property* format and can be used as a value of a configuration property in the properties file, clover.xml file or web.xml file (see details about configuration sources in Chapter 12, <u>Configuration Sources</u> (p. 50)).

Example of a configuration property file with encrypted password:

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://127.0.0.1/clover_db?charSet=UTF-8
jdbc.username=yourUsername
jdbc.password=conf#eCflGDlDtKSJjh9VyDlRh7IftAbI/vsH
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
```

Alternatively, you can use the following command:

java -jar secure-cfg-tool.jar


Important

Values encrypted by a Secure parameter form (Chapter 20, <u>Secure Parameters</u>(p. 120)) **cannot** be used as a value of a configuration property.

Advanced Usage - Custom Settings

The way of encrypting configuration values described above uses default configuration settings (a default provider and algorithm). If you need to customize the settings, use the following parameters of the *secure-cfg-tool.jar* utility.

Table 16.1. Parameters

Parameter	Description	Example
algorithm, -a	algorithm to encrypt	algorithm PBEWithMD5AndDES
file, -f	config file location	-f C:\User\John \cloverServer.properties
help, -h	show help	help
providerclass, -c	custom provider class	-c org.provider.ProviderClass
 providerlocation, -l	path to jar/folder containing a custom provider class (it will be added to the classpath)	providerlocation C:\User\John\lib \customprovider.jar, -l C:\User\John \lib\
providers, -p	print available security providers and their algorithms	providers



Note

To demonstrate usage of an external provider the Bouncy Castle provider is used.

To find out a list of algorithms, use -p or --providers

```
C:\secure-cfg-tool>encrypt.bat -p
```

If you want to find out a list of algorithms of an external provider, you must pass the provider's class name and path to jar file(s).

```
C:\secure-cfg-tool>encrypt.bat -p -c
org.bouncycastle.jce.provider.BouncyCastleProvider -l C:\User\John\bcprov-
jdk15on-152.jar
```

Result might look like this:

```
***** List of available providers and their algorithms *****
Provider: SunJCE
Provider class: com.sun.crypto.provider.SunJCE
Algorithms:
PBEWithMD5AndDES
PBEWithSHA1AndDESede
PBEWithSHA1AndRC2_40
Provider: BC
Provider class: org.bouncycastle.jce.provider.BouncyCastleProvider
Algorithms:
PBEWITHMD2ANDDES
PBEWITHMD5AND128BITAES-CBC-OPENSSL
PBEWITHMD5AND192BITAES-CBC-OPENSSL
PBEWITHMD5AND256BITAES-CBC-OPENSSL
```

The provider class is displayed on the row starting with *Provider class*, algorithms are strings with *PBE* prefix. Both can be used to configure encryption.

Configuring the Encryption Process

The algorithm and provider can be passed to the utility in two ways.

• Using command line arguments

To change the algorithm, use the argument -a. The provider remains default (SunJCE in case of Oracle Java):

C:\secure-cfg-tool>encrypt.bat -a PBEWithMD5AndDES

To use an external provider, you must specify the provider's class name (the --providerclass or -c arguments) and add jar(s) to the classpath (the --providerlocation or -l arguments). Provider location must point to a concrete jar file or directory containing the jar(s) and can be used several times for several paths:

```
C:\secure-cfg-tool>encrypt.bat -a PBEWITHSHA256AND256BITAES-CBC-BC -c org.bouncycastle.jce.provider.BouncyCastleProvider -l C:\User\John\bcprov-jdk15on-152.jar
```

• Using configuration file

A configuration file is a common properties file (text file with key-value pairs):

[property-key]=[property-value]

See the following example of secure.config.example.properties distributed within securecfg-tool.zip):

security.config_properties.encryptor.providerClassName=org.bouncycastle.jce.provider.BouncyCastleProvider
security.config_properties.encryptor.algorithm=PBEWITHSHA256AND256BITAES-CBC-BC
security.config_properties.encryptor.provider.location=C:\\User\\libs

You must also set the path to the file using the **-f** argument:

C:\secure-cfg-tool>encrypt.bat -f path/to/secure.config.example.properties



Note

More jar locations can be set in the **security.config_properties.encryptor.providerLocation** property. The locations are delimited by semicolon.

Configuring an application server

CloverDX Server application needs to know how the values have been encrypted, therefore the properties must be passed to the server (see details in Part III, "<u>Configuration</u>" (p. 47)). For example:

```
security.config_properties.encryptor.providerClassName=org.bouncycastle.jce.provider.BouncyCastleProvider
security.config_properties.encryptor.algorithm=PBEWITHSHA256AND256BITAES-CBC-BC
...
```



Important

If a third-party provider is used, its classes must be accessible to the application server. Property **security.config_properties.encryptor.providerLocation** will be ignored.

Chapter 17. Logging

Main Logs (p. 104) Another Useful Logging Settings (p. 104) Access Log in Apache Tomcat (p. 105) Application Server Logs (p. 105) Graph Run Logs (p. 105) Server Audit Logs (p. 105) Designer-Server Integration Logs (p. 106)

Main Logs

The **CloverDX Server** uses the log4j library for logging. The WAR file contains the default log4j configuration. The log4j configuration file log4j.xml is placed in WEB-INF/classes directory.

By default, log files are produced in the directory specified by the java.io.tmpdir system property in the cloverlogs subdirectory.

The java.io.tmpdir system property usually points to a common system temp directory, i.e. /tmp. On Apache Tomcat, it is usually the \$TOMCAT_HOME/temp directory.

The default logging configuration (log4j.xml bundled in the clover.war) may be changed to another log4j configuration file using system property log4j.configuration. If you override the configuration, only the properties from the new file are used.

The log4j.configuration should contain the URL of the new log4j configuration file, not a simple file system path, for example:

```
log4j.configuration=file:/home/clover/config/log4j.xml
```



Tip

It is better to copy the original file and modify the copy, than to create a new file.

Please note that log4j.configuration is not a **CloverDX Server** configuration property, but a system property, thus it must be set on the JVM command line by -Dlog4j.configuration or in other way suitable for the application container. See the <u>Installation (p. 11)</u> chapter for more information on how to set a system property for each application container.

Since such a configuration overrides the default configuration, it may influence Graph run logs. So your own log configuration has to contain following fragment to preserve Graph run logs:

```
<logger name="Tracking" additivity="false">
<level value="debug"/>
</logger>
```

Another Useful Logging Settings

These system properties allow for logging of HTTP requests/responses to stdout:

Client side:

com.sun.xml.ws.transport.http.client.HttpTransportPipe.dump=true (For more information, see the <u>Integrating CloverDX Designer with CloverDX Server</u> chapter of the CloverDX Designer User's Guide.) Server side:

com.sun.xml.ws.transport.http.HttpAdapter.dump=true

Access Log in Apache Tomcat

If you need to log all requests processed by the server, add the following code to $CATALINA_HOME/conf/server.xml$.

```
<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
prefix="localhost_access_log" suffix=".txt"
pattern="%h %l %u %t %D %r %s %b" />
```

The format defined above has following meaning

```
[IP address] [date-time] [processing duration in milliseconds] [method] [URL] [protocol
```

The log will look like the next line

```
172.17.30.243 - - [13/Nov/2014:12:53:03 +0000] 2 "POST /clover/spring-rpc/clusterNodeAp
```

See also <u>Valve</u> in documentation on Apache Tomcat.

Application Server Logs

If you use Apache Tomcat, it logs into \$CATALINA_HOME/logs/catalina.out file.

Graph Run Logs

Each graph or jobflow run has its own log file – for example, in the Server Console, section Execution History (p. 199).

By default, these log files are saved in the subdirectory cloverLogs/graph in the directory specified by java.io.tmpdir system property.

It's possible to specify a different location for these logs with the CloverDX graph.logs_path property. This property does not influence main Server logs.

Server Audit Logs

Server Audit Log logs operations called on ServerFacade and JDBC proxy interfaces.

Audit logging can be enabled by setting (adding) the value of CloverDX property logging.logger.server_audit.enabled to true. In server GUI, you can change the property value in **Configuration** \rightarrow **Setup** \rightarrow **Configuration File**. *Audit logging* is disabled by default.

The name of output file is server-audit.log. The file is in the same directory as main server log files. Default log level is DEBUG, so all operations which may do any change or another important operations (e.g. login or openJdbcConnection) are logged. To enable logging of all operations, change log level to TRACE in the log4j configuration.

Each logged operation is logged by two messages: entering method and exiting method (if the exception is raised, it's logged instead of output parameters)

- Entering method (marked as "inputParams"). All method's parameters (except for passwords) are printed.
- Exiting method (marked as "outputParams"). Method's return value is printed.

• Exception in method (marked as "EXCEPTION"). Exception's stacktrace is printed.

Message also contains:

- username, if the user is known
- client IP address, if it's known
- cluster node ID
- Interface name and the operation name

Values of transient and lazy initialized (in entity classes) fields and fields with binary content are not printed.

Designer-Server Integration Logs

The logging of Designer-Server integration can be enabled with <u>logging.logger.server integration.enabled</u> (p. 86) configuration property. The name of the log file is server-integration.log.

The log format is date and time, IP address of Designer, user name, operation, result of the operation (success/failure) and duration in milliseconds.

2018-03-07 16:42:00,525 10.0.3.2 user=clover, operation=executeGraph SUCCESS duration=576 ms

Worker Log

Worker log logs operation performed by the Worker. The Worker is configured with a separate log4j.xml configuration file.

Part IV. Administration

Chapter 18. Monitoring

Monitoring section in the Server GUI displays useful information about current performance of the standalone CloverDX Server or all Cluster nodes if the Clustering is enabled.

Monitoring section of the Standalone server has slightly different design from the Cluster environment - the serverview is the same as node detail in Cluster environment.

The section is refreshed every 15 seconds so the displayed data stays up-to-date. The page can also be refreshed manually by the **Refresh** button.

Standalone Server Detail

Standalone Server detail view displays info collected from the standalone Server. The info is grouped in several panels. The following panels are displayed by default.

- <u>Resource Utilization</u> (p. 108)
- Worker and System (p. 109)
- <u>License</u> (p. 109)
- <u>Performance</u> (p. 109)
- <u>CPU Load</u> (p. 110)
- <u>Running Jobs</u> (p. 110)
- Status History (p. 110)

You can display the hidden actions with **Actions** button: choose **Actions** \rightarrow **Show details**.

CloverDX 5.1.0.14		✓ Page Loaded 20	18-12-17 08:54 +01:00 📮 😌
Monitoring Status History Server Logs Server	RESOURCE UTILIZATION 0 1 of 80 GB 0 1 of 18 GB 2.0 of 18 GB 0.1 of 12 GB 0.2 of 12 GB 0.1 of 12 GB <th>1 RUNNING JOB 12059063 twe status HISTORY (LAST 10) READY 2016-12-1 updated by updated by updated by updated by updated by</th> <th>default graphflest.grf I nobiol 07 39 32</th>	1 RUNNING JOB 12059063 twe status HISTORY (LAST 10) READY 2016-12-1 updated by updated by updated by updated by updated by	default graphflest.grf I nobiol 07 39 32
HTTPS Connectors Event Listeners	Name [worker0@node01] UTC time 2018-12-17 07-54.04 GMT Ports 0 10550, 10551, 10592, Uptime 1 hr 14 mins Running jobs 0 Clock drift 0 2 ms	STOPPED updated by 2018-12-13	1 07:39:16 node01 7 07:38:20
E Sandboxes	PERFORMANCE	READY updated by 2018-12-11 updated by 2018-12-11 updated by 2018-12-10	node01 5 07:39:32 node01 5 07:39:17
Users Groups Security	and Doumury (1)	STOPPED updated by 2018-12-10 updated by updated by	node01 5 07:38:20 node01
Temp Spaces System Info	12 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	STARTING updated by 2018-12-1!	5 07:39:31 node01 5 07:39:16
 CloverDX Info Export Import 	CPU LOAD Select time interval	STOPPED updated by 2018-12-12 READY updated by 2018-12-12	node01 5 07:38:19 node01 1 21:06:53
• Setup 🛛	2 75 75 75 75 75 75 75 75 75 75 75 75 75		

Figure 18.1. Standalone server detail

Resource Utilization

Resource Utilization panel shows the amount of used System memory, System disk swap Server Core heap memory, Server Core non-heap memory, Worker heap memory and Worker non-heap memory.



Figure 18.2. Resource Utilization

Worker and System

Contain basic information about Worker and operating system.

WORKER		SYSTEM		
in or main the		or or citi		
Status	RUNNING	Current time	2018-12-10 13:35:31 CET	
Uptime	3 mins 37 secs	Java timezone	Europe/Prague	
Name	[worker0@node01]	UTC time	2018-12-10 12:35:31 GMT	
Ports ()	10500, 10501, 10502, -, -	Uptime	4 mins 10 secs	
Running jobs	1	Clock drift 🚯	2 ms	



License

Contains information about license.

CENSE			
NAME	VALUE		
License expiration	42 days		
License number			
License location	Database		



Performance

The Performance panel contains a chart with three basic performance statistics: a number of running jobs, amount of used up Server and Worker heap memory. The graph displays values gathered within a specific interval. The interval can be set up with the combo box above the graph or configured by the cluster.node.sendinfo.history.interval config property.

Note that the heap memory is constantly oscillating, even in idle state, since it is periodically managed by JVM garbage collector (i.e. the temporary data required for running **CloverDX Server** and Worker is periodically removed from/allocated to the heap memory).





CPU Load

The CPU Load panel displays a chart with info about total CPU load and CPU load caused by JVM (both Core Server and Worker).



Figure 18.6. CPU Load

Running Jobs

Running jobs panel lists currently running jobs, 10 most recent runs are displayed.



Figure 18.7. Running jobs

Status History

Status history panel displays node statuses history since restart of the Server.

STATUS HISTORY (LAST 10)

READY	updated by node01 2018-12-10 13:31:55
STARTING	updated by node01 2018-12-10 13:31:22
STOPPED	updated by node01 2018-12-10 13:30:30
READY	updated by node01 2018-12-10 13:23:04
O STARTING	updated by node01 2018-12-10 13:22:33
STOPPED	updated by node01 2018-12-10 13:17:16
O READY	updated by node01 2018-12-10 13:06:23
O STARTING	updated by node01 2018-12-10 13:05:25
STOPPED	updated by node01 2018-12-10 13:03:13
O READY	updated by node01 2018-12-10 10:37:14

Figure 18.8. Status History

Users' Accesses

This panel lists info about activities on files performed by users. The list displays a timestamp of an event, username, address and name of the method.

USERS'	ACCESSES	

DATE	USERNAME	ADDRESS	METHOD
2018-12-10 13:57:36 CET	clover	127.0.0.1	executeGraphAsync
2018-12-10 13:57:34 CET	clover	127.0.0.1	executeGraphAsync
2018-12-10 13:56:24 CET	clover	127.0.0.1	executeGraphAsync
2018-12-10 13:56:22 CET	clover	127.0.0.1	executeGraphAsync
2018-12-10 13:33:44 CET	clover	127.0.0.1	executeGraphAsync

Figure 18.9. Users' Accesses panel

Classloader cache

Classloader cache lists all currently cached classloaders. The classloader cache may be empty as classloader caching is disabled by default.

Status

Status panel displays current node status since last server restart. It displays current server status, exact Java version, exact **CloverDX Server** version, way of access to database, etc.

STATUS	
NAME	VALUE
Last status check time	2018-12-10 14:13:32
Available heap memory (used / commited / max)	499 MB / 666 MB / 1,820 MB
Available non-heap memory (used / commited / max)	195 MB / 202 MB / 2,288 MB
Status	READY
CloverDX version	5.1.0.6
Java version	1.8.0_144
Active threads	79
Loaded classes	20370
Running jobs	0
Active locks	0
Event listeners	
Number of job pools	0
Number of jobs in the pools	0
Number of transformation definitions in use	0
Nodes accessible by asynchronous messaging	
Nodes accessible by synchronous messaging	
Database access	local PostgreSQL JDBC Driver 42.2.5
Asynchronous messaging info	



Resource Utilization Detail

Provides information about CloverDX Server utilization of operating system resources.

RESOURCE UTILIZATION DETAIL	
NAME	VALUE
Average load	0.0
Time CPU spent idle	83%
CPU queue length	0
Swap I/O	0 / 0 (IN / OUT)
Direct memory	0 MB / 0 MB
Free temp space	127 GB

Figure 18.11. Status

Heartbeat

Heartbeat panel displays a list of heartbeat events and their results.

FARIBEAL						
LARIDEAL						
CURRENT TIME	CLOCK DRIFT	HEAP MEM USED	ACTIVE THREADS	LOADED CLASSES	RUNNING JOBS	-
2018-12-10 14:21:13	0 ms	354 MB	79	20218	0	
2018-12-10 14:21:11	0 ms	354 MB	79	20218	0	
2018-12-10 14:21:09	0 ms	354 MB	79	20218	0	
2018-12-10 14:21:07	0 ms	354 MB	79	20218	0	
2018-12-10 14:21:05	0 ms	353 MB	81	20218	0	
2018-12-10 14:21:03	0 ms	353 MB	79	20218	0	
2018-12-10 14:21:01	0 ms	326 MB	79	20218	0	-



Threads

Threads panel lists Java threads and their states.

THREADS			
THREAD NAME	THREAD STATE	WAITED TIME (MS)	BLOCKED TIME (MS)
AbortedJobsCleanup	TIMED_WAITING	-1	-1
AsyncFileHandlerWriter-491044090	TIMED_WAITING	-1	-1
Attach Listener	RUNNABLE	-1	-1
ContainerBackgroundProcessor[Sta ndardEngine[Catalina]]	TIMED_WAITING	-1	-1
DRDAConnThread_2	WAITING	-1	-1
DRDAConnThread_3	WAITING	-1	-1
DefaultQuartzScheduler_QuartzSch edulerThread	TIMED_WAITING	-1	-1



Quartz

Quartz panel lists scheduled actions: their name, description, start time, end time, time of previous event, time of next event and expected final event.

QUARTZ						
NAME	DESCRIPTION	START TIME	END TIME	PREVIOUS EVENT (WITHIN THIS UP- TIME)	NEXT EVENT	EXPECTED FINAL EVENT
trigger_45	Delete old execution artifacts	2018-12-10 10:32:57		2018-12-10 14:00:00	2018-12-10 15:00:00	
trigger_47	Delete old debug files	2018-12-10 10:32:57		2018-12-10 14:20:00	2018-12-10 14:30:00	

Figure 18.14. Quartz

Cluster Overview

Cluster overview displays info collected from all cluster nodes. The info is grouped in several panels:

- · List of nodes with a toolbar allows manipulation with selected nodes
- Status history Displays last 10 status changes for all cluster nodes
- Node detail Displays several basic performance attributes for selected nodes. It's visible on the right side only when activated by button on the toolbar.
- Running jobs It's displayed only when there are running jobs.



Figure 18.15. Cluster overview

Node Detail

Node Detail is similar to the <u>Standalone Server Detail</u> (p. 108) mentioned above, however it displays detail info about node selected in the menu on the left.



Figure 18.16. Node detail

Server Logs

Server Logs tab allows user to investigate log messages logged on other cluster nodes. Since the log messages are collected in memory, the maximum number of collected messages is relatively low by default, however it's customizable.

There are different "Log types":

- COMMON Ordinary server logs as stored in log files. It contains information on successful and unsuccessful logins, start and end of job execution etc.
- WORKER Worker related log.

The log file is \${java.io.tmpdir}/cloverlogs/worker-[node_name].log.

- CLUSTER Only cluster related messages are visible in this log. It contains information on job delegation and other types of messages related to cluster communication.
- LAUNCH_SERVICES Only requests for launch services (deprecated)
- AUDIT Detail logging of operations called on the **CloverDX Server** Core. Since the full logging may affect server performance, it's disabled by default. See <u>Server Audit Logs</u> (p. 105) for details

The log file is \${java.io.tmpdir}/cloverlogs/user-action.log.

• USER_ACTION - Contains some of user operations, e.g. login, logout, user creation, job execution, file synchronization (upload to server)

The corresponding log file is \${java.io.tmpdir}/cloverlogs/user-action.log.

• SERVER_INTERACTION - Interaction between Designer and Server.

The log file is \${java.io.tmpdir}/cloverlogs/server-integration.log.

CloverDX 5.1.0.1	4 ✔ Page Loaded node011/deveClaster 2018-12-17 09:33-91160 🔲 🤤
Monitoring Status History Server Logs	Node ID node81 • Log type COMMON • Show log Log size 500 Set size
▶₀ Execution History ₨₀ Task History ๓ Schedules ເ≩ Data Services	2011-12-12 00140124,340[mal initialize] twenthelper 1000 TaskProcessors dictionery map(email.com.clovert1.server.i.r@07040517, shell_command.com.clovert1.server.i.r@07040517, shell_command.com.clovert1.server.serv
Event Listeners Sandboxes Configuration Hep	2016:1:7:0:0:0:0:3:4:0:0:1:filling: 1 100 server initialing clustering 2016:1:7:0:0:0:3:4:0:0:1:filling: 1 100 server initialing: 100 2016:1:7:0:0:0:3:3:4:0:0:1:filling: 1 100 lices file system property clower license file:null 2016:1:7:0:0:0:3:3:4:0:0:1:filling: 1 100 license file system property clower license file:null 2016:1:7:0:0:0:3:3:4:0:0:1:filling: 1 100 license file system property clower license file:null 2016:1:7:0:0:0:3:3:4:0:0:1:filling: 1 100 trigot bask server claster for adabase 2016:1:7:0:0:0:3:3:4:0:0:1:filling: 1 100 trigot bask server claster(51:n) 2016:1:7:0:0:0:3:3:4:0:0:1:filling: 1 100 trigot bask server claster(51:n) 2016:1:7:0:0:0:3:3:4:0:0:1:filling: 1 100 trigot bask server claster(51:n) 2016:1:7:0:0:0:0:3:3:0:0:1:1:0:0:0:1:1:0:0:0:0:0
¢	<pre>2001-13-12 004-015.013[val_it[sal_iter] 1 INO Engine lagins locations: (Pluginication(fler/C/Users/Jan/Desktop/CloverOKGerver.5.1.0.Tomcat-9.0.10/webapps/clover/MEB-DW/plugins/org.jettl.spreadsheet.c 10 - org.jettl.sml version - 5.1.0 protect and the plugins locations: (Pluginication(fler/C/Users/Jan/Desktop/CloverOKGerver.5.1.0.Tomcat-9.0.10/webapps/clover/MEB-DW/plugins/org.jettl.spreadsheet.c version - 5.1.0 protect and the plugins locations: (Pluginication(fler/C/Users/Jan/Desktop/CloverOKGerver.5.1.0.Tomcat-9.0.10/webapps/clover/MEB-DW/plugins/org.jettl.spreadsheet.c id - org.jettl.sml version - 5.1.0 protect and the plugins location - 10 Plu</pre>

Figure 18.17. Server Logs

See also: Chapter 17, Logging (p. 104).

Using the Monitoring

Restarting the Worker

Switch to **Monitoring** > **Status** tab.

In the worker tile, click Actions button and select Finish jobs and restart from the menu.

The worker will be restarted.

See also Worker and System (p. 109).

Showing Worker's Command Line Arguments

Switch to **Monitoring** > **Status** tab.

In the worker tile, click the Actions button and select Show command line.

The command line arguments and parameters will be displayed.

See also Worker and System (p. 109).

Suspending the Server

Switch to **Monitoring** > **Status** tab.

Click the Actions button (in the upper left corner) and select Suspend from the menu.

The server will be suspended. If there is a job running, the node will wait until it finishes.

If you need to suspend the node immediately without waiting for jobs to finish, use **Suspend at once** instead of **Suspend**.

Resuming the Server

Switch to **Monitoring** > **Status** tab.

Click the Actions button (in the upper right corner) and select Resume from the menu.

The server will be resumed.

Displaying List of Threads of the Server Core

Switch to **Monitoring** > **Status** tab.

Click the **Actions** button (in the upper right corner) and select **Show Details**. Several new tiles will appear. Search for the **Threads** tile.

Chapter 19. Temp Space Management

Many of the components available in the **CloverDX Server** require temporary files or directories in order to work correctly. *Temp space* is a physical location on the file system where these files or directories are created and maintained. **CloverDX Server** allows you to configure and manage temp spaces - you can specify their locations, see usage of the filesystem etc.

To access this administration section, you need Temp Space Management permission (p. 141).

Overview

The overview of temp spaces defined in **CloverDX Server** is available under *Configuration > Temp Spaces*.

The overview panel displays list of temp spaces for each node in the Cluster. These properties are displayed for each temp space:

- Node name of the node on which the temp space is located (only in Clustered environment)
- Root path location of the temp space with unresolved placeholders (see note below for placeholders)
- **Resolved path** location of the temp space with resolved placeholders (see note below for placeholders)
- Free space remaining space for the temp space
- File system size all available space for the temp space (actual size of the filesystem where the temp space resides)
- File system usage size of used space in percentage
- Available the directory exists and is writable
- Status current status of temp space, can be Active or Disabled



Note

It is possible to use system properties and environment variables as placeholders. See <u>Using</u> environment variables and system properties (p. 118).

New Temp Space								
Node	Root path	Resolved path	Free space	File system size	File system usage	Available	Status	
virt-alpha	\${java.io.tmpdir}/clover_temp_virt-alpha	/opt/apache-tomcat- 9.0.12/temp/clover_temp_virt-alpha	9.8 GB	44.7 GB	78%	⊘	Active	
virt-gray	\${java.io.tmpdir}/clover_temp_virt-gray	/opt/tomcat9/temp/clover_temp_virt-gray	19.9 GB	31.4 GB	36%	\bigcirc	Active	
virt- proteus	{java.io.tmpdir}/clover_temp_virt-proteus	/opt/clover-geo- cluster/tempspace/clover_temp_virt-proteus	13.3 GB	24.9 GB	46%	0	Active	
krychle- brno	{java.io.tmpdir}/clover_temp_krychle-brno	/opt/clover-geo- cluster/tempspace/clover_temp_krychle-brno	287.7 GB	299.9 GB	4%	0	Active	

Figure 19.1. Configured temp spaces overview - one default temp space on each cluster node

Management

Temp space management offers an interface with options to add, disable, enable and delete a temp space.

Initialization (p. 118) Adding Temp Space (p. 118) Using environment variables and system properties (p. 118) Disabling Temp Space (p. 119) Enabling Temp Space (p. 119) Removing Temp Space (p. 119)

Initialization

When **CloverDX Server** is starting the system checks temp space configuration: in case no temp space is configured a new default temp space is created in the directory where java.io.tmpdir system property points. The directory is named as follows:

- \${java.io.tmpdir}/clover_temp in case of a standalone Server
- \${java.io.tmpdir}/clover_temp_<node_id> in case of Server Cluster

Adding Temp Space

In order to define new temp space, click the **New Temp Space** button and specify its path. In case of Cluster environment, specify the node on which the new temp space should be created. If the directory entered does not exist, it will be created.



Tip

The main point of adding additional temp spaces is to enable higher system throughput - therefore the paths entered should point to directories residing on different physical devices to achieve maximal I/O performance.

Using environment variables and system properties

Environment variables and system properties can be used in the temp space path as a placeholder; they can be arbitrarily combined and resolved paths for each node may differ in accord with its configuration.



Note

The environment variables have higher priority than system properties of the same name. The path with variables are resolved after system has added new temp space and when the Server is starting. In case the variable value has been changed, it is necessary to restart the Server so that the change takes effect.

Examples:

- Given that an environment variable USERNAME has a value cloverdxUser. and is used as a placeholder in the path C:\Users\\${USERNAME}\tmp, the resolved path is C:\Users\cloverdxUser\tmp.
- Given that Java system property java.io.tmpdir has a value C:\Users\cloverdxUser \AppData\Local\Temp and the property is used as a placeholder in the path \${java.io.tmpdir}\temp_folder, the resolved path is C:\Users\cloverdxUser\AppData \Local\Temp\temp_folder.
- Node node01 has been started with -Dcustom.temporary.dir=C:\tmp_node01 parameter. Node node02 has been started with -Dcustom.temporary.dir=C:\tmp_node02 parameter. The declared

path is ${custom.temporary.dir}$. The resolved path is different for each node, C:\tmp_node01 for node01 and C:\tmp_node02 for node02.

• When the declared path is \${java.io.tmpdir}\\${USERNAME}\tmp_folder, the resolved path is C: \tmp\cloverdxUser\tmp_folder.

Disabling Temp Space

To disable a temp space, click on the three vertical dots menu on the right side of the respective temp space and select **Disable**. Once the temp space has been disabled, no new temporary files will be created in it, but the files already created may be still used by running jobs. In case there are files left from previous or current job executions a notification is displayed.



Note

The system ensures that at least one enabled temp space is available.

Enabling Temp Space

To enable a temp space, click on the three vertical dots menu on the right side of the respective disabled temp space and select **Enable**. Enabled temp space is active, i.e. available for temporary files and directories creation.

Removing Temp Space

To remove a temp space, click on the three vertical dots menu on the right side of the respective temp space and select **Delete**. Only disabled temp spaces may be removed. If there are any running jobs using the temp space, the system will not allow its removal.

Chapter 20. Secure Parameters

<u>Secure parameters configuration</u> (p. 122) <u>Installing Bouncy Castle JCE Provider</u> (p. 123)

Transformation graphs in **CloverDX Server** environment allow you to define secure graph parameters. Secure graph parameters are regular graph parameters, either internal or external (in a * .prm file), but the values of the graph parameters are not stored in plain text on the file system - encrypted values are persisted instead. This allows you to use graph parameters to handle sensitive information, typically credentials such as passwords to databases.

Secure parameters are only available in **CloverDX Server** environment, including working with **CloverDX Server** Projects in **CloverDX Designer**.

The encryption algorithm must be initialized with a **master password**. The master password has to be manually set after Server installation in **Configuration** \rightarrow **Security**. Secure parameters cannot be used before the master password is set.

The maximum length of the master password is 255 characters; there are no other restrictions or complexity requirements.

Master password			
8	The master password is used to encrypt/decrypt secure parameters.		
A	The master password has not been set yet!		
	Set master password		

Figure 20.1. Master password initialization

After setting the master password secure parameters are fully available in **Graph parameter editor** in **CloverDX Designer**. When setting value of a secure parameter, it will be automatically encrypted using the master password. Secure parameters are automatically decrypted by the Server in graph runtime. A parameter value can also be encrypted in the **CloverDX Server** Console in the *Configuration* > *Security* > *Secure Parameters* page - use the **Encrypt text** section.



Figure 20.2. Graph parameters tab with initialized master password

If you change the master password, the secure parameters encrypted using the old master password cannot be decrypted correctly anymore. In that case existing secure parameters need to be encrypted again with the new master password. That can be accomplished simply by setting their value (non-encrypted) again in the **Graph parameter editor**. Similar master password inconsistency issue can occur if you move a transformation graph with some secure parameters to another Server with a different master password. So it is highly recommended to use the identical master password for all your **CloverDX Server** installations.

See documentation of secure parameters in **CloverDX Designer** manual for further details.

Secure parameters configuration

Encryption of secure parameters can be further customized via Server configuration parameters.

Table 20.1. Secure parameters configuration parameters

Property name	Default value	Description
security.job_parameters.encry	ptor.algorithm PBEWithMD5AndDES	The algorithm to be used for encryption. This algorithm has to be supported by your JCE provider (if you specify a custom one, or the default JVM provider if you don't). The name of algorithm should start with <i>PBE</i> prefix. The list of available algorithms depends on your JCE provider, e.g. for the default <i>SunJCE</i> provider you can find them on SunJCEProvider or for the <i>Bouncy Castle</i> provider on Bouncy Castle Specifications (section <i>Algorithms/PBE</i>)).
security.job_parameters.encry	ptor.master_password_encryp	tion.password
	clover	The password used to encrypt values persisted in the database table <i>secure_param_passwd</i> (the master password is persisted there).
security.job_parameters.encry	ptor.providerClassName Empty string. The default JVM provider is used (e.g. for Oracle Java the SunJCE provider is used)	The name of the security provider to be asked for the encryption algorithm. It must implement <i>java.security.Provider</i> interface. For example set to <i>org.bouncycastle.jce.provider.BouncyCastleProvide</i> for the <i>Bouncy Castle</i> JCE provider, see below.

Installing Bouncy Castle JCE Provider

Algorithms provided by JVM could be too weak to satisfy an adequate security. Therefore it is recommended to install a third-party JCE provider. Following example demonstrates installation of one concrete provider, *Bouncy Castle* JCE provider. Another provider would be installed similarly.

- 1. Download Bouncy Castle provider jar (e.g. bcprov-jdk15on-150.jar) from http://bouncycastle.org/ latest_releases.html
- 2. Add the jar to the classpath of your application container running **CloverDX Server**, e.g. to directory WEB-INF/lib
- 3. Set value of the *security.job_parameters.encryptor.providerClassName* attribute to *org.bouncycastle.jce.provider.BouncyCastleProvider* in the config.properties file.
- 4. Set value of the *security.job_parameters.encryptor.algorithm* attribute to the desired algorithm (e.g. *PBEWITHSHA256AND256BITAES-CBC-BC*).

Example of configuration using Bouncy Castle:

security.job_parameters.encryptor.algorithm=PBEWITHSHA256AND256BITAES-CBC-BC
security.job_parameters.encryptor.providerClassName=org.bouncycastle.jce.provider.BouncyCastleProvider

Chapter 21. Users and Groups

The **CloverDX Server** has a built-in security module that manages users and groups. User groups control access permissions to sandboxes and operations the users can perform on the Server, including authenticated calls to Server API functions. A single user can belong to multiple groups.

LDAP or Active Directory can be configured with the Server to authenticate users and optionally assign their effective groups (and permissions) from a global directory.

You can manage users and user groups in **Configuration > Users and Groups**. Please note that you need a **List users** and **List groups** permissions, respectively.

LDAP Authentication

<u>Configuration</u> (p. 126) <u>Basic LDAP connection properties</u> (p. 126) <u>Configuration of user and group lookup</u> (p. 126)

Since version 3.2, it is possible to configure the Server to use an LDAP server for users authentication. This way, the credentials of users registered in LDAP may be used for authentication to any **CloverDX Server** interface (API or web console).

However, the authorization (access levels to sandboxes content and privileges for operations) is still handled by the **CloverDX** security module. Each user, even when logged-in using LDAP authentication, must have their own "user" record (with related groups) in the **CloverDX** security module. So there must be a user with the same username and domain set to "LDAP". Such a record has to be created by a Server administrator before the user can log in.

To configure LDAP authentication, use the Setup GUI (p. 60).

Use LDAP for user authentication only

If this **Authentication Policy** is selected, **CloverDX Server** uses LDAP directory to verify only the user's password:

- 1. The user specifies the LDAP credentials in the login form to the Server web console.
- 2. CloverDX Server looks up the user's record and checks whether it has the "LDAP" domain set.
- 3. The Server attempts to connect to the LDAP server using the user's credentials. If it succeeds, the user is logged in.

Use LDAP for user authentication and user synchronization

In this mode, **CloverDX Server** verifies user's credentials and synchronizes additional information (group, name and email) with those stored in LDAP.

In case the Server is configured for user authentication and group synchronization, the procedure is as follows:

- 1. The user specifies the LDAP credentials in the login form to the Server web console.
- 2. CloverDX Server looks up the user's record and checks whether it has the "LDAP" domain set.
- 3. CloverDX Server connects to the LDAP server and checks whether the user exists (it uses specified search to lookup in LDAP).
- 4. If the user exists in LDAP, CloverDX Server performs authentication.
- 5. If the authentication is successful, CloverDX Server searches LDAP for user's groups.
- 6. CloverDX user is assigned to the CloverDX groups according to his current assignation to the LDAP groups.
- 7. User is logged-in.



Note

Switching domains:

• If a user was **created as LDAP** and then switched to clover domain, you have to **set a password** for him in the **Change password** tab.

• If a user was **created as clover** and then switched to LDAP domain, they have a password in clover domain, but it is overridden by the LDAP password. After switching back to clover domain, the **original password is re-used**. It can be reset in the **Change password** tab, if needed.

Configuration

By default **CloverDX Server** allows only its own internal mechanism for authentication. To enable authentication with LDAP, set the configuration property "security.authentication.allowed_domains" properly. It is a list of user domains that are used for authentication.

Currently there are 2 authentication mechanism implemented: "LDAP" and "clover" ("clover" is an identifier of **CloverDX** internal authentication and may be changed by security.default_domain property, but only for white-labeling purposes). To enable LDAP authentication, set value to "LDAP" (only LDAP) or "clover,LDAP". Users from both domain may login. It's recommended to allow both mechanisms together, until the LDAP is properly configured. So the admin user can still login to web GUI although the LDAP connection isn't properly configured.

You can use Setup to configure LDAP authentication. See LDAP (p. 60) in Chapter 13, Setup (p. 54).

Basic LDAP connection properties

```
# Implementation of context factory.
security.ldap.ctx_factory=com.sun.jndi.ldap.LdapCtxFactory
# URL of LDAP server.
security.ldap.url=ldap://hostname:port
# User DN pattern that will be used to create LDAP user DN from login name.
security.ldap.user_dn_pattern=uid=${username},dc=company,dc=com
```

Depending on the LDAP server configuration the property security.ldap.user_dn_pattern can be pattern for user's actual distinguished name in the LDAP directory, or just the login name - in such case just set the property to \${username}.

Configuration of user and group lookup

In order to be able to synchronize the **Clover** groups with those defined in LDAP directory, the security.ldap.user_dn_pattern has to be left unspecified. There are additional properties required so that the server is able to search the LDAP directory.

```
# User DN of a user that has sufficient privileges to search LDAP for users and groups.
security.ldap.userDN=cn=Manager,dc=company,dc=com
# The password for user mentioned above.
security.ldap.password=
```

There are optional settings affecting how the LDAP directory is searched.

```
# Timeout for queries searching the directory.
security.ldap.timeout=5000
# Maximal number of records that the query can return.
security.ldap.records_limit=2000
# How LDAP referrals are processed, possible values are: 'follow', 'ignore' and 'throw'.
# The default depends on the context provider.
security.ldap.referral=
```

Specified values work for this specific LDAP tree:

```
• dc=company,dc=com
```

• ou=groups

- cn=admins
- (objectClass=groupOfNames,member=(uid=smith,dc=company,dc=com),member=(uid=jones,dc=company,dc=com))
- cn=developers (objectClass=groupOfNames,member=(uid=smith,dc=company,dc=com))
- cn=consultants (objectClass=groupOfNames,member=(uid=jones,dc=company,dc=com))
- ou=people
 - uid=smith (fn=John,sn=Smith,mail=smith@company.com)
 - uid=jones (fn=Bob,sn=Jones,mail=jones@company.com)

Following properties are necessary for lookup for the LDAP user by his username. (step [4] in the login process above)

```
# Base specifies the node of LDAP tree where the search starts.
security.ldap.user_search.base=dc=company,dc=eu
# Filter expression for searching the user by his username.
# Note, that this search query must return just one record.
# Placeholder ${username} will be replaced by username specified by the logging user.
security.ldap.user_search.filter=(uid=${username})
# Scope specifies type of search in "base". There are three possible values: SUBTREE | ONELEVEL | DBJECT
# http://download.oracle.com/javase/8/docs/api/javax/naming/directory/SearchControls.html
security.ldap.user_search.scope=SUBTREE
```

Following properties are names of attributes from the search defined above. They are used for getting basic info about the LDAP user in case the user record has to be created/updated by **CloverDX** security module: (step [6] in the login process above)

```
security.ldap.user_search.attribute.firstname=fn
security.ldap.user_search.attribute.lastname=sn
security.ldap.user_search.attribute.email=mail
# This property is related to the following step "searching for groups".
# Groups may be obtained from specified user's attribute, or found by filter (see next paragraph)
# Leave this property empty if the user doesn't have such attribute.
security.ldap.user_search.attribute.groups=memberOf
```

In the following step, **CloverDX** tries to find groups which the user is assigned to. (step [4] in the login process above). There are two ways how to get list of groups which the user is assigned to. The user-groups relation is specified on the "user" side. The user record has some attribute with list of groups. It's "memberOf" attribute usually. Or the relation is specified on the "group" side. The group record has an attribute with list of assigned users. It's "member" attribute usually.

In case the relation is specified on users side, please specify property:

security.ldap.user_search.attribute.groups=memberOf

Leave it empty otherwise.

In case the relation is specified on the groups side, set properties for searching:

```
security.ldap.groups_search.base=dc=company,dc=com
# Placeholder ${userDN} will be replaced by user DN found by the search above.
# If the filter is empty, searching will be skipped.
security.ldap.groups_search.filter=(&(objectClass=groupOfNames)(member=${userDN}))
security.ldap.groups_search.scope=SUBTREE
```

Otherwise, please leave property security.ldap.groups_search.filter empty, so the search will be skipped.

Clover user record will be assigned to the clover groups according to the LDAP groups found by the search (or the attribute). (Groups synchronization is performed during each login)

Value of the following attribute will be used for lookup for the CloverDX group by its code.

So the user will be assigned to the CloverDX group with the same "code".
security.ldap.groups_search.attribute.group_code=cn

Users

This section is intended to users management. It offers features in dependence on user's permissions (i.e. a user may enter this section, but cannot modify anything, or they may modify, but cannot create new users).

The Users section of the Configuration menu allows you to:

<u>Create New User</u> (p. 129) <u>Edit Users Record</u> (p. 129) <u>Change Users Password</u> (p. 129) <u>Assign Users to Groups</u> (p. 130) <u>Disable / Enable Users</u> (p. 130)

After default installation on an empty database, the admin user is created automatically.

Table 21.1. Admin user

User name	Description
clover	Clover user has admin permissions, thus default password clover should be changed after installation.

Create New User

When creating a new User, you must enter the following information:

Attribute	Description
Domain	Domain which is the origin of the user.r Currently, there are only two possible values: "clover" or "ldap".
Username	A common user identifier. Must be unique, cannot contain spaces or special characters, just letters and numbers.
First name	The user's first name.
Last name	The user's last name.
E-mail	Email address which may be used by CloverDX administrator or by CloverDX Server for automatic notifications. See <u>Send an Email</u> (p. 173) for details.
Password	Case sensitive password. If the user looses his password, the new one must be set. The password is stored in an encrypted form for security reasons, so it cannot be retrieved from a database and must be changed by the user who has proper permission for such operation.
Verify password	Verify the entered password.

Edit Users Record

A user with a Create user or Edit user permission can use this form to set basic user parameters.

Change Users Password

If user looses his password, the new one must be set. So a user with the **Change passwords** permission can use this form to do it.

Assign Users to Groups

Assignment to groups gives the user proper permissions. Only logged user with the **Groups assignment** permission can access this form and specify groups which the user is assigned in. For details about permissions, see <u>Groups</u> (p. 131).

Disable / Enable Users

Since a user record has various relations to the logs and history records, it can't be deleted. So it is disabled instead. This means that the record doesn't display in the list and the user can't login.

However, a disabled user may be enabled again. **Note** that the disabled user is removed from their groups, so groups should be assigned properly after re-enabling.

Groups

Group is an abstract set of users, which gives assigned users some permissions. So it is not necessary to specify permissions for each single user.

There are independent levels of permissions implemented in CloverDX Server

- *permissions to Read/Write/eXecute in sandboxes* The sandbox owner can specify different permissions for different groups. For details, see <u>Sandbox Content Security and Permissions</u> (p. 148).
- *permissions to perform some operation* user with an operation permission **Permission assignment** may assign specific permission to existing groups.
- *permissions to launch specific service* For details, see Chapter 39, <u>Launch Services</u> (p. 250), Chapter 40, <u>Data Services</u> (p. 257).

Table 21.3. Default groups created during installation

Group name	Description
admins	This group has an operation permission all assigned, which means, that it has unlimited permission. Default user clover is assigned to this group, which makes him administrator.
all users	By default, every single CloverDX user is assigned to this group. It is possible to remove a user from this group, but it is not a recommended approach. This group is useful for some permissions to sandbox or some operation, which you would like to make accessible for all users without exceptions.

Users Assignment

Relation between users and groups is N:M. Thus in the same way, how groups are assignable to users, users are assignable to groups.

Groups permissions

Groups permissions are structured as a tree, where permissions are inherited from the root to leafs. Thus if some permission (tree node) is enabled (blue dot), all permissions in sub tree are automatically enabled (white dot). Permissions with red cross are disabled.

Thus for the **admin** group just the **all** permission is assigned, every single permission in the sub tree is assigned automatically.

With none of the following privileges, a user can: log into the Server console, create a server project (in Designer) from its own sandbox, create a file in its own existing sandbox, and run graphs.

• all

The user with this permission has all available permissions. The Admin group has all permissions by default.

• Unlimited access to sandboxes

Allows the user to perform operations on all sandboxes, even if the sandbox accessibility is not specified explicitly.

This permission does not include the suspend sandbox permission (p. 139).

• Sandboxes

Allows the user to work with sandboxes. This permission contains all the permissions below. The user can perform operations only on sandboxes owned by himself or on sandboxes with explicitly added access to him, see Chapter 22, <u>Sandboxes - Server Side Job Files</u> (p. 145).

• List sandbox

In the Server web interface, it allows the user to list their sandboxes and sandboxes with **read** permission granted to the user's group.

In the Server web interface, this permission is necessary to create, edit, or delete sandboxes.

Within a sandbox with the **write** access granted, the user can edit or remove files and create or delete directories even without this permission.

• Create sandbox

Allows the user to create new sandboxes.

If a sandbox is to be created in web interface, the user must have the list sandbox permission (p. 132).

• Delete sandbox

Allows the user to delete sandboxes.

If a sandbox is to be deleted in web interface, the user must have the list sandbox permission (p. 132).

• Edit sandbox

Allows the user to edit sandboxes.

If a sandbox is to be modified in web interface, the user must have the list sandbox permission (p. 132).

• May delete files missing in uploaded ZIP

In **Sandbox** \rightarrow **Upload ZIP**, it allows the user to use a checkbox to delete files missing in the ZIP to be uploaded. If the user does not have this permission, the checkbox to delete mission files in ZIP is not displayed.

If a sandbox is to be uploaded from a ZIP file in the Server web interface, the user must have the <u>list</u> sandbox permission (p. 132).

• Scheduling

Allows the user to manage schedules, see Chapter 30, Scheduling (p. 193).

• List schedule

Allows the user to list all schedules.

• List schedule limited

Allows the user to list the enabled schedules.

• Create schedule

Allows the user to create new schedules.

The user must have the <u>list schedule limited permission</u> (p. 132) to access the scheduling section to create a new schedule.

• Delete schedule

Allows the user to delete schedules.

The user must have the <u>list schedule limited permission(p. 132)</u> or <u>list schedule permission(p. 132)</u> to access the scheduling section to delete the schedule.

• Edit schedule

Allows the user to edit schedules.

The user must have the <u>list schedule limited permission(p. 132)</u> or <u>list schedule permission(p. 132)</u> to access the scheduling section to edit the schedule.

• Event listeners

Allows the user to manage event listeners, see Chapter 32, Listeners (p. 204).

• List of Event Listeners

Allows the user to list all event listeners.

• List of Jobflow Event Listeners unlimited

Allows the user to list jobflow event listeners.

See Jobflow Event Listeners (p. 212)

List of Jobflow Event Listeners limited

Allows the user to list jobflow event listeners of sandboxes the user can read from.

• List of Graph Event Listeners unlimited

Allows the user to list all graph event listeners, see Graph Event Listeners (p. 206).

• List of Graph Event Listeners limited

Allows the user to list graph event listeners from sandboxes the user can read from.

• List of File Event Listeners unlimited

Allows the user to list all file event listeners, see File Event Listeners (remote and local) (p. 221).

• List of File Event Listeners limited

Allows the user to list all file event listeners.

• List of JMS Event Listeners unlimited

Allows the user to list all JMS listeners, see JMS Message Listeners (p. 214).

• List of JMS Event Listeners limited

Allows the user to list all JMS listeners.

• List of Universal Event Listeners unlimited

Allows the user to list all universal event listeners, see Universal Event Listeners (p. 219).

• List of Universal Event Listeners limited

Allows the user to list all universal event listeners.

See Universal Event Listeners (p. 219).

• List of Task Event Listeners unlimited

Allows the user to list all task event listeners, see Task Failure Listeners (p. 229).

• List of Task Event Listeners limited

Allows the user to list all task event listeners from sandboxes the user can read from.

See <u>Task Failure Listeners</u> (p. 229).

Create Event Listener

Allows the user to create event listeners.

If an event listener is to be created in the Server web interface, the user must have permission to list the event listeners of the particular type.

• Create Jobflow Event Listener

Allows the user to create new Jobflow Event listeners.

If a Jobflow event listener is to be created in the Server web interface, the user must have the <u>list of</u> jobflow event listeners limited permission (p. 133).

See Jobflow Event Listeners (p. 212).

Create Graph Event Listener

Allows the user to create graph event listeners.

If a graph event listener is to be created in the Server web interface, the user must have the <u>list of graph</u> event listeners limited permission (p. 133).

See Graph Event Listeners (p. 206).

• Create File Event Listener

Allows the user to create graph event listeners.

If a file event listener is to be created in the Server web interface, the user must have the <u>list of file event</u> <u>listeners limited permission</u> (p. 133).

See File Event Listeners (remote and local) (p. 221).

• Create JMS Listener

Allows the user to create JMS event listeners.

If a JMS event listener is to be created in the Server web interface, the user must have the <u>list of JMS</u> event listeners limited permission (p. 133).

See JMS Message Listeners (p. 214).

Create Universal Event Listener

Allows the user to create universal event listeners.

If a universal event listener is to be created in the Server web interface, the user must have the <u>list of</u> <u>universal event listeners limited permission</u> (p. 133).

See Universal Event Listeners (p. 219).

• Create Task Event Listener

Allows the user to create task event listeners.

If a task event listener is to be created in the Server web interface, the user must have the <u>list of task</u> event listeners limited permission (p. 134).

See Task Failure Listeners (p. 229).

• Edit Event Listener

Allows the user to edit event listeners.

If an event listener is to be created in the Server web interface, the user must have permission to list event listener of the particular type.

• Edit Jobflow Event Listener

Allows the user to edit jobflow event listeners.

If a jobflow event listener is to be edited in the Server web interface, the user must have the <u>list of</u> jobflow event listeners limited permission (p. 133).

See Jobflow Event Listeners (p. 212).

• Edit Graph Event Listener

Allows the user to edit graph event listeners.

If a graph event listener is to be edited in the Server web interface, the user must have the <u>list of graph</u> event listeners limited permission (p. 133).

See Graph Event Listeners (p. 206).

• Edit File Event Listener

Allows the user to edit file event listeners.

If a file event listener is to be edited in the Server web interface, the user must have the <u>list of file event</u> <u>listeners limited permission</u> (p. 133).

See File Event Listeners (remote and local) (p. 221).

• Edit JMS Event Listener

Allows the user to edit JMS event listeners.

If a JMS event listener is to be edited in the Server web interface, the user must have the <u>list of JMS</u> event listeners limited permission (p. 133).

• Edit Universal Event Listener

Allows the user to edit universal event listeners.

If a universal event listener is to be edited in the Server web interface, user must have permission <u>list</u> of universal event listeners limited permission (p. 133).

See Universal Event Listeners (p. 219).

• Edit Task Event Listener

Allows the user to edit task event listeners.

If a task event listener is to be edited in the Server web interface, user must have permission <u>list of task</u> event listeners limited permission (p. 134).

See Task Failure Listeners (p. 229).

• Delete Event Listener

Allows the user to delete event listeners.

• Delete Jobflow Event Listener

Allows the user to delete jobflow event listeners.

The user must have the <u>delete graph event listener permission</u>(p. 136) to delete a jobflow event listener.

It a jobflow event listener is to be deleted in the Server web interface, the user must have the <u>list of</u> jobflow event listeners limited permission (p. 133)

• Delete Graph Event Listener

Allows the user to delete graph event listeners.

If a graph event listener is to be deleted in the Server web interface, the user must have the <u>list of graph</u> event listeners limited permission (p. 133).

See Graph Event Listeners (p. 206).

• Delete File Event Listener

Allows the user to delete file event listeners.

The user must have the <u>delete graph event listener permission</u> (p. 136) to delete a file event listener.

If a file event listener is to be deleted in the Server web interface, the user must have the <u>list of file event</u> <u>listeners limited permission</u> (p. 133).

See File Event Listeners (remote and local) (p. 221).

• Delete JMS Event Listener

Allows the user to delete JMS Event Listeners.

The user must have the delete graph event listener permission (p. 136) to delete a JMS event listener.

If a graph event listener is to be deleted in the Server web interface, the user must have the <u>list of JMS</u> event listeners limited permission (p. 133).

• Delete Universal Event Listener

Allows the user to delete universal event listeners.

The user must have the <u>delete graph event listener permission</u> (p. 136) to delete universal event listener.

If a universal event listener is to be deleted in the Server web interface, the user must have the <u>list of</u> <u>universal event listeners limited permission</u> (p. 133).

See Universal Event Listeners (p. 219).

Delete Task Event Listener

Allows the user to delete task event listeners.

If a task event listener is to be deleted in the Server web interface, the user must have the <u>list of task</u> event listeners limited permission (p. 134).

See Task Failure Listeners (p. 229).

• Manual task Execution

Allows the user to manually execute a task (send an email, execute a script, etc.) with an immediate effect.

See Chapter 29, Manual Task Execution (p. 192).

· Unlimited access to execution history

Allows the user to perform the same operations as <u>unlimited access to execution history list</u> <u>permission</u> (p. 137).

• Unlimited access to execution history list

Allows the user to view execution history of all jobs.

• Limited access to execution history list

Allows the user to view execution history of jobs from sandboxes the user can read from. In Designer, this permission is required to be able to view **Execution log** in Designer's console and execution history in **Execution** tab.

• Launch Services

Allows the user to list, create, edit, and delete launch services, see Chapter 39, Launch Services (p. 250).

• List Launch Services unlimited

Allows the user to list all launch services.

• List Launch Services Limited

Allows the user to list launch services from sandboxes the user can read from.

• Create Launch service

Allows the user to create a new launch service.

User has to have the <u>create graph event listener permission</u>(p. 134) to bind the launch service with a graph.

If the launch service is to be created in the Server web interface, the user has to have the <u>list launch services</u> <u>limited permission</u> (p. 137) (or the <u>list launch services unlimited permission</u> (p. 137) to access the section with launch services.

• Delete Launch Service

Allows the user to delete a launch service.

User has to have delete graph event listener permission (p. 136) to delete a launch service.

If the launch service is to be deleted in the Server web interface, the user must have the <u>list launch services</u> <u>limited permission</u> (p. 137) to access the section with launch services.
• Edit Launch Service

Allows the user to edit a launch services.

User has to have edit graph event listener (p. 135) to edit the launch service.

If the launch service is to be edited in the Server web interface, the user must have the <u>list launch services</u> <u>limited permission</u> (p. 137) to choose the launch service in the server interface.

• Data service

Allows the user to access the Data service section, see Chapter 40, Data Services (p. 257).

• List data services

Allows the user to list data services.

Manage data services

Allows the user to manage data services.

• Delete data services

Allows the user to delete data services.

• Execute and access documentation

Allows the user to execute and access documentation.

• Manage HTTPS connectors

Allows the user to manage HTTPS connectors.

• Tasks history

Allows the user to access the Tasks history section, see Chapter 28, Tasks (p. 172).

• Monitoring

Monitoring permission grants user all its subpermissions.

• Monitoring section

Allows the user to access the monitoring section.

See Chapter 18, Monitoring (p. 108).

• Suspend

Allows the user to suspend the server, a cluster node, or a sandbox.

The user must have the monitoring section permission (p. 138) to access the Monitoring section.

• Suspend server

Allows the user to suspend or resume the server.

The user must have the monitoring section permission (p. 138) to access the monitoring section.

• Suspend cluster node

Allows the user to suspend or resume a cluster node.

The user must have the monitoring section permission (p. 138) to access the monitoring section.

• Suspend sandbox

Allows the user to suspend a sandbox. The user must have <u>list sandbox permission</u> (p. 132) to view the sandboxes to suspend them.

See also Chapter 22, <u>Sandboxes - Server Side Job Files</u> (p. 145).

Reset caches

Deprecated.

• Running jobs unlimited

If the graph is to be run from server web interface, the user must have the <u>list sandbox permission</u> (p. 132) to list the graphs.

• Running jobs limited

If the graph is to be run from server web interface, the user must have the <u>list sandbox permission</u>(p. 132) to list the graphs.

• Configuration

Allows the user to access the configuration section.

• Users

This permission allow user to access the Users section and configure user accounts.

• List user

Allows the user to list users and access to the Users administration section (Configuration \rightarrow Users)

• Change passwords

Allows the user to change his password and to change password of another user.

To see list of users, the user needs the list user permission (p. 139).

• Edit user

Allows the user to change group assignment.

To see the list of users, the user must have the list user permission (p. 139).

• Edit own profile and password

Allows the user to change his profile (first name, last name, email, and password).

The user can access her profile in main web console view under username, in upper right corner of the page.

• Delete user

Allows the user to disable a user.

The user must have the list user permission (p. 139) to list available users.

Create user

Allows the user to create a new user.

If the user is to be created in the Server web interface, the creating user must have the <u>list user</u> permission (p. 139) to list users to access this option.

• Groups assignment

Allows the user to assign users to groups.

The user must have the <u>edit user permission</u> (p. 139) to successfully finish the assignment of users to groups.

If the user is to be created in the Server web interface, the creating user must have the <u>list user</u> permission (p. 139) to list users to access this option.

• Groups

Allows the user to manage groups: user can list groups, create groups, delete groups, edit the group, assign users to the group, and change permissions of the group.

• List groups

Allows the user to list groups. This permission is necessary for use of other options from the Groups group.

• Create group

Allows the user to create a new user group.

If the user group is to be created in the Server web interface, the user must have the <u>list groups</u> permission (p. 140) to view a list of groups and to access this option.

• Delete group

Allows the user to delete a user group.

Only empty groups can be deleted. You need to have the <u>list groups permission</u> (p. 140) to view list of groups and to access this option.

• Edit group

This permission allow user to edit user groups.

This permission does not include User assignment and Permission assignment.

If the user group is to be edited from server web interface, the user must have the <u>list groups</u> permission (p. 140).

• Users assignment

Allows the user to assign users to groups.

The user needs Edit group permission (p. 140) to commit the changes in the assignment.

If the assignment is to be edited in the Server web interface, the user must have the <u>list groups</u> <u>permission</u> (p. 140) to list the groups.

• Permission assignment

Allows the user to configure group **Permissions**.

The user needs have the Edit group permission (p. 140) to commit the changes.

If the permissions are to be edited in the Server web interface, the user must have the <u>list groups</u> permission (p. 140) to list the groups.

• Secure parameters administration

• Secure params

Allows the user to change the value of a secure parameter.

The user can use secure parameters in graphs even without this permission.

• CloverDX/System info sections

Allows the user to view System Info and CloverDX Info sections.

• CloverDX Server properties

Allows the user to view Server Properties tab and Data Profiler properties tab in CloverDX Info section.

The user must have the <u>CloverDX/System info sections permission</u> (p. 141) to access **CloverDX Info** section.

• Reload license

Allows the user to reload and view the server license.

The user must have the <u>CloverDX/System info sections permission</u>(p. 141) to access the **Configuration** section.

• Upload license

Allows the user to update the server license.

The user must have the <u>CloverDX/System info sections permission</u>(p. 141) to access the **Configuration** section.

See Activation (p. 32).

• Server Configuration Management

Allows the user to import and export the server configuration.

See Chapter 23, <u>Server Configuration Migration</u> (p. 156).

• Export Server Configuration

Allows the user to export the server configuration.

See Server Configuration Export (p. 157).

• Import Server Configuration

Allows the user to import the server configuration.

See Server Configuration Import (p. 158).

• Temp Space Management

Allows the user to access **Temp Space Management** section.

• Server Setup

Allows the user to access the server setup.

See Chapter 13, <u>Setup</u> (p. 54).

• Heap Memory Dump

Allows the user to create a Thread dump and a Heap Memory Dump.

See Chapter 25, Diagnostics (p. 163).

• Groovy Code API

Allows the user to run Groovy scripts.

• Open Profiler Reporting Console

Allows the user to login to the **Profiler reporting console**.

The permission is necessary to view the results of CloverDX Profiling Jobs in Designer.

Even without this permission, a user can create and run .cpj jobs from Designer.

User Lockout

CloverDX can lock out a user access after a set number of unsuccessful login attempts as a way of protecting against brute force attacks on users' credentials.

The lockout occurs only in **CloverDX**. For example, it will not affect LDAP in the case of LDAP user authentication. By default, the feature is disabled.

Information regarding user lockout is stored in the **USER_ACTION** server log. Notifications can be sent via email; however, it is necessary to set up a connection to an SMTP server in the E-mail (p. 59) tab of the **Setup GUI**.

The feature has several parameters which can be set by modifying the following lines in the configuration file; either directly or in the **Configuration File** tab of the Setup GUI (p. 54):

Table 21.4. User lockout parameters

Parameter	Description
security.lockout.login	Attenitpts number of login attempts of the user. The next failed login attempt will lock the user's access. When setting the value, keep in mind that CloverDX Designer with several server projects can attempt to log in multiple times. The recommended value is 50. Change the value to 0 to disable the feature.
security.lockout.reset	preprint sents the period (in seconds) during which failed login attempts are counted. If no such attempt occurs during this period, the counter of failed login attempts is reset to 0. This way, the user does not have to worry about accidentally locking himself out of the system after a certain number of failed login attempts over an extended period of time.
	The default value is 60 (1 minute). Change the value to 0 to set the period to infinity.
security.lockout.unloo	Rpprised nts the period (in seconds) after which a successful login attempt will unlock the previously locked user. After this period, the user is able to login using his credentials again without the need to have his account unlocked by the administrator. The parameter protects the system against denial of service (DoS) attacks and should be set to a reasonable value so you are not locked out of the system for too long in case the administrator's account is affected by the attack.
	The default value is 300 (5 minutes). Change the value to 0 to set the period to infinity.
security.lockout.notif	CENCE paramidter represents a comma separated list of emails of persons who should be notified when a user lockout occurs. Note that the locked out user receives the notification email automatically (if the server's SMTP is configured and they have provided their email address). This parameter should therefore be set, for example, to an administrators' mail group so they are aware of the situation.

The recommended, default values are set in such a way as to efficiently protect the system against brute force attacks, prevent complete lockout of the administrator access and not limit users in standard usage of **CloverDX Server**.

The properties can be set in the following section of the properties file:

Uncomment lines bellow to enable user lockout after number of failed logins

Number of failed login attempts after which a next failed login attempt will lock the user

0 means feature is switched off

default suggested value is 50

#security.lockout.login.attempts=50

Periods are specified in seconds

Period of time during which the failed login attempts are counted

Default is 60s (1 min)

#security.lockout.reset.period=60
Period of time after which a successful login attempt will unlock previously locked user
Default is 300s (5 min)
#security.lockout.unlock.period=300
Comma separated list of emails which will be notified when user is locked out.
#security.lockout.notification.email=

Unlocking User

Once the user's access is locked, you can see the status in the Users tab of the Configuration section.

To unlock the user, click on the ... button in the respective row of the Action column and choose Unlock.

Chapter 22. Sandboxes - Server Side Job Files

A sandbox is a place where you store all your project's transformation graph files, jobflows, data, and other resources. It's a server side analogy to a Designer project. The Server adds additional features to sandboxes, like user permissions management and global per-sandbox configuration options.

The Server and the Designer are integrated so that you are able to connect to a Server sandbox using a **Server Project** in your Designer workspace. Such a project works like a remote file system – all data is stored on the Server and accessed remotely. Nonetheless, you can do everything with Server Projects the same way as with local projects – copy and paste files, create, edit, and debug graphs, etcetera. See the **CloverDX Designer** manual for details on configuring a connection to the Server.

Technically, a sandbox is a dedicated directory on the Server host file system and its contents are managed by the Server. Advanced types of sandboxes, like "partitioned sandbox" have multiple locations to allow distributed parallel processing (more about that in Chapter 41, <u>Sandboxes in Cluster(p. 277)</u>). A sandbox cannot contain another sandbox within – it's a single root path for a project.

It is recommended to put all sandboxes in a folder outside the **CloverDX Server** installation (by default the sandboxes would be stored in the $\{user.data.home\}/CloverDX/sandboxes$, where the user.data.home is automatically detected user home directory). However, each sandbox can be located on the file system independently of the others if needed. The containing folder and all its contents must have read/ write permission for the user under which the **CloverDX Server** is running.

CloverDX 5.1.0.14	i							✓ Page Loaded	2018-12-17 09:45 +01:00		θ
Monitoring	New Sandbox	€ 8	Overview	File Content	Classpath	Config Properties	Graph structured info				
🕰 Task History	BasicExamples		🥷 graph/0	3enerateRepor	t.grf					•	×
Schedules	BiguataExamples Conn Conn	- 1	File name File size			GenerateReport.grf 3.3 KB (3,290 B)					
渝 Data Services 🗸	data-an data-out data-out	- 1	File path in	sandbox		graph/GenerateRepo	rt.grf				
Event Listeners	i data-Imp	- 1	File type	ad		ETL Graph 2018-12-14 09:03:07	IL Graph 018-12-14 09-03-07				
Sandboxes	subgraph	- 1	Sandbox co	ıde		BigDataExamples					
🖉 Configuration 🗸	BigDataExample_HIVE.grf R CheckParameters-Hive.grf	- 1	Sandbox ro	ot path		\${sandboxes.home}/E	BigDataExamples				
C Help	R CheckParameters-MapReduce.grf R CloverDX-CountVisits.grf	- 1	CONFIG PRO	PERTIES							
	RenerateReport.grf Hadoophive.CountVisits.grf Hadoophive.CountVisits.grf Hadoophive.CountVisits.grf Hadoophive.CountVisits.grf		NAME No Config	g Properties.		VALUE					
			PROPERTIES	S INHERITED FROM	THE SANDBOX						
	 MongoDB-CountVisits.grf PrepareInputData.grf 	- 1	NAME No prope	rties inherited fror	n the Sandbox	VALUE					
	PrepareinputData_MongoDB.grt Diplfow										
	i lookup										
	i profile										
	i seq ▶ in sec										
<	irans classpath										

Figure 22.1. Sandboxes Section in CloverDX Server Web GUI

Each sandbox in non-cluster environment is defined by the following attributes:

Name	Description
Sandbox	A sandbox name used just for display. It is specified by the user during sandbox creation and it can be modified later.
Sandbox ID	A unique name of the sandbox. It is used in server APIs to identify sandbox. It must meet common rules for identifiers. It is specified by user in during sandbox creation and it can be modified later. <i>Note: modifying is not recommended, because it may be already used by some APIs clients.</i>
Owner	It is set automatically during sandbox creation. It may be modified later.
Sandbox root path	Absolute server side file system path to sandbox root. It is specified by user during sandbox creation and it can be modified later. Instead of the absolute path, it's recommended to use \${sandboxes.home} placeholder, which may be configurable in the CloverDX Server configuration. So e.g. for the sandbox with ID "dataReports" the specified value of the "root path" would be \${sandboxes.home}/dataReports. Default value of sandboxes.home config property is \${user.data.home}/CloverDX/sandboxes where the user.data.home is configuration property specifying home directory of the user running JVM process - it's OS dependent). Thus on the unix-like OS, the fully resolved sandbox root path may be: / home/clover/CloverDX/sandboxes root path in Cluster environment.
Sandbox type	Type of the sandbox. It can be: local (p. 277), shared (p. 277) or partitioned (p. 278).

Referencing Files from the Graph or Jobflow

In some components you can specify file URL attribute as a reference to some resource on the file system. Also external metadata, lookup or DB connection definition is specified as reference to some file on the filesystem. With **CloverDX Server** there are more ways how to specify this relation.

• Relative path

All relative paths in your graphs are considered as relative paths to the root of the same sandbox which contains job file (graph or Jobflow).

• sandbox:// URLs

Sandbox URL allows user to reference the resource from different sandboxes with standalone **CloverDX Server** or the cluster. In cluster environment, **CloverDX Server** transparently manages remote streaming if the resource is accessible only on some specific cluster node.

For details about the sandbox URLs, see Using a Sandbox Resource as a Component Data Source (p. 279).

Sandbox Content Security and Permissions

Each sandbox has its owner who is set during sandbox creation. This user has unlimited privileges to this sandbox as well as administrators. Another users may have access according to sandbox settings.

Overview	Permissions	Upload ZIP	Classpath	Config Properties								
/ BigDa	ataExamples								P	Sal P	:	×
USER G	ROUP			READ	WRITE	EXECUTE	PROFILER READ	PROFILER ADMIN				
all user	S			✓	*	✓				Rem	ove	
	Add permission:	-										
									Updat	e perm	issior	IS

Figure 22.2. Sandbox Permissions in CloverDX Server Web GUI

Permissions to a specific sandbox are modifiable in **Permissions** tab in sandbox detail. In this tab, selected user groups may be allowed to perform particular operations.

There are the following types of operations:

Table 22.2. San	dbox permissions
-----------------	------------------

Name	Description
Read	Users can see this sandbox in their sandboxes list.
Write	Users can modify files in the sandbox through CS APIs.
Execute	Users can execute jobs in this sandbox.
	Note: job executed by graph event listener and similar features is actually executed by the same user as job which is the source of the event. See details in graph event listener . Job executed by schedule trigger is actually executed by the schedule owner. See details in Chapter 30, <u>Scheduling</u> (p. 193). If the job needs any files from the sandbox (e.g. metadata), the user also must have read permission, otherwise the execution fails.
Profiler Read	User can view results of profiler jobs executed from the sandbox.
Profiler Admin	User can administer results of profiler jobs executed from the sandbox.

Note that these permissions modify the access to the content of specific sandboxes. In addition, it is possible to configure permissions to perform operations with sandbox configuration (e.g. create sandbox, edit sandbox, delete sandbox, etc). For details, see Chapter 21, <u>Users and Groups</u> (p. 124).

Sandbox Content and Options

Download sandbox as ZIP (p. 149) Upload ZIP to sandbox (p. 149) Download file in ZIP (p. 149) Download file HTTP API (p. 150) Delete Sandbox (p. 150) A sandbox should contain jobflows, graphs, metadata, external connection and all related files. Files, especially graph or jobflow files, are identified by a relative path from sandbox root. Thus you need two values to identify a specific job file: sandbox and path in sandbox. The path to the Jobflow or graph is often referred to as **Job file**.

Although the web GUI section **sandboxes** is not a file-manager, it offers some useful features for sandbox management:

CloverDX 5.1.0.14				✓ Page Loaded	2018-12-17 09:48 +01:00	,		0
Monitoring ~	• New Sandbox	8	Overview Permissions Upload ZIP Classpath Config Properties			B Z		v
🖏 Task History	 BasicExamples BigDataExamples 	Î	BigDataExamples Sandbox BigDataExamples				•	_
B Schedules	Conn Download sandbox as ZIP Download sandbox as ZIP Download sandbox as ZIP	1	Sandbox code BigDataExamples					
জি Data Services 👻	Gata-out Suspend at once		Owner clover Sandhov root nath S/sandhoves home\/BirDataEvamples					
Event Listeners	data-tmp + Create directory Delete sandbox		Sandbox type Shared					
Sandboxes	🗀 subgraph 🇶 BigDataExample_HDFS.grf	1	RESOLVED SANDBOX ROOT PATH					

Figure 22.3. GUI - Sandboxes context menu

Download sandbox as ZIP

Select a sandbox in the left panel, then the web GUI displays the **Download sandbox as ZIP** button in the tool bar on the right side.

Created ZIP contains all readable sandbox files in the same hierarchy as on the file system. You can use this ZIP file to upload files to the same sandbox, or another sandbox on a different Server instance.

Upload ZIP to sandbox

Select a sandbox in the left panel. You must have the **write** permission for the selected sandbox. Then select the **Upload ZIP** tab in the right panel. Upload of a ZIP is parametrized by couple of switches, which are described below. Click the + **Upload ZIP** button, to open a common file browser dialog. When you choose a ZIP file, it is immediately uploaded to the Server and a result message is displayed. Each row of the result message contains a description of one single file upload. Depending on selected options, the file may be skipped, updated, created or deleted.

Label	Description
Encoding of packed file names	File names which contain special characters (non ASCII) are encoded. In the drop-down list, you choose the right encoding, so filenames are decoded properly.
Overwrite existing files	If this checkbox is checked, the existing file is overwritten by a new one, if both of them are stored in the same path in the sandbox and both of them have the same name.
Delete folders and files missing in uploaded zip file	If this option is enabled, all files which are missing in uploaded ZIP file, but they exist in destination sandbox, will be deleted. This option might cause a loss of data, so the user must have the May delete files missing in uploaded ZIP (p. 132) permission to enable it.

Download file in ZIP

Select a file in the left pane, then the web GUI displays the **Download file as ZIP** button in the tool bar on the right side.

Created ZIP contains just the selected file. This feature is useful for large files (i.e. input or output file) which cannot be displayed directly in the web GUI, so the user can download it.

Download file HTTP API

It is possible to download/view the sandbox file accessing "download servlet" by simple HTTP GET request:

http://[host]:[port]/[Clover Context]/downloadFile?[Parameters]

The Server requires BASIC HTTP Authentication. Thus with Linux command line HTTP client "wget" it would look like this:

```
wget --user=clover --password=clover
http://localhost:8080/clover/downloadFile?sandbox=default\&file=data-out/data.dat
```

Please note, that ampersand character is escaped by back-slash. Otherwise it would be interpreted as commandline system operator, which forks processes.

URL Parameters

- sandbox Sandbox code. Mandatory parameter.
- file Path to the file relative from sandbox root. Mandatory parameter.
- zip If set to true, the file is returned as ZIP and the response content type is application/x-zipcompressed. By default it is false, so the response is the content of the file.

Delete Sandbox

You can delete a sandbox by selecting the sandbox and clicking the **Delete sandbox** button on the top of the right pane, or by right-clicking the sandbox in the tree pane on the left and selecting the **Delete sandbox** option.

After that, a confirmation dialog opens where you can choose to delete sandbox files on disk, as well.

Job Config Properties

Each graph or Jobflow may have a set of configuration properties, which are applied during the execution. Properties are editable in the web GUI section **Sandboxes**. Select the job file and go to the **Config properties** tab.

The same configuration properties are editable even for each sandbox. Values specified for a sandbox are applied for each job in the sandbox, but with a lower priority than configuration properties specified for the job.

If neither the sandbox nor the job have configuration properties specified, defaults from main Server configuration are applied. Global configuration properties related to Job configuration properties have the executor. prefix. For example, the server property executor.classpath is default for the Job configuration property classpath. (See Part III, "Configuration" (p. 47) for details)

In addition, it is possible to specify additional job parameters, which can be used as placeholders in job XML. Please keep in mind, that these placeholders are resolved during loading and parsing of the XML file, thus such a job cannot be pooled.

If you use a relative path, the path is relative to \${SANDBOX_ROOT}.

In a path definition, you can use system properties - e.g. f(java.io.tmpdir) - and some of server configuration properties: f(sandboxes.home), f(sandboxes.home.partitioned) and f(sandboxes.home.local).

Property name	Default value	Description
classloade	r_caching false	CloverDX creates new classloaders when necessary to load a class in runtime. For example, a Reformat component with a Java transformation has to create a new classloader to load the class. It is worth noting that classloaders for JDBC drivers are not re-created. Classloader cache is used to avoid PermGen out of memory errors (some JDBC drivers automatically register themself to DriverManager, which can cause the classloader cannot be released by garbage collector). This behavior can be inconvenient for example if you want to share POJO between components. For example, a Reformat component creates an object (from a jar file on runtime classpath) and stores it into a dictionary. Another Reformat component recover the object from the dictionary and attempts to cast the object to the expected class. ClassCastException is thrown due different classloaders used in the Reformat components. Using this flag you can force CloverServer to re-use classloader when possible.
classpath		List of paths or jar files which contain external classes used in the job file (transformations, generators, JMS processors). All specified resources will be added to the runtime classpath of the transformation job. All CloverDX Engine libraries and libraries on application-server's classpath are automatically on the classpath. Separator is specified by the Engine property DEFAULT_PATH_SEPARATOR_REGEX. The directory path must always end with a slash character "/", otherwise ClassLoader doesn't recognize it is a directory. Server always automatically adds trans subdirectory of job's sandbox, so it doesn't have to be added explicitly.
compile_c	lasspath	List of paths or jar files which contain external classes used in the job file (transformations, generators, JMS processors) and related libraries for their compilation. Please note, that libraries on application-server's classpath are not included automatically. Separator is specified by the Engine property DEFAULT_PATH_SEPARATOR_REGEX. The directory path must always end with a slash character "/", otherwise ClassLoader doesn't recognize it is a directory. Server always automatically adds a SANDBOX_ROOT/trans/

Table 22.4. Job configuration properties

Chapter 22. Sandboxes - Server Side Job Files

Property name	Default value	Description
		directory and all JARs in the SANDBOX_ROOT/lib/ directory, so they don't have to be added explicitly.
debug_mo	de false	If true, edges with debug enabled will store data into files in a debug directory. Without explicit setting, running of a graph from Designer with server integration would set the debug_mode to true. On the other hand, running of a graph from the server console sets the debug_mode to false.
delete_obs	olete_temp_files false	If true, the system will remove temporary files produced during previous finished runs of the respective job.
		This property is useful together with enabled debug mode ensuring that obsolete debug files from previous runs of a job are removed from temp space. This property is set to true by default when executing job using Designer-Server integration.
enqueue_e	xecutions false	Boolean value. If true, executions above max_running_concurrently are enqueued, if false, executions above max_running_concurrently fail.
jobflow_to	ken_tracking true	If false, token tracking in jobflow executions will be disabled.
locale	DEFAULT_LOC engine property	ALE Can be used to override the DEFAULT_LOCALE engine property.
log_level	INFO	Log4j log level for this graph executions. (ALL TRACE DEBUG INFO WARN ERROR FATAL). For lower levels (ALL, TRACE or DEBUG), root logger level must be set to lower level, as well. Root logger log level is INFO by default, thus a transformation run log does not contain more detail messages then INFO event if the job configuration parameter log_level is set properly. For details about log4j configuration, see Chapter 17, Logging (p. 104).
max_grap	h_instance_age 0	A time interval in milliseconds which specifies how long may a transformation instance last in server's cache. 0 means that the transformation is initialized and released for each execution. The transformation cannot be stored in the pool and reused in some cases (a transformation uses placeholders using dynamically specified parameters)
max_runn	ing_concurrently unlimited	The maximum number of concurrently running instances of this transformation. In cluster environment, the limit is per node.
password		This property is deprecated. Password for decoding of encoded DB connection passwords.
skip_chec	<_config default value is taken from engine property	Specifies whether check config must be performed before a transformation execution.
time_zone	DEFAULT_TIM engine property	E_ZONE Can be used to override the DEFAULT_TIME_ZONE engine property.
tracking_i	nterval 2000	Interval in milliseconds for sampling nodes status in a running transformation.
use_jmx	true	If true, job executor registers jmx mBean of a running transformation.
use_local_	context_url false	If true, the context URL of a running job will be a local file: URL. Otherwise, a sandbox: URL will be used.

Chapter 22. Sandboxes - Server Side Job Files

Property name	Default value	Description
verbose_n	iode true	If true, more descriptive logs of job runs are generated.
worker_ex	ecution true	Set to false to enforce execution in Server Core. Can be set per file or per sandbox.

Overview	File Content	Classpath	Config Properties	Graph structured info			
graph/(CreditCardFra	udDetection.	grf		0	6 19	: ×
NAME			VALUE				
verbose	e_mode		true			Î	
Boolean	value which specifies	whether to put mo	ore detail messages to job run	og.			
trackin	ng_interval		500			Î	
Interval in	n ms for sampling noo	des status in runnir	ng job.				
		•				0	
PROPERTIE	S INHERITED FROM	THE SANDBOX					
NAME			VALUE				
No prope	erties inherited fro	m the Sandbox.					
					🖬 Սբ	odate Ch	anges

Figure 22.4. Job config properties

WebDAV Access to Sandboxes

Since 3.1

WebDAV API allows you to access and manage sandbox content using a standard WebDAV specification.

Specifically, it allows for:

- Browsing a directory structure
- Editing files
- Removing files/folders
- Renaming files/folders
- Creating files/folders
- Copying files
- Moving files

The WebDAV interface is accessible from the URL: "http://[host]:[port]/clover/webdav".

Note: Although common browsers will open this URL, most of them are not rich WebDAV clients. Thus, you will only see a list of items, but you cannot browse the directory structure.

WebDAV Clients

There are many WebDAV clients for various operating systems, some OS support WebDAV natively.

Linux like OS

Great WebDAV client working on Linux systems is Konqueror. Please use different protocol in the URL: webdav://[host]:[port]/clover/webdav

Another WebDAV client is Nautilus. Use different protocol in the URL dav://[host]:[port]/clover/webdav.

MS windows

Last distributions of MS Windows (Win XP and later) have native support for WebDAV. Unfortunately, it is more or less unreliable, so it is recommended to use some free or commercial WebDAV client.

- The best WebDAV client we've tested is BitKinex: http://www.bitkinex.com/webdavclient
- Another option is to use Total Commander (http://www.ghisler.com/index.htm) with WebDAV plugin: http://www.ghisler.com/plugins.htm#filesys

Mac OS

Mac OS supports WebDAV natively and in this case it should be without any problems. You can use "finder" application, select "Connect to the server ..." menu item and use URL with HTTP protocol: "http://[host]:[port]/ clover/webdav".

WebDAV Authentication/Authorization

CloverDX Server WebDAV API uses the HTTP Basic Authentication by default. However it may be reconfigured to use HTTP Digest Authentication.

Digest Authentication may be useful, since some WebDAV clients can't work with HTTP Basic Authentication, only with Digest Authentication.

HTTP Digest Authentication is feature added to the version 3.1. If you upgraded your older **CloverDX Server** distribution, users created before the upgrade cannot use the HTTP Digest Authentication until they reset their passwords. So when they reset their passwords (or the admin does it for them), they can use Digest Authentication as well as new users.

The HTTP Digest Authentication is configured with security.digest_authentication.* configuration properties. To enable it, set security.digest_authentication.features_list to contain features that are listed in the security.digest_authentication.features_list. As items in security.basic_authentication.features_list have higher priority, you should empty it to allow HTTP Digest Authentication to be used.

See Chapter 15, <u>List of Configuration Properties</u> (p. 82) for details.

For details on authentication methods see https://tools.ietf.org/html/rfc7617 and https://tools.ietf.org/html/rfc2617

Chapter 23. Server Configuration Migration

CloverDX Server provides means to migrate its configuration (e.g. event listeners, schedules, etc.) or parts of the configuration between separate instances of the Server. A typical use case is deployment from test environment to production - this involves not only deployment of graphs, but also copying parts of the configuration, such as file event listeners, etc.

Configuration migration is performed in 2 steps - export of the configuration from the source Server, followed by import of the configuration at the destination Server. After exporting, the configuration is stored as an XML file. The file can be modified manually before import, for example to migrate only parts of the configuration. Additionally, the configuration file can be stored in a versioning system (such as Subversion or Git) for versioning of the **CloverDX Server** configuration.

It is recommended to perform import of configuration on a suspended **CloverDX Server** and to plan for maintenance. Additionally, it is recommended to backup the **CloverDX Server** configuration database before the import.

The following items are parts of the Server Configuration and can be migrated between servers:

- Users & Groups (p. 124)
- Sandboxes (p. 145)
- Job Parameters (p. 168)
- Schedules (p. 193)
- <u>Graph Event Listeners(p. 206)</u> Jobflow Event Listeners(p. 212) JMS Message Listeners(p. 214) File Event Listeners (remote and local) (p. 221)
- Temp Spaces (p. 117)

Permissions for Configuration Migration

Whether a user is entitled to perform configuration migration is determined by having *Server Configuration Management* permission; this permission has two sub-permissions: *Export Server Configuration* and *Import Server Configuration* (see Groups permissions (p. 131) for further information on permissions). These permissions are of higher priority than permissions related to a particular migrated item type - so even if the user does not have a permission e.g. to list server's schedules, with *Export Server Configuration* he will be allowed to export all of defined schedules. The same is true for adding and changing items with the *Import Server Configuration* permission.

See Server Configuration permission (p. 141).

Server Configuration Export

Export of Server configuration is performed in the **Configuration** > **Export** section. You can choose which items will be exported (see Figure 23.1 (p. 157)). Click the **Export Configuration** button to save an XML file with the configuration. The name of the XML file includes current timestamp.

If you manually edit the file, make sure that the content is valid. This can be done by validation against XSD schema. The schema for a configuration XML document can be found at http://[host]:[port]/[contextPath]/schemas/clover-server-config.xsd.

The XML file contains selected items of the **CloverDX Server** instance. The file can by modified before importing (p. 158) to another Server instance - for example to import schedules only.

If you want to automate the process of configuration export, use the HTTP API <u>Operation</u> export server config (p. 244).

Export Configuration	
Current server configuration can be exported as an XML document. The XML can be modified (e.g. to include only the data needed) and imported to another server instance.	
ITEMS TO EXPORT	
✓ Users ✓ Sandboxes	
✓ User Groups ✓ Job Configuration Parameters	
✓ Schedules	
✓ Data Services ✓ Event Listeners	
✓ Temp Spaces	
Select All Deselect All	
Export Configuration	

Figure 23.1. Server Configuration Export screen

Server Configuration Import

This function merges exported configuration into the Server. The configuration is loaded from an XML file which can be created by the <u>Server Configuration Export (p. 157)</u> function, or automatically generated. The **Import** function is located in **Configuration** > **Import**.

If you want to automate the process of configuration import, use the HTTP API <u>Operation</u> import server config (p. 245).



Figure 23.2. Server Configuration Import screen

The XML file defines configuration items to be imported. The items are matched against current configuration of the destination Server. Depending on result, the items are either added to the destination Server or existing item are updated. Matching of items is based on a key that depends on the item type:

	Table 23.1.	Codes	for matching	items in	configuration	import
--	-------------	-------	--------------	----------	---------------	--------

Item	Code
users	user code
user groups	group code
sandboxes	sandbox code
job parameters	triplet: job parameter name, sandbox code, job file
event listeners	event listener name
schedule	schedule description
launch service	triplet: service name, server user, service user group
data service	pair: sandbox, rjob
temp spaces	pair: temp space node ID, temp space path

Configuration Import Process

Uploading Configuration

Select the XML file with exported configuration. As the first step, the Server executes a safe *dry run* import, without actually changing your Server's configuration. Once uploaded, the Server checks the validity of the configuration and displays a dry run log containing added/updated items or errors.

You will be notified if the source and target differ in at least minor version number (e.g. 4.8.1 and 4.9.0). Importing configuration from different version may generate more warnings which will require your attention.

Verifying Configuration

There are three main states that can occur in a dry run:

- 1. **The configuration is valid:** no errors have occurred and the configuration can be committed.
- 2. A The dry run ended with warning(s): the configuration can be committed, but the warning(s) should be resolved after import.
- 3. **•** The dry run ended with error(s): the configuration cannot be committed until the error(s) are fixed. Consult the dry run log to fix the errors in the XML file and re-upload it.

Dry run log

The **Dry run log** displays changes in the configuration, warns the user about items in configuration that require their attention and notifies them about errors that must be fixed before commit. Changes in the configuration are displayed in a diff view; items before update have a red background while added items or items after update have a green background with changes highlighted:

Below is an example of a log entry indicating a change in the configuration where a user smithj is added to a previously empty group configuration. The diff view shows the change in two steps (two lines grouped together with no space between them) as 'replacing' the empty group (•) with the same group with the new user (•).



Figure 23.3. Updated User Groups - User added to a group

Committing Import

Once all errors are resolved and the configuration is valid, you can **Commit Changes**. After confirmation, **Log of committed changes** will display the results. The log can be downloaded, as well.

Some items may not be initialized properly after the import (e.g. their initialization requires presence of a cluster node that went down in the meantime or someone made changes on the Server between dry run and committing). User is notified about these problems in **Log of committed changes** with link to the affected items. You should check such items in respective sections of the Server console and change their settings to fix the issue or remove them.

Chapter 24. Upgrading Server to Newer Version

General Notes on Upgrade

- An upgrade of CloverDX Server requires down time; plan a maintenance window.
- A successful upgrade requires about 30 minutes; rollback requires 30 minutes.
- Perform the steps below in a development/testing environment first before moving onto a production environment.

Upgrade Prerequisites

- Having a new **CloverDX Server** web application archive (clover.war appropriate for the application server used) & license files available.
- Having <u>release notes</u> for the particular **CloverDX** version available (and all versions between current and intended version to be upgraded to).
- Having the graphs and jobs updated and tested with regards to known issues & compatibility for the particular **CloverDX** version.
- Having the **CloverDX Server** configuration properties file externalized from default location, see Chapter 12, <u>Configuration Sources</u> (p. 50).
- Standalone database schema where **CloverDX Server** stores configuration, see Chapter 14, <u>System Database</u> <u>Configuration</u> (p. 63).
- Having a separate sandbox with a test graph that can be run at any time to verify that **CloverDX Server** runs correctly and allows for running jobs.

Upgrade Instructions

- 1. Suspend all sandboxes, wait for running graphs to finish processing.
- 2. Shutdown the CloverDX Server application (or all servers, if they run in a cluster mode).
- 3. Backup the existing **CloverDX** database schema (if any changes to the database schema are necessary, the new server will automatically make them when you start it for the first time).
- 4. Backup the existing **CloverDX** web application archive (clover.war) & license files (on all nodes).
- 5. Backup the existing CloverDX sandboxes (on all nodes).
- Re-deploy the CloverDX Server web application. Instructions how to do that are application server dependent

 see <u>Production Server</u> (p. 15) for installation details on all supported application servers. After the re-deployment, your new server will be configured based on the previous version's configuration.
- 7. Replace old license files with the valid one (or you can later use the web GUI form to upload new license). The license file is shipped as a text containing a unique set of characters. If you:
 - received the new license as a file (*.dat), then simply use it as new license file.
 - have been sent the license text, e.g. inside an email, then copy the license contents (i.e. all text between Company and END LICENSE) into a new file called clover-license.dat. Next, overwrite the old license file with the new one or upload it in the web GUI.

For details on license installation, see Activation (p. 32).

8. • Standalone Server

Start the CloverDX Server application.

• Cluster

- a. Start CloverDX Server on the first node;
- b. Wait until CloverDX Server is accessible (i.e. you can log in);
- c. Start CloverDX Server on other nodes.

To prevent database corruption, do not start all Cluster nodes simultaneously.

- 9. Review that contents of all tabs in the **CloverDX Server** Console, especially scheduling and event listeners looks OK.
- 10.Update graphs to be compatible with the particular version of **CloverDX Server** (this should be prepared and tested in advance).
- 11.Resume the test sandbox and run a test graph to verify functionality.
- 12.Resume all sandboxes.

Upgrade from 4.8.x or earlier to 4.9.x or later



Significant architectural change in version 4.9.0

By default since 4.9.0, jobs (graphs, jobflow, data services) are executed in a standalone JVM called Worker (p. 5).

To run jobs in the Server Core (i.e. in the same way as in earlier versions of **CloverDX Server**), you can disable execution in Worker for particular jobs or sandboxes (p. 153) or disable Worker completely (p. 90).

Note that Launch Services (deprecated) and Profiler are only available in the Server Core. These features require no additional configuration.

Configuration Changes

Worker is the executor of jobs, all jobs run in the Worker by default. It runs in a separate process (JVM), so it requires configuration in addition to the Server Core.

The Worker's configuration relates to memory size, classpath, command line options and JNDI. For an overview of Worker related configuration, see Introduction to Worker configuration (p. 5). The introduction provides an overview of the new Worker specific configuration with links to details.

With Worker, new configuration properties (p. 90) were introduced to set up various parameters (timeouts, heap size, etc.). These properties can be added to the properties file (p. 56).

You should split the main memory used by the Server between the Server Core and Worker. Generally, Worker would require more memory than the Server Core as it runs graphs.

The command line arguments and parameters, you were used to add to the Server Core command line in earlier versions, should be added to the Worker's command line, too.

Libraries added to the Server Core classpath should be added to the classpath of Worker. The place to copy these libraries is the fclover.home/worker-lib directory.

Changes to Jobs

If your jobs (DX graphs, jobflows, etc.) use JNDI for database or JMS connections, you need to configure JNDI on Worker (see JNDI in Worker (p. 92)). You might also need to update your jobs to use the new JNDI resources

configured in Worker - they might be available on new JNDI paths. If you do not need these JNDI resources on the Server Core anymore, consider removing them from the Server Core.

Rollback Instructions

- 1. Shutdown the **CloverDX Server** application.
- 2. Restore the CloverDX Server web application (clover.war) & license files (on all nodes).
- 3. Restore the CloverDX Server database schema.
- 4. Restore the CloverDX sandboxes (on all nodes).
- 5. Start the CloverDX Server application (on all nodes).
- 6. Resume the test sandbox and run a test graph to verify functionality.
- 7. Resume all sandboxes.



Important

Evaluation Version - a mere upgrade of your license is not sufficient. When moving from an evaluation to production server, you should not use the default configuration and database. Instead, take some time to configure **CloverDX Server** so that it best fits your production environment.

Chapter 25. Diagnostics

CloverDX Server allows you to create a thread dump or a heap dump for both Server Core and Worker. The heap and thread dumps are useful for investigation of performance and memory issues.

A **heap dump** is a content of the JVM process memory stored in a binary file. The generated heap dump file can be investigated with tools like jvisualvm or jhat.

A **thread dump** is a list of existing JVM threads with their callstacks and held locking objects (if supported). It can be viewed in a text editor.

Note that generating a heap or thread dump requires the <u>Heap Memory Dump permission</u> (p. 142).

Downloading Heap/Thread Dump

In Server GUI, go to **Configuration** \rightarrow **System Info** \rightarrow **Diagnostics**.

To download a heap dump or thread dump of either Server Core or Worker, click on the respective link. Note that in case of a heap dump, it may take a moment before the download is ready.



Important

The size of the heap dump temp file is determined by the Server Core or Worker <u>heap memory</u> <u>size setting</u> (p. 37) - make sure there is enough free space in the <u>temp directory</u> (p. 117) before downloading the file.

The temp file is automatically deleted after download, but note that it may contain sensitive information (e.g. passwords) in plain text.

A garbage collection is triggered before the heap dump which may cause the Server to briefly stop responding.

The Dump live objects only checkbox allows you to avoid dumping of objects awaiting garbage collection.

Downloading Heap/Thread Dump Using jcmd Command

Heap Dump

The heap dump of Worker can be created with jcmd command: jcmd <pidOfWorker> GC.heap_dump <filename> You should specify the file name with full path to avoid searching for the file as jcmd does not always create it in the working directory.

Thread Dump

The thread dump of Worker can be created with jcmd command: jcmd <pidOfWorker> Thread.print

See details on jcmd.

Generating Heap Dump on Out of Memory Errors

To generate a heap dump on Out of Memory errors, add -XX:+HeapDumpOnOutOfMemoryError to the worker.jvmOptions (p. 91) property. A dump file java_pid.hprof will be generated when an Out of Memory error occurs. The heap dump will be located in the working directory of the Worker's process (same as the Server Core's working directory). You can override the directory location with the -XX:HeapDumpPath=/disk2/ dumps option. Important: the generated file can be large, its size is equal to the heap size.

Enabling GC Logging

Some memory and performance issues can be investigated with help of garbage collection logs.

Core

To enable the logging of the Server Core, add -verbose:gc -XX:+PrintGCTimeStamps -Xloggc:server_core_gc_log.txttoJAVA_OPTS in \$CATALINA_HOME/bin/setenv.sh. Restart of the Server is required to reflect the configuration change.

Worker

To enable the garbage collection logging in Worker, add the flags -verbose:gc -XX: +PrintGCTimeStamps -Xloggc:worker_gc_log.txt to the Worker's JVM arguments (p. 58) field in the Setup GUI, or the worker.jvmOptions (p. 91) property.

The -Xloggc option sets the path for the detailed garbage collector log. For the GC logging of Worker, use a different file name than for the Server. You can analyze the log file with various tools, e.g. http://gceasy.io/.

Restart of Worker is required.

More Details

```
Another useful garbage collector flags are -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:+PrintGCCause -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 - XX:GCLogFileSize=5M -XX:+PrintTenuringDistribution
```

For details on JVM flags, see Oracle's Java HotSpot VM Options.

Additional Diagnostic Tools

Below are additional useful diagnostic tools:

Investigating usage of direct memory

To investigate usage of direct memory, add -XX:NativeMemoryTracking=summary the worker.jvmOptions (p. 91) property. The details on native memory usage can be displayed with **jcmd <pid>VM.native_memory summary**. For more information, see <u>Native Memory Tracking tool</u>.

Enable Remote JMX Monitoring

Add the following options to the worker.jvmOptions (p. 91) property:

-Dcom.sun.management.jmxremote=true	-
Dcom.sun.management.jmxremote.port=8687	-
Dcom.sun.management.jmxremote.authenticate=false	-
Dcom.sun.management.jmxremote.ssl=false	_
Djava.rmi.server.hostname=example.com	

With the above options, you enable remote connection to JMX monitoring, which provides a wide range of information about the running JVM, JNDI resources etc. Change the value of java.rmi.server.hostname to the hostname of your Server.

Chapter 26. Troubleshooting Worker

Worker Logs

When investigating issues with Worker itself or jobs running in Worker, there are several logs with useful information:

Logs found in the **Monitoring** \rightarrow **Server Logs** section of the Server Console.

• **COMMON** log - the main Server log contains also information related to Worker. This log contains the full command line used to start Worker, this allows you to check the command line arguments. Additionally, standard output of the Worker process is redirected to this log - this is useful especially if the Worker process crashes during startup.

The COMMON log file is located in \${java.io.tmpdir}/cloverlogs/all.log

• **WORKER** log - the main Worker log provides information about Worker startup, initialization, executed jobs, runtime activities, etc. The initialization details contain information about Worker's JNDI resources, etc.

The WORKER log file is located in \${java.io.tmpdir}/cloverlogs/worker_[nodeID].log

You can also open this log via the Go to logs action in the Worker section of the Monitoring page.

Worker Command Line

The full command line that was used to start the Worker process can be found in:

- Monitoring section, use the **Show command line** action on Worker. For more details, see <u>Showing Worker's</u> <u>Command Line Arguments</u> (p. 116).
- the COMMON log of the Server (found in the **Monitoring** →**Server Logs** page). See section above for more details.

Investigate the command line options in case Worker does not correctly start or if the configuration of the running Worker is not correct.

Worker Does Not Start

If Worker does not start, check the following:

- Server's COMMON log and the WORKER log, see above (p. 165). Look for errors during Worker startup and initialization.
- Worker's command line arguments, see above (p. 165). Look for invalid command line arguments. Additionally, check the custom JVM arguments set on Worker, in the Worker (p. 58) tab of Setup (p. 54) or via the worker.jvmOptions (p. 91) configuration property.

Restarting Worker

In case Worker gets into an unrecoverable state (e.g. out of heap memory, etc.) and you fix the source issue, you can restart it from the Monitoring section (For more details, see <u>Restarting the Worker</u> (p. 116).):

- restart immediately, which will abort jobs currently running in Worker;
- restart after running jobs finish, in case the currently running jobs are crucial.

Worker does not start

In case Worker does not start (i.e. remains in the STARTING status), check the COMMON log first.

Worker Crashes

If common log (all.log file) contains row similar to the following one, the worker crashed due to exhausted heap space.

2018-03-22 16:08:29,008[s StdOut reader] WorkerProcess INFO [worker0@N1:10500]: java.lang.OutOfMemoryError: Java heap

You can configure it to generate heap dump for further investigation. To do so, add -XX: +HeapDumpOnOutOfMemoryError to JVM arguments in Worker configuration (**Configuration** > **Setup** > **Worker**). The generated file can be investigated with tools like jvisualvm or jhat.

Another cause of the crash can be swapped Worker's main memory. If there is insufficient free space in the main memory and the pages of Worker process are swapped on the hard drive, Worker is slowed down, it does not receive heart beat from the Server in time and it kills itself. Lower the maximum heap size of Worker to avoid swapping. Make note java process uses also non-heap memory, e.g. metaspace or direct memory.

To investigate usage of direct memory, add -XX:NativeMemoryTracking=summary to Worker's JVM arguments (**Configuration** > **Setup** > **Worker**). The details on native memory usage can be displayed with jcmd <pid> VM.native_memory summary. See https://docs.oracle.com/javase/8/docs/technotes/guides/troubleshoot/tooldescr007.html for details on native memory tracking.

Worker Hangs

Usually, it is caused by garbage collector. Try tweaking the garbage collection.

Another cause can be swapping of worker process pages on hard drive.

Issues with Classloading

To debug issues with classloading, add -verbose: class to **JVM arguments** of Worker. Loaded and unloaded classes will be printed to the output. The output can be seen in the common log.

Part V. Using Graphs

Chapter 27. Graph/Jobflow Parameters

The CloverDX Server passes a set of parameters to each graph or jobflow execution.

Keep in mind that $\{paramName\}$ placeholders (parameters) are resolved only during the initialization (loading of XML definition file), so if you need the parameters to be resolved for each job execution, you cannot set the job to be pooled. However, current parameter values are always accessible by an inline Java code:

String runId = getGraph().>getGraphProperties().getProperty("RUN_ID");

Properties may be added or replaced:

```
getGraph().getGraphProperties().setProperty("new_property", value);
```

This is a set of parameters which are always set by **CloverDX Server** (for more information, see <u>Job Config</u> <u>Properties</u> (p. 151)):

Table 27.1. Defaults for graph execution configuration

Key	Description
SANDBO	X_CODE An identifier of a sandbox which contains the executed graph.
JOB_FILE	A path to the file (graph, subgraph, jobflow). The path relative to the sandbox root path.
SANDBO	X_ROOT An absolute path to sandbox root.
RUN_ID	An ID of the graph execution. In a standalone mode or cluster mode, it is always unique. It may be lower than 0 value, if the run record isn't persistent. For more information, see Chapter 39, <u>Launch</u> <u>Services</u> (p. 250).
PARENT_	RUN_ID A run ID of the graph execution which is a parent to the current one. Useful when the execution is a subgraph, child-job of some jobflow or worker for distributed transformation in a cluster. When the execution doesn't have a parent, the PARENT_RUN_ID is the same as RUN_ID.
ROOT_RI	JN_ID A run ID of the graph execution which is a root execution to the current one (the one which doesn't have a parent). Useful when the execution is a subgraph, child-job of some jobflow or worker for distributed transformation in a cluster. When the execution doesn't have a parent, the ROOT_RUN_ID is the same as RUN_ID.
CLOVER.	USERNAME The username of the user who launched the graph or jobflow.
NODE_ID	An ID of the node running the graph or jobflow.

Parameters by Execution Type

Additional parameters are passed to the graph depending on how the graph is executed:

Executed from Web GUI (p. 169) Executed by Launch Service Invocation (p. 169) Executed by HTTP API Run Graph Operation Invocation (p. 169) Executed by RunGraph Component (p. 169) Executed by WS API Method executeGraph Invocation (p. 169) Executed by Task Graph Execution by Scheduler (p. 169) Executed from JMS Listener (p. 169) Executed by Task Start a graph by Graph/Jobflow Event Listener (p. 170) Executed by Task Graph Execution by File Event Listener (p. 170)

Executed from Web GUI

Graphs executed from the web GUI have no additional parameters.

Executed by Launch Service Invocation

Service parameters which have the **Pass to graph** attribute enabled are passed to the graph not only as "dictionary" input data, but also as a graph parameter.

Executed by HTTP API Run Graph Operation Invocation

Any URL parameter with a param_ prefix is passed to an executed graph but without the param_ prefix, i.e. param_FILE_NAME specified in a URL is passed to the graph as a property named FILE_NAME.

Executed by RunGraph Component

Since 3.0, only parameters specified by the **paramsToPass** attribute are passed from the parent graph to the executed graph. However common properties (RUN_ID, PROJECT_DIR, etc.) are overwritten with new values.

Executed by WS API Method executeGraph Invocation

Parameters with values may be passed to the graph with a request for execution.

Executed by Task Graph Execution by Scheduler

Table 27.2. passed parameters

Key	Description
EVENT_SCHEDULI	EVENT_TYPE
EVENT_SCHEDULI	E_LAST_EVENT
EVENT_SCHEDULI	E_DESCRIPTION A schedule description, which is displayed in the web GUI
EVENT_USERNAM	E The owner of the event. For schedule it is the user who created the schedule.
EVENT_SCHEDULI	E_ID An ID of the schedule which triggered the graph

Executed from JMS Listener

There are many graph parameters and dictionary entries passed, depending on the type of incoming message. See details in <u>JMS Message Listeners</u> (p. 214).

Executed by Task Start a Graph by Graph/Jobflow Event Listener

Since 3.0, only *specified* properties from a "source" job are passed to the executed job, by default. This behavior can be changed by the graph.pass_event_params_to_graph_in_old_style Server config property so that *all* parameters from a "source" job are passed to the executed job. This switch is implemented for backwards compatibility. With the default behavior, in the editor of graph event listener, you can specify a list of parameters to pass. For more information, see <u>Start a Graph</u> (p. 179).

The following parameters with current values are always passed to the target job

Table 27.3. passed parameters

Key	Description
EVENT_RUN_SANI	DBOX
	A sandbox with the graph which is the source of the event
EVENT_JOB_EVEN	T_TYPE
	GRAPH_STARTED GRAPH_FINISHED GRAPH_ERROR GRAPH_ABORTED GRAPH_TIMEOUT GRAPH_STATUS_UNKNOWN, analogically JOBFLOW_* for jobflow event listeners.
EVENT_RUN_JOB_	FILE
	A jobFile of the job which is the source of the event
EVENT_RUN_ID	An ID of the graph execution which is the source of the event.
EVENT_TIMEOUT	A number of milliseconds which specifies an interval of timeout. Useful only for "timeout" graph event.
EVENT_RUN_RESU	ЛТ
	A result (or current status) of the execution which is the source of the event.
EVENT_USERNAM	E
	The owner of the event. For graph events it is the user who created the graph event
	listener

Executed by Task Graph Execution by File Event Listener

Table 27.4. passed parameters

Key	Description
EVENT_FILE_PATH	A path to the file which is the source of the event. Does not contain a file name. Does not end with a file separator. Is passed only for the local file event listener.
EVENT_FILE_NAM	E A filename of the file which is the source of the event. Is passed only when the
EVENT_FILE_URLS	Contains string, which may be used "as is" in the file URL attribute of various CloverDX components. It may contain a URL to one or more (if grouping is enabled) files. It may contain local path(s) or remote URL(s) where credentials are replaced by placeholders (due to security reasons).
EVENT_FILE_AUT	I_USERNAME A username/ID to the remote location.
EVENT_FILE_AUT	I_USERNAME_URL_ENCODED The same as EVENT_FILE_AUTH_USERNAME, but the value is also URL encoded, so it may be used in the URL.
EVENT_FILE_AUT	H_PASSWORD Password/key to the remote location. It's encrypted by the master password. It is passed only when the file listener uses user+password authentication.

Chapter 27. Graph/ Jobflow Parameters

Key	Description
EVENT_FILE_AUT	H_PASSWORD_URL_ENCODED The same as EVENT_FILE_AUTH_PASSWORD, but the value is also URL encoded, so it may be used in the URL (EVENT_FILE_URLS parameter).
EVENT_FILE_EVEN	NT_TYPE SIZE CHANGE_TIME APPEARANCE DISAPPEARANCE
EVENT_FILE_PATT	ERN A pattern specified in a file event listener
EVENT_FILE_LIST	ENER_ID
EVENT_USERNAM	E Owner of the event. For file events, it is the user who created the file event listener.

Adding Another Graph Parameters

Additional "Graph Config Parameters"

It is possible to add so-called additional parameters in the web GUI (Chapter 22, <u>Sandboxes - Server Side</u> <u>Job Files</u> (p. 145)) for the selected graph or for all graphs in the selected sandbox. See details in <u>Job Config</u> <u>Properties</u> (p. 151).

Task "execute_graph" Parameters

The **execute graph** task may be triggered by a schedule, graph event listener, or file event listener. The task editor allows you to specify key=value pairs which are passed to the executed graph.

Chapter 28. Tasks

A task is a graph, jobflow, Groovy script, etc. that can be started manually, started on scheduled time, or triggered by some event. A task specifies WHAT to do.

There are several tasks implemented for a schedule and graph event listener as follows:

- <u>Start a Graph</u> (p. 179)
- <u>Start a Jobflow</u> (p. 182)
- <u>Abort job</u> (p. 185)
- Execute Shell Command (p. 176)
- <u>Send an Email</u> (p. 173)
- Execute Groovy Code (p. 190)
- <u>Archive Records</u> (p. 186)

Tasks in Cluster Environment

In Cluster environment, you can specify a node where the task runs. The task can run on **Any node** or on **one of selected nodes**. If there is no node ID specified, the task may be processed on any cluster node; so in most cases, it will be processed on the same node where the event was triggered. If there are some nodeIDs specified, task will be processed on the first node in the list which is connected in cluster and ready.

Tasks are used in

Chapter 30, <u>Scheduling</u> (p. 193) Chapter 32, <u>Listeners</u> (p. 204) Chapter 29, <u>Manual Task Execution</u> (p. 192)

Send an Email

The **send e-mail** task is useful for notifications about a result of graph execution. For example, you can create a listener with this task type to be notified about each failure in the specified sandbox or a failure of the particular graph.

This task is very useful, but for now only as a response for graph events. This feature is very powerful for monitoring. (for description of this task type, see <u>Graph Event Listeners</u> (p. 206)).

Note: It seems useless to send emails periodically, but it may send current server status or daily summary. These features will be implemented in further versions.

Table 28.1. Attributes of "Send e-mail" task

Name	Description			
Task type	"Send an email"			
То	The recipient's email address. It is possible to specify more addresses separated by a comma. It is also possible to use placeholders. For more information, see <u>Placeholders</u> (p. 174).			
Cc	Cc stands for 'carbon copy'. A copy of the email will be delivered to these addresses. It is possible to specify more addresses separated by a comma. It is also possible to use placeholders. For more information, see <u>Placeholders</u> (p. 174).			
Bcc	Bcc: stands for 'Blind carbon copy'. It is similar as Cc, but the others recipients aren't aware, that these recipients received a copy of the email.			
Reply-to (Sender)	Email address of sender. It must be a valid address according to the SMTP server. It is also possible to use placeholders. For more information, see <u>Placeholders</u> (p. 174).			
Subject	An email subject. It is also possible to use placeholders. For more information, see <u>Placeholders</u> (p. 174).			
HTML	A body of the email in HTML. The email is created as multipart, so the HTML body should have a precedence. A plain text body is only for email clients which do not display HTML. It is also possible to use placeholders. For more information, see <u>Placeholders</u> (p. 174).			
Text	A body of the email in plain text. The email is created as multipart, so the HTML body should have a precedence. A plain text body is only for email clients which do not display HTML. It is also possible to use placeholders. For more information, see <u>Placeholders</u> (p. 174).			
Log file as attachmen	If this switch is checked, the email will have an attachment with a packed log file of the related graph execution.			
Sandbox	BasicExamples -			
----------------	---	--	--	--
Graph	graph/CreditCardFraudDetection.grf 🔹			
Event type	Graph finished 💌			
TRIGGERED TASK				
Task type	Send an email 🔹			
Email template	No template Job finished			
То	\${user.email}			
Subject	CloverDX Server notification - Graph run \${run.id} of \${run.jobFile} finish			
HTML	HTML PUBLIC "-//W3C//DTD HTML 4.0</td			
	<pre><html>chead></html></pre>			
	<pre><html><head></head></html></pre>			

Figure 28.1. Web GUI - send email

Note: Do not forget to configure a connection to an SMTP server (see Part III, "Configuration" (p. 47)).

Placeholders

Placeholder may be used in some fields of tasks. They are especially useful for email tasks, where you can generate the content of email according to context variables.

Note: In most cases, you can avoid this by using email templates (See E-mail task for details)

These fields are preprocessed by Apache Velocity templating engine. See the Velocity project URL for syntax description <u>http://velocity.apache.org/</u>.

There are several context variables, which you can use in placeholders and even for creating loops and conditions.

- event
- now
- user
- run
- sandbox

Some of them may be empty depending on the type of the event. For example, if a task is processed because of a graph event, then *run* and *sandbox* variables contain related data, otherwise they are empty.

name	Contains
now	Current date-time
user	The user, who caused this event. It may be an owner of a schedule, or someone who executed a graph. It contains sub-properties which are accessible using dot notation (i.e. \${user.email}) email:
	 user.email user.username user.firstName user.lastName user.groups (list of values)
run	A data structure describing one single graph execution. It contains sub-properties which are accessible using dot notation (i.e. \${run.jobFile})
	 job.jobFile job.status job.startTime job.stopTime job.errNode job.errMessage job.errException job.logLocation
tracking	A data structure describing a status of components in a graph execution. It contains sub-properties which are accessible using the Velocity syntax for loops and conditions.
condhou	<pre>#if (\${tracking}) #foreach (\$phase in \$tracking.trackingPhases) tr>phase: \${phase.phaseNum} #foreach (\$port in \$node.trackingPorts) <td< td=""></td<></pre>
sandbox	A data structure describing a sandbox containing an executed graph. It contains sub-properties which are accessible using dot notation (i.e. \${sandbox.name})
	 sandbox.name sandbox.code sandbox.rootPath
schedule	A data structure describing a schedule which triggered this task. It contains sub-properties which are accessible using dot notation (i.e. \${schedule.description})
	 schedule.description schedule.startTime schedule.endTime schedule.lastEvent schedule.nextEvent schedule.fireMisfired

Table 28.2 Placebolders useful in email templates

Execute Shell Command

Execute Shell Command executes a system command or shell script.

This task is used in Chapter 30, <u>Scheduling(p. 193)</u> Chapter 32, <u>Listeners(p. 204)</u> and Chapter 29, <u>Manual</u> <u>Task Execution</u> (p. 192).

Table 28.3. Attributes of "Execute shell command" task

Name	Description	
Task type	"Execute shell command"	
Start on	Node IDs to process the task.	
	This attribute is accessible only in Cluster environment. If there are nodes specified, the task will be processed on the first node which is online and ready.	
Shell script	Command line for execution of external process.	
Working directory	A working directory for the process.	
	If not set, the working directory of the application server process is used.	
Timeout	Timeout in milliseconds. After a period of time specified by this number, the external process is terminated and all results are logged.	

Create graph event lis	ener	
Name	GraphEventListener_1	
Owner	clover -	
GRAPH TO CHECK	Enabled	
Sandbox	BasicExamples •	
Graph	graph/CreditCardFraudDetection.grf	•
Event type	Graph finished 🔻	
TRIGGERED TASK		
Task type	Execute shell command 🔹	
Shell script	1 du -hs . >> size.log	4
Working directory	/home/clover	
Timeout	10000 ms	
	Available variables	~
	Available variables	te Cano

Figure 28.2. Web GUI - shell command

Execute Shell Command Parameters

Some parameters are available only in particular context: scheduling, event listeners, or manual task execution.

Table 28.4. Parameters of Execute shell command

Name	Description
event	An event that has triggered the task
now	Current date-time
task	The triggered task
user	The object representing the user who executed the graph/jobflow. It contains sub- properties that are accessible using dot notation (i.e. \${user.email})
	 user.email user.username user.firstName user.lastName user.groups (list of values)
Parameters availabl	e in Scheduling
schedule	The object representing the schedule that triggered this task. It contains sub-properties that are accessible using dot notation (i.e. \${schedule.description})
	schedule.descriptionschedule.startTime
	schedule.endTime schedule lastEvent
	schedule.nastEvent
	• schedule.fireMisfired
EVENT_USERNAM	E The name of the user who caused the event.
EVENT_USER_ID	The numeric ID of the user who caused the event.
EVENT_SCHEDULI	E_DESCRIPTION A description of the schedule.
EVENT_SCHEDULI	E_EVENT_TYPE The type of the schedule - SCHEDULE_ONETIME or SCHEDULE_PERIODIC.
EVENT_SCHEDULI	E_ID The numeric ID of the schedule
EVENT_SCHEDULI	E_LAST_EVENT Date-time of the latest schedule triggering (in java.util.Date.toString() format).
Parameters availabl	e in Listeners
run	The object representing a single graph/jobflow execution. It contains sub-properties that are accessible using dot notation (i.e. \${run.jobFile}).
	• run.jobFile
	• run.status
	• run.stopTime
	• run.errNode
	• run.errMessage
	• run.errException

Name	Description
sandbox	The object representing a sandbox containing the executed graph/jobflow. It contains sub-properties that are accessible using dot notation (i.e. \${sandbox.name})
	sandbox.namesandbox.codesandbox.rootPath
tracking	An object representing a status of components in a graph execution. It contains sub- properties that are accessible using Velocity syntax for loops and conditions.
EVENT_USERNAM	EThe name of the user who caused the event.
EVENT_USER_ID	A numeric ID of the user who caused the event.
EVENT_RUN_SANI	BOX A code of the sandbox containing the graph/jobflow.
EVENT_RUN_JOB_	FILE A sandbox-relative path to the graph/jobflow file.
EVENT_RUN_RESU	 ILT The current status of the graph/jobflow execution N_A READY RUNNING WAITING FINISHED_OK ERROR ABORTED TIMEOUT
EVENT RUN ID	UNKNOWN A numeric ID of the run record representing graph/iohflow execution
EVENT_RON_ID	A specified timeout (in milliseconds) for the TIMEOUT event to occur
EVENT_JOB_EVEN	T_TYPE Graph event that triggered the task • GRAPH_STARTED • GRAPH_PHASE_FINISHED • GRAPH_FINISHED • GRAPH_ERROR • GRAPH_ABORTED
	GRAPH_TIMEOUTGRAPH_STATUS_UNKNOWN
Parameters availabl	e in Manual Task Execution
parameters	Task parameters - container for String-String key-value pairs passed to this task.

Start a Graph

Start a Graph starts a specified graph from a specified sandbox.

Table 28.5. Attributes of "Graph execution" task

Name	Description		
Task type	"Start a graph"		
Start on	Node(s) to process the task.		
	This attribute is accessible only in Cluster environment. If there are nodes specified, the task will be processed on the first node which is online and ready.		
Sandbox	This select box contains sandboxes which are readable for the logged user. Select the sandbox which contains the graph to execute.		
Graph	The graph to be executed.		
	This select box is filled with all graphs files accessible in the selected sandbox. Type a graph name or path to filter available items.		
Save run record	Saves run record to database.		
	If the task runs too often (once in several seconds), you can increase the database performance by disabling this attribute.		
Parameters	A list of parameters passed to the graph.		
	Event parameters like EVENT_RUN_RESULT, EVENT_RUN_ID, etc. are passed to the executed job without limitations. The EVENT_RUN_RESULT and EVENT_RUN_ID parameters are used in context of event listeners. They are not used in context of scheduling.		

Create graph event	listener	3
Graph even success or f	t listeners allow you to define a task that will be executed as a reaction to the ailure of executing a specific graph.	
Name	GraphEventListener_1	
Owner	clover 👻	
	Enabled	
GRAPH TO CHECK		
Sandbox	BasicExamples •	
Graph	graph/CreditCardFraudDetection.grf 👻	
Event type	Graph finished 💌	
TRIGGERED TASK		
Task type	Start a graph 🗸	
Sandbox	BasicExamples -	
Graph	graph/DataSelectionAdvanced.grf 🗸	
	Save execution history 0	
	Pass parameters from the checked graph	
Parameters	Name Value	
	Parameters passed from the listener to the task	~
	Create Canc	el

Figure 28.3. Web GUI - Graph execution task

Please note that the behavior of this task type is almost the same as <u>Start a Jobflow</u> (p. 182).

Parameters

You can start a graph with parameters.

To start a graph with a parameter, choose an existing parameter from the list, set its value, and click the **plus sign** button at the end of line.

If the graph is started by an event listener, it receives additional parameters from the triggering job.

Parameters passed to graph by Event Listeners

Table 28.6. Additional parameters available in Event Listeners

Name	Description	
EVENT_USERNAM	EThe name of the user who caused the event	
EVENT_USER_ID	A numeric ID of the user who caused the event.	
EVENT_RUN_SAN	DBOX	
	A code of the sandbox containing the graph/jobflow	
EVENT_RUN_JOB_	FILE	
	A sandbox-relative path to the graph/jobflow file.	
EVENT_RUN_RESULT		
	The current status of the graph/jobflow execution (N_A, READY, RUNNING,	
	WAITING, FINISHED_OK, EKKOK, ABOKTED, TIMEOUT OF UNKNOWN).	
EVENT_RUN_ID	A numeric ID of the run record representing graph/jobflow execution.	
EVENT_JOB_EVEN	T_TYPE	
	A graph/jobflow event type that triggered the task.	
EVENT_TIMEOUT	A specified timeout (in milliseconds) for the TIMEOUT event to occur	

Start a Jobflow

Start a jobflow starts a specified jobflow from a specified sandbox.

Table 28.7. Attributes of "Jobflow execution" task

Name	Description		
Task type	"Start a jobflow"		
Start on	Node(s) to process the task.		
	This attribute is accessible only in Cluster environment. If there are nodes specified, the task will be processed on the first node which is online and ready.		
Sandbox	This select box contains sandboxes which are readable for the logged user. Select sandbox which contains jobflow to execute.		
Jobflow	This select box is filled with all jobflow files accessible in the selected sandbox. Type the jobflow name or path to filter available items.		
Save run record	Saves run record to database. If the task runs too often (once in several seconds), you can increase the database performance by disabling this attribute.		
Parameters	Key-value pairs which are passed to the executed job as parameters.		
	Event parameters like EVENT_RUN_RESULT, EVENT_RUN_ID, etc. are passed to the executed job without limitations. The EVENT_RUN_RESULT and EVENT_RUN_ID parameters are used in context of event listeners. They are not used in context of scheduling.		

Graph even	It listeners allow you to define a task that will be executed as a reaction to the	
success or	failure of executing a specific graph.	
Name	GraphEventListener	
Owner	clover 👻	
	Enabled	
GRAPH TO CHECK		
Sandbox	All sandboxes 🔻	
Graph	Select sandbox first	
Event type	Graph finished 👻	
TRIGGERED TASK		٦
Task type	Start a jobflow 👻	
Sandbox	JobflowExamples 👻	
Jobflow	jobflow/03-ErrorHandling-Basic.jbf	
	Save execution history 0	
	Pass parameters from the checked graph	
Parameters	Name Value	
		_

Figure 28.4. Web GUI - Jobflow execution task

Please note that the behavior of this task type is almost the same as <u>Start a Graph</u> (p. 179).

If the jobflow start is triggered by an event, the same set of parameters as in a graph event listener is passed to the jobflow. <u>Parameters</u> (p. 181).

Start a Profiler Job

Start a profiler job starts a specified profiler job from a specified directory.

You can pass parameters to the profiler job in the same way as in case of starting a graph (p. 179) or jobflow (p. 182).

In case of triggering the profiler job by an event listener, the same set of additional parameters, as in case of execution of graph, is passed to the profiler job. See <u>Parameters passed to graph by Event Listeners</u> (p. 181).

success or fa	ilure of executing a specific gra	iph.	
Name	GraphEventListener		
Owner	clover -		
	 Enabled 		
GRAPH TO CHECK			
Sandbox	All sandboxes	•	
Graph	Select sandbox first		
Event type	Graph finished 🔹		
TRIGGERED TASK			
Task type	Start a profiler job		•
Sandbox	DataQualityExamples		•
Profiler job	profile/example1-XLS.	;pj	•
Parameters	Name	Value	0 -5

Figure 28.5. Web GUI - Profiler job execution task

Abort job

This task kills/aborts a specified job (graph or jobflow), if it is currently running.

Table 28.8. Attributes of "Abort job" task

Name	Description
Task type	"Abort job"
Start on	Node(s) to process the task.
	This attribute is accessible only in Cluster environment. If there are nodes specified, the task will be processed on the first node which is online and ready.
Kill source of event	If this switch is on, the task will kill the job which is the source of the event, which activated this task. Attributes sandbox and job are ignored.
	This checkbox is useful only if Abort job is activated by some event.
Sandbox	Select a sandbox which contains the job to kill.
	This attribute works only when the Kill source of event switch is off.
Job	This select box is filled with all jobs accessible in the selected sandbox. All instances of the selected job that are currently running and will be killed. This attribute works only when Kill source of event switch is off.

Name GraphEventListener Dwner clover Enabled SRAPH TO CHECK Sandbox JobflowExamples Graph graph/03-ErrorHandling-Basic/03-Graph-1.grf Event type Graph timeout Volo timeout interval 7200 seconds 120.0 minutes 2.0 hours	ner	Name GraphEve
Dwner clover Enabled Graph JobflowExamples Graph graph/03-ErrorHandling-Basic/03-Graph-1.grf Event type Graph timeout T200 seconds 120.0 minutes 2.0 hours FRIGGERED TASK FridgeRED TASK		
C Enabled Enabled C Enabled Enabled C Enabled C Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Enabled Ena		Owner clover
GRAPH TO CHECK Sandbox JobflowExamples Graph graph/03-ErrorHandling-Basic/03-Graph-1.grf Event type Graph timeout Graph timeout T200 Seconds 120.0 minutes 2.0 hours TRIGGERED TASK		✓ Enable
Sandbox JobflowExamples Graph graph/03-ErrorHandling-Basic/03-Graph-1.grf Event type Graph timeout Job timeout interval 7200 seconds 120.0 minutes 2.0 hours TRIGGERED TASK		GRAPH TO CHECK
Graph graph/03-ErrorHandling-Basic/03-Graph-1.grf Event type Graph timeout Job timeout interval 7200 seconds 120.0 minutes 2.0 hours TRIGGERED TASK	s v	Sandbox JobflowEx
Event type Graph timeout Job timeout interval 7200 seconds 120.0 minutes 2.0 hours TRIGGERED TASK	andling-Basic/03-Graph-1.grf	Graph graph/03-
Job timeout interval 7200 seconds 120.0 minutes 2.0 hours	•	Event type Graph tim
120.0 minutes 2.0 hours	conds	Job timeout interval 7200
2.0 hours	nutes	120.0
TRIGGERED TASK	urs	2.0
T al de la companya d		TRIGGERED TASK
Task type Abort job •	•	Task type Abort job
Kills/aborts specified job (ETL graph or jobflow), if it is currently runn	ed job (ETL graph or jobflow), if it is currently running.	Kills/aborts
✓ Kill source of event	event	✓ Kill so

Figure 28.6. Web GUI - "Abort job"

Archive Records

This task can archive (or delete) obsolete records from a database or filesystem.

Table 28.9. Attributes of "Archivator" task

Name	Description
Task type	"Archivator"
Start on	This attribute specifies a cluster node on which the task may process.
	This attribute is accessible only in Cluster environment.
	If it is empty, it may be any node; if there are nodes specified, the task will be processed on the first node which is online and ready.
Archivator type	There are two possible values: delete or archive.
	Delete removes records without any possibility of recovery.
	Archive option removes records from the database, but creates a ZIP package with CSV files containing the deleted data.
Older than	Time period (in minutes) - specifies which records are evaluated as obsolete. Records older than the specified interval are stored in archives/deleted.
Output path for archives	This attribute is useful only for the archive option.
Include executions history	If checked, Execution History (p. 199) will be archived/deleted as well.
Include temp files	If checked, the archivator saves/removes all graph temporary files older than the value in Older than attribute. The temporary files are files with graph debug data, dictionary files and files created by graph components. Note that log files are not considered as temp files.
Include tasks history	If checked, the archivator will include run records. Log files of graph runs are included as well.
Include profiler runs	If checked, the archivator will include profiler job results.
Include server instance history	If checked, the archivator will include server instance history as well.

GRAPH TO CHECK		
Sandbox	BasicExamples -	
Graph	graph/CreditCardFraudDetection.grf 🗸	
Event type	Graph finished 🔹	
TRIGGERED TASK		
Task type	Archivator	
	Archives (or deletes) obsolete logs or records from DB.	
Action	Delete Archive	
Older than	1440 minutes	
Output path for archives		
	✓ Include executions history	
Record status	<any status=""></any>	
Sandbox	<any sandbox=""></any>	
Job file	Select sandbox first	
	Include temp files	
	Include tasks history	
	Include profiler runs	
	Include conver instance run history	

Figure 28.7. Web GUI - archive records

Send a JMS Message

This type of task is useful for notifications about result of a graph execution. For example, you can create a graph event listener with this task type to be notified about each failure in a specific sandbox or failure of a particular graph.

JMS messaging requires JMS API (jms.jar) and third-party libraries. All these libraries must be available on the application server classpath. Some application servers contain these libraries by default, some do not, thus the libraries must be added explicitly.

Name	Description
Task type	"JMS message"
Initial context	Choose between the default and custom initial context.
Initial context class name	A full class name of javax.naming.InitialContext implementation. Each JMS provider has its own implementation. For example, in case of Apache MQ, it is org.apache.activemq.jndi.ActiveMQInitialContextFactory. If it is empty, the Server uses the default initial context.
Broker URL	
Connection factory JNDI name	The JNDI name of a connection factory. It depends on a JMS provider.
Destination JNDI name	The JNDI name of a message queue/topic on the server
Username	A username for connection to a JMS message broker
Password	A password for connection to a JMS message broker
URL	A URL of a JMS message broker
JMS pattern	This select box is available only when the user is creating a new record. It contains all predefined JMS message patterns. If the user chooses any of them, the text field below is automatically filled with a value from the pattern.
Text	The body of a JMS message. It is also possible to use placeholders. <i>See</i> <u>Placeholders</u> (p. 174) of send email task for details.

Table 28.10. Attributes of JMS message task

TRIGGERED TASK					
Task type	Send a JMS	message	•		
Initial context	Default	Custom			
Initial context factory class	org.apache.a	activemq.jndi.ActiveMQInitia	IContextFactory		
Broker URL	tcp://virt-brol	ker:61616			
Connection factory	QueueConn	ectionFactory			Q
Username	admin			0	
Password	•••••				
Queue/Topic	dynamicQue	ues/test			Q
Fill message from template	graph finishe	ed	•		
Message	Sandbox: \$ Graph: \${r Result: \${ Started: \$ Finished: Error node Error mess Error exce Log file:	<pre>{sandbox.code} un.graphId} run.status} {run.startTime} \$(run.stopTime} : \$!{run.errNode} age: \$!{run.errMessage ption: \$!{run.errKeep \$!{run.logLocation}</pre>	} tion}		
	Available v	ariables			*

Figure 28.8. Web GUI - Task JMS message editor

Table 28.11. Parameters of "Send a JMS Message"

Name	Description
event	The event that triggered the task.
now	Current date-time
task	The triggered task.
user	The object representing the owner of the schedule. It contains sub-properties that are accessible using dot notation (i.e. \${user.email}) email, username, firstName, lastName, groups (list of values).
schedule	The object representing the schedule that triggered this task. It contains sub-properties that are accessible using dot notation (i.e. \${schedule.description}) description, startTime, endTime, lastEvent, nextEvent, fireMisfired.
EVENT_USERNAM	EThe username of the user who caused the event
EVENT_USER_ID	A numeric ID of the user who caused the event.
EVENT_SCHEDULI	E_DESCRIPTION A description of the schedule
EVENT_SCHEDULI	E_EVENT_TYPE The type of the schedule - SCHEDULE_ONETIME or SCHEDULE_PERIODIC.
EVENT_SCHEDULI	_ID
	A numeric ID of the schedule.
EVENT_SCHEDULI	LAST_EVENT Date-time of the latest schedule triggering (in java.util.Date.toString() format).

Execute Groovy Code

This type of task allows to execute a code written in the Groovy script language. The script can be defined in place or using a path to external .groovy file. It is possible to use some variables.

The basic attribute of this task is a source code of written in Groovy.

If the source codes are provided from both a file and through the input form, only the code from the input form will be executed.

In Cluster environment, there is also one additional attribute **Node IDs to process the task**. If it is empty, it may be any node; if there are nodes specified, the task will be processed on the first node which is online and ready.

CloverDX Server contains Groovy version 2.0.0

Table 28.12. List of variables available in Groovy code

Variable	Class	Description	Availability
event	com.cloveretl.server.events.A	bstractServerEvent	every time
task	com.cloveretl.server.persister	it.Task	every time
now	java.util.Date	current time	every time
parameters	java.util.Properties	Properties of a task	every time
user	com.cloveretl.server. persistent.User	Same as event.getUser()	every time
run	com.cloveretl.server. persistent.RunRecord		When the event is an instance of GraphServerEvent
tracking	com.cloveretl.server. worker.commons.persistent.T	same as r ack.igg(Grapk ingGraph()	When the event is an instance of GraphServerEvent
sandbox	com.cloveretl.server. persistent.Sandbox	same as run.getSandbox()	When the event is an instance of GraphServerEvent
schedule	com.cloveretl.server. persistent.Schedule	same as ((ScheduleServerEvent)event). getSchedule()	When the event is an instance of ScheduleServerEvent
servletContext	javax.servlet.ServletContext		every time
cloverConfiguration	com.cloveretl.server.spring.C	Gærfignfägionatiohues for CloverDX Server	every time
serverFacade	com.cloveretl.server.facade. api.ServerFacade	The reference to the facade interface. Useful for calling CloverDX Server core. WAR file contains JavaDoc of facade API and it is accessible on URL: http://host:port/ clover/javadoc/index.html	every time
sessionToken	String	A valid session token of the user who owns the event. It is useful for authorization to the facade interface.	every time

Variables run, tracking and sandbox are available only if the event is an instance of GraphServerEvent class. A variable schedule is only available for ScheduleServerEvent as an event variable class.

Example of use Groovy script

This example shows a script which writes a text file describing the finished graph. It shows use of the 'run' variable.

```
import com.cloveretl.server.persistent.RunRecord;
String dir = "/tmp/";
RunRecord rr = (RunRecord)run ;
String fileName = "report"+rr.getId()+"_finished.txt";
FileWriter fw = new FileWriter(new File(dir+fileName));
fw.write("Run ID :"+rr.getId()+"\n");
fw.write("Graph ID :"+rr.getGraphId()+"\n");
fw.write("Sandbox :"+rr.getGraphId()+"\n");
fw.write("Sandbox :"+rr.getSandbox().getName()+"\n");
fw.write("Start time :"+rr.getStartTime()+"\n");
fw.write("Start time :"+rr.getStartTime()+"\n");
fw.write("Btart time :"+rr.getStartTime()+"\n");
fw.write("Duration :"+rr.getDurationString()+"\n");
fw.write("Status :"+rr.getStatus()+"\n");
fw.close();
```

Chapter 29. Manual Task Execution

Since 3.1

A manual task execution allows you to invoke a task directly with an immediate effect, without defining and triggering an event.

There are a number of task types that are usually associated with a triggering event, such as a file listener or a graph/jobflow listener. You can execute any of these tasks manually.

Additionally, you can specify task parameters to simulate a source event that would normally trigger the task. The following is a figure displaying how a 'file event' could be simulated. The parameters for various event sources are listed in the Chapter 27, <u>Graph/Jobflow Parameters</u> (p. 168).

without definin	kecution allows you to invok g a listener.	e a task directly with an immedia	te effect,
Task parameters	Name	Value	0-5
	EVENT FILE NAME	data file	•
	EVENT FILE PATH	C:\Users\clover\data	•
Task type	Start a graph	•	
Task type	Start a graph	•	
Sandbox	BasicExamples	•	
Graph	graph/CreditCardFraud	Detection.grf	•
	 Save execution histo 	огу 🚯	
			0
Parameters	Name	 Value 	

Figure 29.1. Web GUI - "Manual task execution" form

Using Manual Task Execution

In the Server GUI, switch to the **Event Listeners** tab. In the **New Listener** drop-down menu, select the **Manual Task Execution** option.

Choose the task type you would like to use. See documenation on chosen tasks:

Send an Email (p. 173) Execute Shell Command (p. 176) Start a Graph (p. 179) Start a Jobflow (p. 182) Start a Profiler Job (p. 184) Abort job (p. 185) Archive Records (p. 186) Send a JMS Message (p. 188) Execute Groovy Code (p. 190)

To access the Manual Task Execution form, you need Manual task execution permission (p. 137).

Chapter 30. Scheduling

The scheduling module allows you to create a time schedule for operations you need to trigger in a repetitive or timely manner.

Similar to cron from Unix systems, each schedule represents a separate time schedule definition and a task to perform.

In the left pane of the Scheduler GUI, you can see the status of schedules, details, date and time of last/next run and three-dot button with actions: **Run Now, Edit, Show in the Task History, Disable** and **Delete**. The right pane shows time line and details of the selected schedule.

Note that if a schedule is failing, you can see the error message in the right pane and there is an additional action available: **Clear Failure Indication**. When selecting this action, the schedule is no longer indicated as failing. It acts as if it were never run, but its timeline is still visible in the detail pane. This function is useful when you fix a failing schedule with a long interval between runs and by resetting the state, you mark the schedule as OK. This way, it doesn't cause distraction by being shown as failing until its status is refreshed in the next run.



Figure 30.1. Web GUI - Scheduling

Tasks you can schedule are described in Chapter 28, Tasks (p. 172).

Send an Email (p. 173) Execute Shell Command (p. 176) Start a Graph (p. 179) Start a Jobflow (p. 182) Start a Profiler Job (p. 184) Abort job (p. 185) Archive Records (p. 186) Send a JMS Message (p. 188) Execute Groovy Code (p. 190)

Timetable Setting

This section describes specification of triggering schedules. Note that exact trigger times are not guaranteed. There may be couple of seconds delay. Schedule itself can be specified in different ways.

<u>Onetime Schedule</u> (p. 194) <u>Periodical schedule by Interval</u> (p. 195) <u>Periodical schedule by timetable (cron Expression)</u> (p. 196)

Onetime Schedule

This schedule is triggered just once.

Table 30.1. Onetime schedule attribut

Name	Description
Periodicity	Onetime
Start time	Date and time, specified in the yyyy-mm-dd hh:mm:ss format.
Start skipped executions as soon as possible	If checked and execution is skipped for any reason (e.g. server restart), it will be triggered immediately when it is possible. Otherwise it is ignored and it will be triggered at the next scheduled time.

reate Schedule			
Schedule name	One time schedule		
Periodicity	Onetime	•	
Start time	2018-10-02 06:00:00	ė	
Jser	clover	•	
	 Start skipped execut 	ions as soon as poss	ible 🚯
	✓ Enable schedule		

Figure 30.2. Web GUI - onetime schedule form

Periodical schedule by Interval

This type of schedule is the simplest periodical type. Trigger times are specified by these attributes:

Table 30.2. Periodical schedule attributes

Name	Description
Periodicity	Interval
Run every	Specifies interval between two trigger times (in minutes). The next task is triggered even if the previous task is still running.
Active from/to	Date and time, specified in the yyyy-mm-dd hh:mm:ss format.
Start skipped executions as soon as possible	If checked and execution is skipped for any reason (e.g. server restart), it will be triggered immediately when it is possible. Otherwise it is ignored and it will be triggered at the next scheduled time.

C	reate Schedule		ж
1	Schedule name	Interval schedule	
	Periodicity	Interval -	
	Run every	60 minutes 👻	
	Active from	2018-09-01 08:30:00 🗎 to 2018-11-01 00:00:00 🗎	
	User	clover -	
		✓ Start skipped executions as soon as possible ()	
		Enable schedule	

Figure 30.3. Web GUI - periodical schedule form

Periodical schedule by timetable (cron Expression)

Timetable is specified by a cron expression.

Table 30.3. Cron periodical schedule attributes

Name	Description
Periodicity	Timetable
Cron expression	Cron is a job scheduler which uses its own format for scheduling. i.e. 0 0/2 4-23 * * ? means "every 2 minutes between 4:00 AM and 11:59 PM".
Active from/to	Date and time, specified in the yyyy-mm-dd hh:mm:ss format.
Start skipped executions as soon as possible	If checked and execution is skipped for any reason (e.g. server restart), it will be triggered immediately when it is possible. Otherwise it is ignored and it will be triggered at the next scheduled time.

Schedule name	Delete old debug files
Periodicity	Timetable •
ron expression	0 0 1 * * ? • Next run 2018-11-30 01:00:00
	Hint: At 01:00, every day
	SECOND MINUTE HOUR DAY MONTH DAY-OF-WEEK [YEAR]
Active from	<now> to <forever></forever></now>
Active from	<now> to <forever></forever></now>
Active from User	<now> to <forever> Cover Start skipped executions as soon as possible ①</forever></now>

Figure 30.4. Cron periodical schedule form

When setting up a cron expression, a hint displays it in a human readable format. Furthermore, when you click on each of the field in the expression, the hint expands, indicating which part of the expression you are editing and listing symbols, their meaning and values that can be used in the expression.



Note

Server cron expression for *Days of Week* differs from *nix cron expression. Days in cron expression in Server start from 1 which corresponds to Sunday. *nix cron expression uses 0 or 7 for Sunday.

Allocations of Scheduled Task on Nodes

In **Cluster** environment, you can set the node on which the scheduled task will be launched. If not set, the node will be selected automatically from all available nodes (but always just one).

If you choose:

- Any node one of the available nodes will be selected automatically.
- One of selected nodes you can select which node will run the task.

Nodes to process	Any node	One of selected nodes	
			-
	krychle-brno		
	virt-alpha		
	virt-gray		
	virt-proteus		

Figure 30.5. Schedule allocation - One ore more specific nodes

Scheduling the Tasks - Examples

Start a graph at specific time

This example shows scheduling the start of a graph at specific time.

- 1. In the Scheduling section of the Server GUI, click on the New schedule button.
- 2. Enter a **Schedule name**.
- 3. In the **Periodicity** drop-down list, choose **Onetime** to start the graph just once.
- 4. Enter **Start time**. Use the calendar into enter the required date and time in a correct format.
- 5. In the Task type field, choose Start a graph.
- 6. Select a **Sandbox** and a **Graph** within the sandbox.

Start a Jobflow once an hour

This example shows scheduling of a periodic task.

Create a new schedule that runs the UserStats.jbf jobflow from the reports sandbox once an hour.

- 1. Enter a **Schedule name**.
- 2. In the Periodicity drop-down list, choose Interval.
- 3. Enter the interval between two jobflow starts in the **Run every** field. In the drop-down list, you can choose between seconds, minutes and hours units.
- 4. The task will be started once an hour within a specified time period. Enter the beginning and end of this period in the **Active from to** fields.
- 5. Select the Sandbox and Graph.

Complex Scheduling

This example shows a complex scheduling using a cron expression.

Start a graph WeekendGraph.grf every Saturday and Sunday at 22:15.

- 1. Enter a **Schedule name**.
- 2. In the Periodicity drop-down list, choose Timetable.
- 3. Edit the **Cron expression** field. The expression for every Saturday and Sunday at 22:15 is 0 15 22 ? * 1, 7.
- 4. In the Task type field, choose Start a graph.
- 5. Select the Sandbox and Graph.

Chapter 31. Viewing Job Runs - Execution History

Filtering and ordering (p. 199) Tracking (p. 201) Log File (p. 202)

Execution History shows the history of all jobs that the Server has executed – transformation graphs, jobflows, and Data Profiler jobs. You can use it to find out why a job failed, see the parameters that were used for a specific run, and much more.

The table shows basic information about the job: Run ID, Node, Job file, Executed by, Status, and time of execution. After clicking on a row in the list, you can see additional details of the respective job, such as associated log files, parameter values, tracking and more.



Note

Some jobs might not appear in the Execution History list. These are jobs that have disabled persistence for increased performance (for example, some Launch Services disable storing the run information in order to increase service responsiveness).

Filtering and ordering

Use the Filter panel to filter the view. By default, only parent tasks are shown (Show executions children) – e.g. master nodes in a cluster and their workers are hidden by default.

Use the up and down arrows in the table header to sort the list. By default, the latest job is listed first.

Sclover DX	5.1.0.11				्रै Page Loading virt-alpha / test5	1 2018-12-17	09:58 +01:00	≡ θ
 Monitoring Status History Server Logs 	^	Filter Run ID Date from		Executed by Status	Sandbox File			^
b Execution History		Date to		Error message	Nested jobs			Filter Reset
Task History		Run ID	Job		N	ode Us	er Status	Started
🛱 Schedules	0	335	BasicExamples 🛸 graph/DatabaseAccess.grf		vi	t-alpha clov	ver 🕕	2018-12-13 15:54:02
😭 Data Services	0 ~	334 🥷	clovereti.test.scenarios 🛸 graph/bigRecords/XMLWriterExtract1.grf		vi	t-alpha clor	ver 📀	2018-12-13 15:34:16
 Event Listeners Launch Services 		333	clovereti.test.scenarios 🛸 graph/bigRecords/XMLWriterExtract.grf		vi	t-proteus clo	ver 📀	2018-12-13 15:34:14
4 Launar corricco		332 🧸	clovereti.test.scenarios 🛸 graph/bigRecords/DedupLast.grf		vi	t-alpha clo	ver 🔮	2018-12-13 15:34:11
Sandboxes		331 🥷	clovereti.test.scenarios 🐃 graph/bigRecords/DataIntersectionKeyDuplicates.grf		vi	t-alpha clo	ver 🔮	2018-12-13 15:34:09
& Configuration	^	330	clovereti.test.scenarios 🐃 graph/bigRecords/CloverDataReaderWriter.grf		vi	t-alpha clo	ver 🔮	2018-12-13 15:34:07
 Users 		329 🧸	clovereti.test.scenarios 🛸 graph/bigRecords/BufferedEdge.grf		k	ychle-brno clov	ver 🔮	2018-12-13 15:34:04
 Groups Security 		328	cloveretl.test.scenarios 🐃 graph/bigRecords/bigRecords_UDR.grf		vi	t-alpha clo	ver 🔮	2018-12-13 15:34:02
 Temp Spaces 		327	cloveretl.test.scenarios 🐃 graph/bigRecords/bigRecords_UDR-Verbose.grf		vi	t-alpha clo	ver 🔮	2018-12-13 15:34:00
 System Info CloverDX Info 		326	cloveretl.test.scenarios 🐃 graph/bigRecords/CloverDataReaderWriter.grf		vi	t-alpha clo	ver 🕑	2018-12-13 15:33:03
 Export 		325	cloveretl.test.scenarios 🐃 graph/bigRecords/XMLWriterExtract.grf		vi	t-alpha clov	ver 📀	2018-12-13 15:32:58
 Import 		324	BasicExamples 🛸 graph/XMLProcessing.grf		vi	t-proteus clo	ver 📀	2018-12-13 15:30:11
- Setup		323 27 pages	BasicExamples 🛸 graph/Lining&ogregating.grf	₩4 ← 18 19 20 21 22 23 24 25 26 2	27 → ₩	t-alpha clov	ver 📀	2018-12-13 15-30-08 25 ¥

Figure 31.1. Execution History - executions table

When some job execution is selected in the table, the detail info is shown on the right side.

Name	Description
Run ID	A unique number identifying the run of the job. Server APIs usually return this number as a simple response to the execution request. It is useful as a parameter of subsequent calls for specification of the job execution.
Execution type	A type of a job as recognized by the Server. STANDALONE for graph, JOBFLOW for Jobflow, PROFILER_JOB for profiler, MASTER for the main record of partitioned execution in a cluster, PARTITIONED_WORKER for the worker record of partitioned execution in a cluster
Parent run ID	A run ID of the parent job. Typically the jobflow which executed this job, or master execution which encapsulates this worker execution.
Root run ID	A run ID of the root parent job. Job execution which wasn't executed by another parent job.
Execution group	Jobflow components may group sub-jobs using this attribute. See the description of Jobflow components for details.
Nested jobs	Indication that this job execution has or has not any child execution.
Node	In cluster mode, it shows the ID of the cluster node which this execution was running on.
Executor	If it runs on worker, it contains the text "worker".
Executed by	The user who executed the job. Either directly using some API/GUI or indirectly using the scheduling or event listeners.
Sandbox	The sandbox containing a job file. For jobs which are sent together with an execution request, so the job file doesn't exist on the Server site, it is set to the "default" sandbox.
Job file	A path to a job file, relative to the sandbox root. For jobs which are sent together with an execution request, so the job file doesn't exist on the Server site, it is set to generated string.
Job version	The revision of the job file. A string generated by CloverDX Designer and stored in the job file.
Status	Status of the job execution. READY - waiting for execution start, RUNNING - processing the job, FINISHED OK - the job finished without any error, ABORTED - the job was aborted directly using some API/GUI or by the parent jobflow, ERROR - the job failed, N/A (not available) - the server process died suddenly, so it couldn't properly abort the jobs. After restart, the jobs with unknown status are set as N/A
Started	Server date-time (and timezone) of the execution start.
Finished	Server date-time (and timezone) of the execution finish.
Duration	Execution duration
Error in component ID	If the job failed due the error in a component, this field contains the ID of the component.
Error in component type	If the job failed due the error in a component, this field contains type of the component.
Error message	If the job failed, this field contains the error description.
Exception	If the job failed, this field contains error stack trace.
Input parameters	A list of input parameters passed to the job. A job file can't be cached, since the parameters are applied during loading from the job file. The job file isn't cached, by default.
Input dictionary	A list of dictionary elements passed to the job. A dictionary is used independently of job file caching.
Output dictionary	A list of dictionary elements at the moment the job ends.

Table 31.1. Persistent run record attributes

For jobs which have some children executions, e.g. partitioned or jobflows also an executions hierarchy tree is shown.

CloverDX 5.1.0.	11				✓ Page Loaded	virt-alpha / test51	2018-12-17 10:01 +01:00			1	9
Monitoring Status History Server Logs	Filter Run ID = Sandbox	Q. Search • Ø 79 Qa virt-alpha . Ø 80 🙊 krychle-b	Status JobflowExamples/jobflow/04-ErrorHandling-RejectFiles.jbf mo_JobflowExamples/graph/04-ErrorHandling-RejectFiles/04	t-Graph-1.orf [Vaidate inputs]			0	۵	•	×	^
C Execution History	Status Date to Nested jobs	 81 @ virt-alpha 82 3 wirt-prote 83 @ krychl 84 @ virt-pro 85 @ krychl 	a JobflowExamples/graph/04-ErrorHandling-RejectFiles/04-G us JobflowExamples/jobflow/04-ErrorHandling-RejectFiles- s-bmo JobflowExamples/graph/04-ErrorHandling-RejectFiles toteus JobflowExamples/graph/04-ErrorHandling-RejectFiles -bmo JobflowExamples/graph/04-ErrorHandling-RejectFiles	raph-2 grt [Report Reject Stats] Julphöllowijh [Consolidate Reject Files] (3 children) J04-Graph-3 grt [Consolidate Each File] J04-Graph-3 grt [Consolidate Each File]							
 i and a Schedules i and a Services ✓ 	Run ID Job 79 Job	Overview Tracking	Log file								
 Event Listeners Launch Services 	78 clov Regra	Run ID	79					*		D	-
Sandboxes	76 clos Recipional clos Recipional clos Recipional close Recipional close Recipional Recipional close Recipional close Recipional	> Nested jobs	 Joonow virt-alpha 								-
 Configuration Users 	75 clov R gra 74 clov	Executor Job file	worker jobflow/04-ErrorHandling-RejectFiles.jbf 📚								
Groups	R gra	Status	Finished OK								
 Security Temp Spaces 	🧶 gra	Finished	2018-11-21 10:28:52 2018-11-21 10:29:11								
 System Info 		Duration	19 secs								Ш
 CloverDX Info 		Input parameters	CLOVER_USERNAME	clover							
- Export			JOB_FILE	jobflow/04-ErrorHandling-RejectFiles.jbf							
 Import 			NODE_ID	virt-alpha							
- Setup			PARENT_RUN_ID	79							
<	27 pages		ROOT_RUN_ID	79							

Figure 31.2. Execution History - overall perspective



Tip

By clicking the arrow on the left side of the detail pane (see the figure above), you can expand the detail pane for easier analysis of job logs.

Executions hierarchy may be rather complex, so it's possible to filter the content of the tree by a fulltext filter. However when the filter is used, the selected executions aren't hierarchically structured.

Tracking

The Tracking tab, contains details about the selected job:

Table 31.2.	Tracking t	table	information
-------------	------------	-------	-------------

Attribute	Description
Component ID	The ID of the component.
Component name	The name of the component.
Status	Status of data processing in the respective component.
	 FINISHED_OK: data was successfully processed ABORTED: data processing has been aborted
	Abor red. data processing has been aborted N A: status unknown
	• ERROR: an error occurred while data was processed by the component
CPU	CPU usage of the component.
Port	Component's ports (both input and output) that were used for data transfer.
Records	The number of records transferred through the port of the component.
kB	Amount of data transferred in kB.
Records/s	The number of records processed per second
KB/s	Data transfer speed in KB.
Records/s peak	The peak value of Records/s .
KB/s peak	The peak value of KB/s .

werview Tracking Log	Sia											
🥑 jobflowiRunTest.jbf									Ł	8	0	×
COMPONENT	STATUS	CPU	PORT	RECORDS	KB	REC 5/5	KB/S	RECS/S PEAK	KB/S PEAI			
 Phase 0 (3 mins), Memory 	y utilization 1,740MB											
EBM case 1 id: EBM_CASE_1	٥	0.00%										
			OUTPUT/0	1	0.18	0	0.00	0			0.00	
EBM case 2 id: EBM_CASE_2	٥	0.07%										
			INPUT/0	1	0.18	0	0.00	0			0.00	
			OUTPUT/0	1	0.18	0	0.00	0			0.11	
GeoCoding id: GEO_CODING	٥	0.00%										
			INPUT/0	1	0.17	0	0.00	0			0.13	
			OUTPUT/0	1	0.17	0	0.00	0			0.00	
JSONBuilding Id: JSONBUILDING	٥	0.00%										
			INPUT/0	1	0.17	0	0.00	0			0.00	
			OUTPUT/0	1	0.18	0	0.00	0			0.17	
KLM id: KLM	٥	0.01%										
			INPUT.0	1	0.20	0	0.00	0			0.13	
			OUTPUT/0	1	0.17	0	0.00	0			0.00	
Mondial		0.017										

Figure 31.3. Execution History - Tracking

Log File

In the **Log file** tab, you can see the log of the job run with detailed information. A log with a green background indicates a successfully run job, while a red background indicates an error.

You can download the log as a plain text file by clicking **Download log** or as a zip archive by clicking **Download log** (zipped).

Overview	Tracking	Log file												
🕑 grap	h/CreditCa	rdFraudD	etectio	on.grf							¥	Į,	0	×
j Download	l log (10 KB)	Down	load lo	g (zipped)										
2018-09-29 2018-09-29 compiled 3	5 08:41:57, 5 08:41:57, 24/09/2018	736 INFO 736 INFO 21:34:22	393221 393221	[http-n: [http-n:	io-8083 io-8083	-exec-3 -exec-3] *** C] Runnin	loverD) g with	(, (c) 20 CloverDX	02-2018 Versio	Javlin n 5.0.0	a.s. build#	*** #015	
:018-09-2: md64 :018-09-2:	5 08:41:57, 5 08:41:57,	737 INFO 964 INFO	393221 393221	[http-n:]JobStai	io-8083 rter_39	-exec-3] 3221] Ex] Runnin xecuting	g on 4	CPU(s),	OS Wind	ows 10,	archit	tecture	
asicExam I	ples/graph/ Run ID: User:	CreditCard	dFraudD 3932 clov	etection 21 er	.grf									
018-09-2	Triggered b Executing o 5 08:41:57,	y: n: 994 INFO	Web [wor 393221	GUI ker0@nod	e01] cutorThi	readPool	lFactory	Bean-4]	Running	on Jav	a(TM) SE	Runt:	ime	
nvironmer 018-09-25	nt, 1.8.0_1 5 08:41:58,	44, Oracle 083 INFO	e Corpo 393221	[jobExe	JDK), ma cutorThi	ax avail readPool	lable me lFactory	mory fo Bean-4]	or JVM 2 Initial	080 768 izing p	KB hase 0		£11	
018-09-2 andbox:/,	5 08:41:58, 5 08:41:58, /BasicExamp	362 INFO les/graph/	393221 (Credit	[]obExe CardFrau	cutorThi dDetect:	readPool ion.grf	lFactory	Bean-4]	Executi	ng job	#393221	-	orige.	
2018-09-25 2018-09-25 CLOVER US	5 08:41:58, 5 08:41:58, ERNAME=clov	363 INFO 364 INFO	393221 393221	[WatchDo [WatchDo	og_3932: og_3932:	21] Job 21] Job	executi paramet	on type ers:	e: ETL_GR	APH				
ONN_DIR=	\${PROJECT}/ R=\${PROJECT TR=\${PROJECT	conn }/data-in	.t											
ATATMP_D	IR=\${PROJEC =\${PROJECT}	T}/data-tm /graph												

Figure 31.4. Execution History - Tracking

Chapter 32. Listeners

Listeners can be seen as 'hooks'. They wait for a specific event and take a used-defined action if the event occurs.

Created listeners are shown in the Event Listener list (see Figure below). The list has the following columns :

Table 32.1. Event Listeners Table Description

Column name	Description
Enabled	Indicates whether the listener is enabled or disabled. Clicking the icon enables/disables the listener.
Name	Shows the name of the listener, task type and event the listener is waiting for.
Last run	Shows the date and time of the listener's last run.
ОК	Shows the number of successful runs.
FAIL	Shows the number of failed runs.
	Click the three vertical dots to open a submenu where you can create an email notification or delete the listener.

New Listener

Enabled	Name	Last run	ок	FAIL	
	Check parameters Start a graph BigDataExamples/graph/CheckParameters-Hive.grf On finished of graph BasicExamples/graph/_Introduction.grf	2018-09-25 13:22:22	0	1	
	Detect transactions Start a graph BasicExamples/graph/subgraph/DetectRiskyTransactions.sgrf On file *.txt added at /opt		0	0	
	Send email Send an email \${user.email} On finished of jobflow /				
	Start cpj Start a profiler job DataQualityExamples/profile/example2-CSV.cpj On finished of graph /	2018-09-25 13:22:27	1	0	***

Figure 32.1. Listeners

The event is specific to the particular listener

Graph Event Listeners (p. 206) Jobflow Event Listeners (p. 212) JMS Message Listeners (p. 214) Universal Event Listeners (p. 219) File Event Listeners (remote and local) (p. 221) Task Failure Listeners (p. 229)

The available actions taken by the listeners are common for all listeners. The actions, that can be taken are:

Send an Email (p. 173) Execute Shell Command (p. 176) Start a Graph (p. 179) Start a Jobflow (p. 182) Start a Profiler Job (p. 184) Abort job (p. 185) Archive Records (p. 186) Send a JMS Message (p. 188) Execute Groovy Code (p. 190)

Graph Event Listeners

Graph Event Listeners allow you to define a task that the Server will execute as a reaction to the success, failure or other event of a specific job (a transformation graph).

Each listener is bound to a specific graph and is evaluated no matter whether the graph was executed manually, scheduled, or via an API call, etc.

You can use listeners to chain multiple jobs (creating a success listener that starts the next job in a row). However, we recommend using Jobflows to automate complex processes because of its better development and monitoring capabilities.

Graph Event Listeners are similar to Jobflow Event Listeners (Jobflow Event Listeners (p. 212)) – for CloverDX Server both are simply 'jobs'.

In the Cluster, the event and the associated task are executed on the same node the job was executed on, by default. If the graph is distributed, the task will be executed on the master worker node. However, you can override where the task will be executed by explicitly specifying a Node IDs in the task definition.

Graph Events

Each event carries properties of a graph, which is the source of the event. If there is an event listener specified, the task may use these properties. For example the next graphs in a chain may use "EVENT_FILE_NAME" placeholder which was activated by the first graph in the chain. Graph properties, which are set specifically for each graph run (e.g. RUN_ID), are overridden by the last graph.

Types of graph events

Graph started

The Graph started event is created, when an graph execution successfully started.

Graph phase finished

The **Graph phase finished** event is created, everytime a graph phase is finished and all its nodes are finished with status FINISHED_OK.

Graph finished

The **Graph finished** event is created, when all phases and nodes of a graph are finished with FINISHED_OK status.

Graph error

The **Graph error** event is created, when a graph cannot be executed for some reason, or when any node of graph fails.

Graph aborted

The **Graph aborted** event is created, when a graph is explicitly aborted.

Graph timeout

The **Graph timeout** event is created, when a graph runs longer than a specified interval. Thus you should specify the **Job timeout interval** for each listener of a graph timeout event. You can specify the interval in seconds, minutes or hours.

Graph unknown status

The **Graph unknown status** event is created when the Server, during the startup, detects run records with undefined status in the Execution History. Undefined status means, that the Server has been killed during the graph run. The Server automatically changes the state of the graph to *Not Available* and sends a *graph unknown status* event.

Please note that this works just for executions, which have a persistent record in the Execution History. It is possible to execute a transformation without a persistent record in the Execution History, typically for better performance of fast running transformations (e.g. using Launch Services).

Listener

User may create a listener for a specific event type and graph (or all graphs in sandbox). The listener is actually a connection between a graph event and a task, where the graph event specifies *when* and the task specifies *what* to do.

Event handling consists of the following course of actions:

- the event is created
- listeners for this event are notified
- each listener performs the related task

Tasks

Task types are described in Chapter 28, Tasks (p. 172).

In the Cluster environment, all tasks have an additional attribute **Node IDs** to process the task. If there is no node ID specified, the task may be processed on any cluster node. In most cases, it will be processed on the same node where the event was triggered. If there are some nodeIDs specified, the task will be processed on the first node in the list which is connected in cluster and ready.

Send an Email (p. 173) Execute Shell Command (p. 176) Start a Graph (p. 179) Start a Jobflow (p. 182) Start a Profiler Job (p. 184) Abort job (p. 185) Archive Records (p. 186) Send a JMS Message (p. 188) Execute Groovy Code (p. 190)

Use Cases

Possible use cases are:

- Execute graphs in chain (p. 207)
- Email notification about graph failure (p. 209)
- Email notification about graph success (p. 210)
- Backup of data processed by graph (p. 211)

Execute graphs in chain

For example, we have to execute graph B, only if another graph A finished without any error. So there is a relation between these graphs. We can achieve this behavior by creating a graph event listener. We create a listener for

graph finished OK event of graph A and choose an execute graph task type with graph B specified for execution. If we create another listener for graph B with the execute graph task with graph C specified, it will work as a chain of graphs.

Graph ever success or	It listeners allow you to define a task that will be executed as a reaction to th failure of executing a specific graph.	e
Name	ChainedGraphs	
Owner	clover -	
	Enabled	
GRAPH TO CHECK	dofault -	
Salubox	default 👻	
Graph	Type to search	•
Event type	Graph finished 👻	
TRIGGERED TASK		
Task type	Start a graph 🔹	
Sandbox	default 👻	
Graph	graph/graphExtFilter.grf	•
	Save execution history 1	
	Pass parameters from the checked graph	
Parameters	Name Value	0-0

Figure 32.2. The event source graph isn't specified, thus the listener works for all graphs in the specified sandbox

Email notification about graph failure

Create graph event li	stener	
GRAPH TO CHECK		
Sandbox	default 👻	
Graph	Type to search	•
Event type	Graph error 👻	
TRIGGERED TASK		
Task type	Send an email	
Email template	No template Job finished	
То	\${user.email}	
	Show Cc, Bcc and Reply-to	
Subject	CloverDX Server notification - Graph run \${run.id} of \${run.jobF	ile} finish
HTML	<hi>Graph run \${run.id} of \${sandbox.code} / \${run.jobFile} error</hi> <pre><pre><pre>><pre>State</pre></pre></pre></pre>	
	<pre>cp>finished: \${run.storTime} #if(\${run.errNode})</pre>	
	Error node: \$!{run.errNode} #end #ff(ff) = 1 = 2 = 2 = 2 = 2 = 2 = 2 = 2 = 2 = 2	▼
	Show plain text	
	 Log file as attachment (if it is available) 	
	Available variables	~
	Create	Cancel

Figure 32.3. Web GUI - email notification about graph failure
default 👻	
Type to search	•
Graph finished 🔹	
Send an email 🔹	
No template Job finished	
\${user.email}	
Show Cc, Bcc and Reply-to	
CloverDX Server notification - Graph run \${run.id} of \${run.jobFil	e} finish
<pre><hi>Graph run \${run.id} of \${sandbox.code} / \${run.jobFile} finished</hi> runId: \${run.id} rp>ser: \${run.user.username} Result: \${run.statu5/rp> Finished: \${run.start1me} Finished: \${run.start7me} #if(\${run.errNode}) Fror node: \$!{run.errNode} #if(\$trun.errNode} the text Log file as attachment (if it is available)</pre>	•
	<pre>default Type to search Graph finished Send an email No template Job finished S(user.email) Show Cc, Bcc and Reply-to CloverDX Server notification - Graph run \${run.id} of \${run.jobFill} (h1>Graph run \${run.id} of \${sandbox.code} / \${run.jobFile} finished (p>Result: \${run.status} (p>Started: \${run.status} (p>Started: \${run.ertNode}) (p>Frinished: \${run.ertNode}) Show plain text Log file as attachment (if it is available)</pre>

Figure 32.4. Web GUI - email notification about graph success

Backup of data processed by graph

reate graph event lis	lener	
Name	DataBackup	
Owner	clover -	
	 Enabled 	
GRAPH TO CHECK		
Sandbox	default 👻	
Graph	graph/graphDataPolicy.grf	-
Event type	Graph finished •	
TRIGGERED TASK		
Task type	Execute shell command -	
Shell script	<pre>1 /opt/scripts/backup_data.sh</pre>	J
Working directory	/var/data	
Timeout	10000 ms	
	Available variables	*

Figure 32.5. Web GUI - backup of data processed by graph

Jobflow Event Listeners

Jobflow Event Listeners allow you to define a task that the Server will execute as a reaction to the success or failure of executing a specific job (a jobflow).

Each listener is bound to a specific jobflow and is evaluated every time the jobflow is executed (no matter whether manually, through another jobflow, via a schedule, API call, etc.).

Jobflow Event Listeners work very similarly to <u>Graph Event Listeners</u> (p. 206) in many ways, since Graphs and Jobflows are both 'jobs' from the point of view of the **CloverDX Server**.

In the Cluster, the event and the associated task are executed on the same node the job was executed on. If the jobflow is distributed, the task will be executed on the master worker node. However, you can override the default setting by explicitly specifying a Node ID in the task definition.

Jobflow Events

Each event carries properties of the event source job. If there is an event listener specified, a task may use these properties. For example, the next job in the chain may use "EVENT_FILE_NAME" placeholder which activated the first job in the chain. Job properties, which are set specifically for each run (e.g. RUN_ID), are overridden by the last job.

Types of jobflow events

Jobflow started

A Jobflow started event is created, when jobflow execution successfully started.

Jobflow phase finished

The **Jobflow phase finished** event is created everytime a jobflow phase is finished and all its nodes are finished with the FINISHED_OK status.

Jobflow finished

The **Jobflow finished** event is created, when all phases and nodes of a jobflow are finished with the FINISHED_OK status.

Jobflow error

The **Jobflow error** event is created, when a jobflow cannot be executed for some reason, or when any node of the jobflow fails.

Jobflow aborted

The Jobflow aborted event is created, when a jobflow is explicitly aborted.

Jobflow timeout

The **Jobflow timeout** event is created when a jobflow runs longer then the specified interval. Thus you have to specify the **Job timeout interval** for each listener of the jobflow timeout event. You can specify this interval in seconds, minutes or hours.

Jobflow unknown status

The **Jobflow unknown status** event is created, when the Server, during the startup, detects run records with undefined status in the Execution History. Undefined status means, that the Server has been killed during the jobflow run. The server automatically changes the state of the jobflow to *Not Available* and sends the *jobflow status unknown* event.

Please note, that this works just for executions, which have a persistent record in the Execution History. It is possible to execute a transformation without a persistent record in the Execution History, typically for better performance of fast running transformations (e.g. using Launch Services).

Listener

The user may create a listener for the specified event type and jobflow (or all jobflows in sandbox). The listener is actually a connection between the jobflow event and task, where the jobflow event specifies *when* and the task specifies *what* to do.

Event handling consists of the following course of actions:

- event is created
- listeners for this event are notified
- each listener performs the related task

Tasks

A task specifies an operation which should be performed as a reaction to a triggered event.

Task types are described in Chapter 28, Tasks (p. 172).

Note: You can use a task of any type for a jobflow event listener. The description of task types is divided into two sections just to show the most evident use cases.

Send an Email (p. 173) Execute Shell Command (p. 176) Start a Graph (p. 179) Start a Jobflow (p. 182) Start a Profiler Job (p. 184) Abort job (p. 185) Archive Records (p. 186) Send a JMS Message (p. 188) Execute Groovy Code (p. 190)

JMS Message Listeners

JMS Message Listeners allow you to listen for incoming JMS messages. You specify the source of the messages (JMS Topic or JMS Queue) and a task that will be executed for each incoming message.

JMS messaging requires a JMS API (jms.jar) and specific third-party libraries. Every one of these libraries must be available on a classpath of an application server. Some application servers contain these libraries by default; however, some do not. In such a case, libraries must be added explicitly before starting the **CloverDX Server**.



JMS Message Listeners on Worker

JMS Message Listeners can be used with Worker as well. In such a case, make sure that the required . jar file and libraries are available on the Worker's classpath as well. For more information, see <u>Adding Libraries to the Worker's Classpath</u> (p. 41).

JMS is a complex topic that goes beyond the scope of this document. For more detailed information about JMS, refer to the Oracle website: <u>https://docs.oracle.com/javaee/7/tutorial/jms-concepts.htm#BNCDQ</u>

Note that the JMS implementation is dependent on the application server that the CloverDX Server is running in.

In Cluster, you can either explicitly specify which node will listen to JMS or not. If unspecified, all nodes will register as listeners. In the case of JMS Topic, all nodes will get the message and trigger the task (multiple instances) or, in the case of JMS Queue, a random node will consume the message and run the task (just one instance).

Attribute	Description
Initialize by	This attribute is useful only in a cluster environment. It is a node ID where the listener should be initialized. If it is not set, the listener is initialized on all nodes in the cluster.
	In the Cluster environment, each JMS event listener has a Node IDs attribute which may be used to specify the cluster node which will consume messages from the queue/ topic. There are the following possibilities:
	• No failover: Just one node ID specified - Only the specified node may consume messages, however the node status must be "ready". When the node isn't ready, messages aren't consumed by any cluster node.
	• Failover with node concurrency: No node ID specified (empty input) - All cluster nodes with status "ready" consume messages concurrently.
	• Failover with node reservation: More node IDs specified (separated by a comma) - Just one of the specified nodes consumes messages at a time. If the node fails for any reason (or its status isn't "ready"), any other "ready" node from the list continues with consuming messages.
	In a standalone environment, the Node IDs attribute is ignored.
JNDI Access	
Initial context	Default or custom
Initial context factory class	A full class name of the javax.naming.InitialContext implementation. Each JMS provider has its own implementation. For example, Apache MQ has org.apache.activemq.jndi.ActiveMQInitialContextFactory. If it is empty, the Server uses a default initial context.
	is usually included in one library with a JMS API implementation for each specific JMS broker provider.

Table 32.2. Attributes of JMS message task

Attribute	Description
Broker URL	A URL of a JMS message broker
Listen To	
Connection factory	A JNDI name of a connection factory. It depends on a JMS provider.
Username	A username for a connection to a JMS message broker
Password	A password for a connection to JMS message broker
Queue/Topic	A JNDI name of a message queue/topic on the Server
Durable subscriber	If false, the message consumer is connected to the broker as 'non-durable', so it receives only messages which are sent while the connection is active. Other messages are lost.
	If the attribute is true, the consumer is subscribed as 'durable' so it receives even messages which are sent while the connection is inactive. The broker stores such messages until they can be delivered or until the expiration is reached.
	This switch is useful <i>only for Topics</i> destinations, because Queue destinations always store messages until they can be delivered or the expiration is reached.
	Please note that consumer is inactive e.g. during server restart and during short moment when the user updates the "JMS message listener" and it must be re-initialized. So during these intervals, the message in the Topic may get lost if the consumer does not have the durable subscription.
	If the subscription is durable, client must have ClientId specified. This attribute can be set in different ways in dependence on JMS provider. E.g. for ActiveMQ, it is set as a URL parameter tcp://localhost:1244?jms.clientID=TestClientID.
Subscriber name	Available only when Durable subscriber is true. By default, a durable subscriber name is generated automatically in the subscr_[clusterNodeId]_[listenerId] format; therefore, a subscriber has a different name on each cluster node. Using this attribute, you can specify a custom subscriber name that will be identical on all cluster nodes.
Message selector	This query string can be used as a specification of conditions for filtering incoming messages. Syntax is well described on Java EE API web site. It has different behavior depending on the type of consumer (queue/topic):
	Queue: Messages that are filtered out remain in the queue.
	Topic: Messages filtered out by a Topic subscriber's message selector will never be delivered to the subscriber. From the subscriber's perspective, they do not exist.
Message Processing	
Number of consumers	E.g. 1
Groovy code	A Groovy code may be used for additional message processing and/or for refusing a message. Both features are described below.

Optional Groovy code

Groovy code may be used for additional message processing or for refusing a message.

- Additional message processing Groovy code may modify/add/remove values stored in the containers "properties" and "data".
- **Refuse/acknowledge the message** If the Groovy code returns Boolean.FALSE, the message is refused. Otherwise, the message is acknowledged. A refused message may be redelivered, however the JMS broker

should configure a limit for redelivering messages. If the Groovy code throws an exception, it's considered a coding error and the JMS message is NOT refused because of it. So, if the message refusal is to be directed by some exception, it must be handled in Groovy.

de

Туре	Key	Description
javax.jms.Message	msg	instance of a JMS message
java.util.Properties	properties	See below for details. It contains values (String or converted to String) read from a message and it is passed to the task which may then use them. For example, the execute graph task passes these parameters to the executed graph.
java.util.Map <string, Object></string, 	data	See below for details. Contains values (Object, Stream, etc.) read or proxied from the message instance and it is passed to the task which may then use them. For example, the execute graph task passes it to the executed graph as dictionary entries.
javax.servlet.ServletC	lontext servletCor	itext An instance of ServletContext.
com cloveretl server ani ServerFacade		
	serverFaca	An instance of serverFacade usable for calling CloverDX Server core
· 1 0. :	·	
java.lang.String	sessionTol	ken SessionToken needed for calling serverFacade methods

Message data available for further processing

A JMS message is processed and the data it contains is stored into two data structures: Properties and Data.

Key	Description
JMS_PROP_[propert key]	For each message property, one entry is created where "key" is made of the JMS_PROP_ prefix and property key.
JMS_MAP_[map entry key]	If the message is an instance of MapMessage, for each map entry, one entry is created where "key" is made of the JMS_MAP_ prefix and map entry key. Values are converted to String.
JMS_TEXT	If the message is an instance of TextMessage, this property contains content of the message.
JMS_MSG_CLASS	A class name of a message implementation
JMS_MSG_CORREI	ATHONHO ID is either a provider-specific message ID or an application-specific String value
JMS_MSG_DESTIN	AIIHONMSDestination header field contains the destination to which the message is being sent.
JMS_MSG_MESSAG	EVIS MessageID is a String value that should function as a unique key for identifying messages in a historical repository. The exact scope of uniqueness is provider-defined. It should at least cover all messages for a specific installation of a provider, where an installation is some connected set of message routers.
JMS_MSG_REPLYT	A destination to which a reply to this message should be sent.
JMS_MSG_TYPE	A message type identifier supplied by the client when the message was sent.
JMS_MSG_DELIVE	RIMMDENF veryMode value specified for this message.
JMS_MSG_EXPIRA	THEN time the message expires, which is the sum of the time-to-live value specified by the client and the GMT at the time of the send.
JMS_MSG_PRIORIT	Y he JMS API defines ten levels of priority value ($0 = $ lowest, $9 =$ highest). In addition, clients should consider priorities 0-4 as gradations of normal priority and priorities 5-9 as gradations of expedited priority.
JMS_MSG_REDELI	VEREDIF this message is being redelivered.
JMS_MSG_TIMEST	AIMP time a message was handed off to a provider to be sent. It is not the time the message was actually transmitted, because the actual send may occur later due to transactions or other client-side queueing of messages.

Note that all values in the "Properties" structure are stored as a String type – however they are numbers or text.

For backwards compatibility, all listed properties can also be accessed using lower-case keys; however, it is a deprecated approach.

Table 32.5. "Data" elements

Key	Description
JMS_DATA_MSG	An instance of javax.jms.Message.
JMS_DATA_STREA	M n instance of java.io.InputStream. Accessible only for TextMessage, BytesMessage, StreamMessage, ObjectMessage (only if a payload object is an instance of String). Strings are encoded in UTF-8.
JMS_DATA_TEXT	An instance of String. Only for TextMessage and ObjectMessage, where a payload object is an instance of String.
JMS_DATA_OBJEC	TAn instance of java.lang.Object - message payload. Only for ObjectMessage.

The **Data** container is passed to a task that can use it, depending on its implementation. For example, the task **execute graph** passes it to the executed graph as dictionary entries.

In a Cluster environment, you can explicitly specify node IDs, which can execute the task. However, if the data payload is not serializable and the receiving and executing node differ, an error will be thrown as the Cluster cannot pass the data to the executing node.

Inside a graph or a jobflow, data passed as dictionary entries can be used in some component attributes. For example, the **File URL** attribute would look like: "dict:JMS_DATA_STREAM:discrete" for reading the data directly from the incoming JMS message using a proxy stream.



Note

If the graph is **executed on Worker**, the dictionary entries must be **serialized**; otherwise, they cannot be passed to the graph.

For backwards compatibility, all listed dictionary entries can also be accessed using lower-case keys; however, it is a deprecated approach.

Universal Event Listeners

Since 2.10

Universal Event Listeners allow you to write a piece of Groovy code that controls when an event is triggered, subsequently executing a predefined task. The Groovy code is periodically executed and when it returns TRUE, the task is executed.

Table 32.6. Attributes of U	Iniversal message task
-----------------------------	------------------------

Attribute	Description
Node IDs to handle the event	In a Cluster environment, each universal event listener has a Node IDs attribute which may be used to specify which cluster node performs the Groovy code. There are following possibilities:
	• No failover: Just one node ID specified - Only the specified node performs the Groovy code, however node status must be "ready". When the node isn't ready, the code isn't performed at all.
	• Failover with node concurrency: No node ID specified (empty input) - All cluster nodes with the status "ready" concurrently perform the Groovy code. So the code is executed on each node in the specified interval.
	• Failover with node reservation: More node IDs specified (separated by a comma) - Just one of the specified nodes performs the Groovy code. If the node fails for any reason (or its status isn't "ready"), any other "ready" node from the list continues with periodical Groovy code processing.
	In a standalone environment, the Node IDs attribute is ignored.
Interval of check in seconds	Periodicity of Groovy code execution.
Groovy code	Groovy code that evaluates either to TRUE (execute the task) or FALSE (no action). See below for more details.

Groovy code

A piece of Groovy is repeatedly executed and evaluated; based on the result, the event is either triggered and the task executed or no action is taken.

For example, you can continually check for essential data sources before starting a graph. Or, you can do complex checks of a running graph and, for example, decide to kill it if necessary. You can even call the **CloverDX Server** Core functions using the ServerFacade interface, see Javadoc: http://host:port/clover/javadoc/index.html

Evaluation Criteria

If the Groovy code returns Boolean. TRUE, the event is triggered and the associated task is executed. Otherwise, nothing happens.

If the Groovy code throws an exception, it is considered a coding error and the event is NOT triggered. Thus, exceptions should be properly handled in the Groovy code.

Туре	Key	Description
java.util.Properties	properties	An empty container which may be filled with String-String key-value pairs in your Groovy code. It is passed to the task which may use them somehow. For example, the task execute graph passes these parameters to the executed graph.
java.util.Map <string, Object></string, 	data	An empty container which may be filled with String-Object key-value pairs in your Groovy code. It is passed to the task which may use them somehow according to its implementation - e.g. the task execute graph passes it to the executed graph as dictionary entries. Note that it is not serializable, thus if the task is relying on it, it can be processed properly only on the same cluster node.
javax.servlet.ServletC	ometextetCor	tAxt instance of ServletContext in which CloverDX Server is running.
com.cloveretl.server.a	use Sær Fær	udende nstance of serverFacade usable for calling CloverDX Server core features.
java.lang.String	sessionTol	kensessionToken needed for calling methods on the serverFacade.

Table 32.7. Variables accessible in Groovy code

File Event Listeners (remote and local)

Local file-system changes: Since 1.3

Remote file-system changes: Since 4.2

File Event Listeners allow you to monitor changes on a specific local file system path or remote URL – for example, new files appearing in a folder – and react to such an event with a predefined task.

You can either specify an exact file name or use a wildcard or regexp, then set a checking interval in seconds and define a task to process the event.

There is a global minimum check interval that you can change if necessary in the configuration (the clover.event.fileCheckMinInterval property). See Chapter 15, <u>List of Configuration</u> <u>Properties</u> (p. 82).

CHECK LOCATION File system Path	Local file system
File system Path	Local file system /opt/unprocessedLogs Resolved paths krychle-brno: /opt/unprocessedLogs irt-alpha: /opt/unprocessedLogs
Path	/opt/unprocessedLogs Resolved paths krychle-brno: /opt/unprocessedLogs irt-alpha: /opt/unprocessedLogs
	virt-gray: /opt/unprocessedLogs virt-proteus: /opt/unprocessedLogs
l	Validate Accessibility Paste URL Send email on check failure
FILE TO CHECK	
Type of check	File added 🔹
Filename match type	Wildcards filename match -
Filename pattern	".txt
Check every	60 seconds
s	✓ Trigger task when file has not changed for 3 checks (180 seconds) ●
	✓ Ignore empty files
	Trigger one task for multiple changed files ()
TRIGGERED TASK	
Task type	Start a graph 👻
Nodes to process	Any node One of selected nodes
Sandbox	default 👻

Figure 32.6. Web GUI - creating a File Event listener

Parameter	Description
EVENT_USERNAM	EThe name of the user who caused the event.
EVENT_USER_ID	A numeric ID of the user who caused the event.
EVENT_FILE_NAM	EThe name of the file (without the path or URL) that triggered the event. Present only when Trigger one task for multiple changed files is disabled.
EVENT_FILE_PATH	A resolved (without placeholders) path to the observed directory on the local filesystem. Valid only for a local file listener.
EVENT_FILE_PATT	ERINename pattern.
EVENT_FILE_EVEN	That WPE of the file event. Possible values: SIZE, CHANGE_TIME, APPEARANCE, DISAPPEARANCE.
EVENT_FILE_LIST	ENTERD 165 the listener which triggered the event.
EVENT_FILE_URLS	Full URLs to access the files, e.g. in the File URL attribute of components. If Trigger one task for multiple changed files is enabled and there are multiple URLs, they are separated by a separator specified by CloverDX Engine property DEFAULT_PATH_SEPARATOR_REGEX.
EVENT_FILE_AUT	I_USERNAME A username/ID to the remote location.
EVENT_FILE_AUT	H_USERNAME_URL_ENCODED The same as EVENT_FILE_AUTH_USERNAME, but the value is also URL encoded, so it may be used in a URL.
EVENT_FILE_AUT	I_PASSWORD a password/key to the remote location. It is encrypted by the master password.
EVENT_FILE_AUT	I_PASSWORD_URL_ENCODED The same as EVENT_FILE_AUTH_PASSWORD, but the value is also URL encoded, so it may be used in a URL.

Table 32.8. Parameters passed from the listener to the task

Cluster environment

In a Cluster environment, each file event listener has a **Node IDs** attribute which may be used to specify which cluster node will perform the checks on its local file system. There are following possibilities:

• **No failover:** Just one node ID specified - Only the specified node observes the local/remote filesystem; however, the node status must be "ready". When the node isn't ready, the file system isn't checked at all.

To create a file event listener with no failover, select **One of selected nodes** in **Initialize by** and select one node from the table below.

• Failover with node concurrency:: No node ID specified (empty input) - All cluster nodes with the status "ready" concurrently check the local/remote filesystem according to file event listener attributes settings. In this mode, when the listener is configured to observe the local filesystem, each cluster node observes its own local file system. So it's useful only when the observed path is properly shared among the cluster nodes. It may behave unpredictably otherwise. On the other hand, when the listener is configured to observe the remote filesystem, listeners running on different cluster nodes may connect to the same remote resource. The nodes use a locking mechanism when accessing the local or remote filesystem, so no conflict between listeners running concurrently on different nodes can occur.

To create file event listener with node cuncurrency, select **Any node** in **Initialize by**.

• Failover with node reservation: More node IDs specified (separated by comma) - Just one of the specified nodes checks its filesystem. If the node fails for any reason (or its status isn't "ready"), any other "ready" node

from the list continues with checking. Please note, that when file event listener is re-initialized on another cluster node, it compares the last directory content detected by the failed node with the current directory content.

To create a file event listener with node reservation, select **One of selected nodes** in **Initialize by** and select more nodes.

In a standalone environment, the Node IDs attribute is ignored.

Supported filesystems and protocols

Local filesystem

The user may specify a path to the directory which the listener shall observe. The listener doesn't read the directory content recursively. The directory must exist.

If the listener can run concurrently on more nodes, the directory must be shared among all these nodes and the directory must exist on all these nodes. In a cluster environment, the directory must exist on each cluster node where the listener may run.

It is recommended to use placeholders to unify the configuration on all nodes. The recommended placeholders are: **CloverDX Server** config property \${sandboxes.home} and JVM system property \${java.io.tmpdir}. It is possible to use any JVM system property or Environment variable.

CHECK LOCATION	
File system	Local file system -
Path	\${sandboxes.home}/default/file.csv
	Resolved path C:\Users\Jan\Desktop\CloverDXServer.5.0.0.Tomcat- 9.0.10/sandboxes/default/file.csv
	Validate Accessibility Paste URL
	Send email on check failure 0

Figure 32.7. File available on local file system

Remote filesystem

The user may specify a URL to the directory which the listener shall observe. The supported protocols are: FTP, S3, SFTP and SMB. Different protocols may use different authentication methods: none, username+password and keystore. The listener doesn't read the directory content recursively. The directory must exist.

CHECK LOCATION	
File system	FTP 🔹
Host	10.212.10.150
Port	21
Path	/data/file.csv
Username	clover
Password	
URL	ftp://clover:*****@10.212.10.150:21/data/file.csv

Figure 32.8. File available on remote location

Currently the subset of the protocols allowed by file-operations is supported:

• FTP - File Transfer Protocol (no authentication or username+password authentication) URL example:

ftp://host:23/observed/path/

• SFTP (SSH/FTP) - SSH File Transfer Protocol (username+private key authentication) URL example:

sftp://host:23/observed/path/

It is recommended to use placeholders to unify the path configuration on all nodes. The recommended placeholders are: **CloverDX Server** config property \${sandboxes.home}, JVM system property \${user.home}. It is possible to use any JVM system property or Environment variable.

• S3 - Amazon S3 Storage (AWS Access Key ID + Secret Access Key authentication) URL example:

s3://s3.amazonaws.com/bucketname/path/

Please specify the AWS Access Key ID as a username and Secret Access Key as a password.

• Microsoft SMB/CIFS Protocol (username+password authentication) URL example:

smb://host/path/

• Microsoft SMBv2/v3 Protocol (username+password authentication) URL example:

smb2://host/path/

Observed file

The local observed file is specified by a directory path and file name pattern.

The remote observed file is specified by a URL, credentials and file name pattern.

The user may specify just one exact file name or file name pattern for observing more matching files in specified directory. If there are more changed files matching the pattern, separated event is triggered for each of these files.

There are three ways how to specify file name pattern of observed file(s):

- <u>Exact match</u> (p. 224)
- Wildcards (p. 224)
- Regular expression (p. 225)

Exact match

You specify the exact name of the observed file.

Wildcards

You can use wildcards common in most operating systems (*, ?, etc.)

- * Matches zero or more instances of any character
- ? Matches one instance of any character
- [. . .] Matches any of characters enclosed by the brackets
- \setminus Escape character

Examples

- *.csv Matches all CSV files
- input_*.csv Matches i.e. input_001.csv, input_9.csv
- input_???.csv Matches i.e. input_001.csv, but does not match input_9.csv

Regular expression

Examples

• (.*?)\.(jpg|jpeg|png|gif)\$ - Matches image files

Notes

- It is strongly recommended to use absolute paths with placeholders. It is possible to use a relative path, but the working directory depends on an application server.
- Use forward slashes as file separators, even on MS Windows. Backslashes might be evaluated as escape sequences.

File Events

For each listener you have to specify event type, which you are interested in.

Please note that since **CloverETL 4.2**, the grouping mode may be enabled for the file listener, so all file changes detected by a single check produce just one 'grouped' file event. Otherwise each single file produces its own event.

There are four types of file events:

- File added (p. 225)
- File removed (p. 225)
- File size changed (p. 225)
- File timestamp changed (p. 225)

File added

Event of this type occurs, when the observed file is created or copied from another location between two checks. Please keep in mind that event of this type occurs immediately when a new file is detected, regardless if it is complete or not. Thus task which may need a complete file is executed when the file is still incomplete. Recommended approach is to save the file to a different location and when it is complete, rename it or move it to an observed location where **CloverDX Server** may detect it. File moving/renaming should be an atomic operation.

An event of this type does not occur when the file has been updated (change of timestamp or size) between two checks. Appearance means that the file didn't exist during the previous check and it exists now, during the current check.

File removed

Event of this type occurs, when observed file is deleted or moved to another location between two checks.

File size changed

Event of this type occurs when the size of the observed file has changed between two checks. Event of this type is never produced when the file is created or removed. The file must exist during both checks.

File timestamp changed

Event of this type occurs, when timestamp of the observed file has changed between two checks. Event of this type is never produced when the file is created or removed. The file must exist during both checks.

Check Interval, Task and Use Cases

- The user may specify the minimum time interval between two checks. Use the **Check every** field to specify the interval in seconds.
- Each listener defines a task which will be processed as a reaction to the file event. All task types and theirs attributes are described in the Scheduling (p. 193) and Graph Event Listeners (p. 206) sections.
 - Graph Execution when a file with input data is accessible.
 - Graph Execution when a file with input data is updated.
 - Graph Execution when a file with generated data is removed and must be recreated.

How to use source of event during task processing

A file(s) which caused the event (considered as a source of the event) may be used during task processing. **CloverDX** graph/jobflow components with the **File URL** attribute (e.g. reader or writer components) may directly use an event source by parameter placeholder: \${EVENT_FILE_URLS}. For another parameters, see <u>Executed</u> by Task Graph Execution by File Event Listener (p. 170).

Note that previous versions used lower-case placeholders. Since version 3.3, placeholders are upper-case, however lower-case still work for backward compatibility.

For **graph execution** task this works only if the graph is not pooled. Thus keep in pool interval must be set to 0 (default value).

Delayed triggering for incomplete files

It is possible to delay task execution for incomplete files. This is useful in cases when the condition to execute the listener's task has been met, but there is still some work that needs to be done on the file, e.g. the whole file needs to be uploaded.

Ignore empty files

If the process creating the file creates an empty file, then switch to different task for several minutes or even hours and finally writes the content, tick **Ignore empty files** checkbox. The task will be triggered only if a non-empty file appears.

Trigger task when file has not changed for n checks

If the file size slowly rises until the file is complete, tick the checkbox **Trigger task when file has not changed**. Then specify the number of additional file size checks that are to be performed on the file. The listener's task will not be triggered until the checks are performed and the file's size stays the same between these checks.

Combination

If you use **Ignore empty files** and **Trigger task when file has not changed for n checks** together, the first one filters out empty files and the latter one the files that are being changed. The task will be triggered only on files that are not empty and that have not changed for the specified number of checks.

Howtos

<u>Create a file event listener listening to changes on local file system</u> (p. 227) <u>Observe file from one cluster node</u> (p. 227) <u>Quickly setup failure notification</u> (p. 227) <u>Quickly enable or disable file event listener</u> (p. 228)

Create a file event listener listening to changes on local file system

This howto shows a way to create a new listener checking appearance of a file (new_invoices.txt) on a local file system (/mnt/sdb2/). The appearance will trigger a graph (graph/checkInvoices.grf) from the **InvoicesProcessing** sandbox.

In **Event Listeners** →**File Event Listeners**, click **New Listener**.

Enter the **Name** of the listener, e.g. **Invoices**.

Enter the **Path** to the directory where files will appear: /mnt/sdb2. You can check that **CloverDX** can access this directory (the directory exists and permissions are set up properly) with the **Validate Accesibility** button.

If the observed directory becomes inaccessible, **CloverDX Server** can send you an email. To do so, tick **Send email on check failure** and enter recipient(s).

The event should be triggered on file appearance - set **Type of check** to **File added**.

Enter the file name new_invoices.txt to Filename pattern.

If the file is created empty, but the content is written after some time, tick **Ignore empty files**. Doing so, the task will be executed after the file contains some data.

If it takes a long time to copy the whole file to the observed position, the **CloverDX Server** can perform several check to ensure that the file to process is not to be changed. Tick **Trigger task when file has not changed for** and enter the number of checks. If you tick **Ignore empty files**, this checks will be performed after the file is not empty.

Choose Sandbox with the graph (InvoicesProcessing) and the graph (graph/checkInvoices.grf).

To save the changes, click on the **Create** button.

Observe file from one cluster node

Create the listener in the same way as on the Server.

Switch Initialize by to One of selected nodes.

Add the particular node(s) from **Available nodes** to **Selected nodes**.

Quickly setup failure notification

To create a notification for when the file event listener fails, check the **Send email on check failure** option. and enter email addresses separated by commas.

	 Send email on check failure ()
То	john.doe@email.com, jane.doe@email

The entered email addresses are remembered and pre-filled the next time the button is pressed. If the popup is closed with invalid email addresses entered, the field is cleared.

When creating the notification, a Task Failure Listener is created with an email task listening to the selected File Event Listener. The first entered email address will be used as the Reply-to(Sender) address. The subject and body of the email is as predefined by the Task Failure template. The trigger limit is set to 5.

Editing failure notification

If there is a Task Failure Listener listening to given File Event Listener then instead of the **Create Notification** button a **Notification Detail** button is displayed. This button redirects to the Task Failure Listener page and shows the details of the Task Failure Listener listening to the File Event Listener. If more than one Task Failure Listeners are listening to the File Event Listener, then the details of the first one is shown.

Quickly enable or disable file event listener

In **Event Listeners** \rightarrow **File Event Listeners**, there is a table with event listeners. In this table, click the icon in the **Enabled** column.

Pasting URL

The whole URL including user name and password can be pasted at once. Click **Paste URL** and paste the string.

If the name or password in URL contain special characters, e.g. +, the special characters should be encoded: ftp://anonymous:test%2B@example.com/dir/file.txt

Note: use encoding accepted by the java.net.URLDecoder.decode() function.

Task Failure Listeners

Task Choice (p. 229) Task Failed E-mail Template (p. 230)

Since 4.4

Task Failure Listeners allow you to detect and react to failures in your server when a task you set up in a listener fails to execute, e.g. a File Listener is set up to detect changes on an FTP server, but it fails to connect to the FTP server.

Task Failure Listeners do not detect failures of the task itself, e.g. a File Listener is set up to detect changes on an FTP server and send an email if the change is detected. If the File Listener fails to send the email for some reason, the Task Failure Listener won't detect it.

The same tasks to be executed are available as with all the other listeners, the difference is that when creating a new Task Failure Listener the pre-selected task is **Sending an email** if the email service is configured in Configuration.

Task failure list a file listener is server.	teners allow you to detect and react to failures of a listener's check, e.g. when a set up to detect changes on an FTP server, but it fails to connect to the FTP	
Name	TaskFailureListener	
Owner	clover 💌	
	 Enabled 	
LISTENER TO CHECK		
Listener type	File Event Listener 👻	
Listener	FileEventListener -	
	Listens to: File added: iAmAFile.txt Action: Execute Groovy code	
Trigger limit	5	
TRIGGERED TASK		
Task type	Start a graph	
Nodes to process	Any node One of selected nodes	
Sandbox	default 👻	
Graph	graph/graphMergeData.grf 🔹	
	Save execution history	
Parameters	Name Value	

Figure 32.9. Web GUI - creating a Task Failure listener

Task Choice

There are three options to choose from: Listening to any task failure, listening to failures from a group (such as File Event Listeners) or listening to a failure of a chosen listener.

Selecting an option from the **Listen to type** menu restricts the **Listen to** combobox to event sources of the chosen category. If there are no event sources in the category, the Failure Listener can still be created, it will react to failures of tasks that will be created in that category.

When selecting an event source, you can type into the text field to filter the dropdown menu. After selecting an event source, some information is presented about the listener.

When sending an email, you can use the 'task failed' template by selecting it from the **E-mail template** dropdown menu.

Task Failed E-mail Template

The default email template may be modified using placeholders described in <u>Placeholders</u> (p. 174) and parameters in Table 32.9, "<u>Parameters usable in task failure email template</u>" (p. 230). Furthermore, some additional parameters can be used if the failed listener is a File Event Listener, see Table 32.10, "<u>File Event Listener specific parameters usable in task failure email template</u>" (p. 230).

Table 32.9. Parameters usable in task failure email template

Parameter	Description
TASK_LISTENER_ID	The ID of the failed listener.
TASK_LISTENER_NAMI	The name of the failed listener.
TASK_LISTENER_TYPE	The type of the failed listener.
TASK_LISTENER_TYPE	The Xfull name of the failed listener's type.
TASK_LISTENER_OWN	ERALISEERIN AIMHE failed listener.

Table 32.10. File Event Listener specific parameters usable in task failure email template

Parameter	Description
FILE_EVENT_LISTENER	IFie FatPlACTHe observed directory.
FILE_EVENT_LISTENER	THE FilREMON FIRING the observed directory.
FILE_EVENT_LISTENER	IFILE EleNANA Farthar files the listener observes.
FILE_EVENT_LISTENER	THE Fype HEAK CATTE Estener performs.
FILE_EVENT_LISTENER	IFALEstonAfECHefaYfREmatch type.
FILE_EVENT_LISTENER	THEN of nodes the listener can be intitalized by.

Chapter 33. Recommendations for Transformations Developers

Add external libraries to app-server classpath

Connections (JDBC/JMS) may require third-party libraries. We strongly recommend adding these libraries to the app-server classpath.

CloverDX allows you to specify these libraries directly in a graph definition so that **CloverDX** can load these libraries dynamically. However, external libraries may cause memory leak, resulting in java.lang.OutOfMemoryError: PermGen space, in this case.

In addition, app-servers should have the JMS API on their classpath – and the third-party libraries often bundle this API as well. So it may result in classloading conflicts if these libraries are not loaded by the same classloader.

Another graphs executed by RunGraph component may be executed only in the same JVM instance

In the server environment, all graphs are executed in the same VM instance. The attribute **same instance** of the **RunGraph** component cannot be set to false.

Chapter 34. Extensibility - CloverDX Engine Plugins

Since 3.1.2

The **CloverDX Server** can use external engine plugins loaded from a specified source. The source is specified by engine.plugins.additional.src config property.

See details about the possibilities with CloverDX configuration in Part III, "Configuration" (p. 47)

This property must be the absolute path to the directory or zip file with additional **CloverDX** engine plugins. Both the directory and zip file must contain a subdirectory for each plugin. These plugins are not a substitute for plugins packed in a WAR file. Changes in the directory or the ZIP file apply only when the Server is restarted.

Each plugin has its own class-loader that uses a parent-first strategy by default. The parent of plugins' classloaders is web-app classloader (content of [WAR]/WEB-INF/lib). If the plugin uses any third-party libraries, there may be some conflict with libraries on the parent-classloaders classpath. These are common exceptions/errors suggesting that there is something wrong with classloading:

- java.lang.ClassCastException
- java.lang.ClassNotFoundException
- java.lang.NoClassDefFoundError
- java.lang.LinkageError

There are sevral ways you can get rid of such conflicts:

- Remove your conflicting third-party libraries and use libraries on parent classloaders (web-app or app-server classloaders)
- Use a different class-loading strategy for your plugin.
 - In the plugin descriptor plugin.xml, set attribute greedyClassLoader="true" in the element plugin
 - It means that the plugin classloader will use a self-first strategy
- Set an inverse class-loading strategy for selected Java packages.
 - In the plugin descriptor plugin.xml, set attribute excludedPackages in the element plugin.
 - It is a comma-separated list of package prefixes for example: excludedPackages="some.java.package,some.another.package"
 - In the previous example, all classes from some.java.package, some.another.package and all their sub-packages would be loaded with the inverse loading strategy, then the rest of classes on the plugins classpath.

The suggestions above may be combined. Finding the best solution for these conflicts may depend on the libraries on app-server classpath.

For more convenient debugging, it is useful to set a TRACE log level for related class-loaders.

See Chapter 17, Logging (p. 104) for details about overriding a server log4j configuration.

Chapter 35. Troubleshooting

Graph hangs and is un-killable

A graph can sometimes hang and be un-killable if some network connection in it hangs. Set a shorter tcp-keepalive so that the connection times out earlier. The default value on Linux is 2 hours (7,200 seconds). You can set it to 10 minutes (600 seconds).

See Using TCP keepalive under Linux.

The file descriptor can be closed manually using gdb. See How to close file descriptor via Linux shell command.

SSL/TLS Issues

SSL-related Failures on WebLogic 12

Certain graphs using SSL-encrypted connections may fail on WebLogic 12 due to damaged library distributed with this application server. The issue can be identified by a SHA-1 digest error in the graph execution stacktrace:

```
Caused by: java.io.IOException: Could not convert socket to TLS
    at com.sun.mail.pop3.Protocol.stls(Protocol.java:659)
    at com.sun.mail.pop3.POP3Store.getPort(POP3Store.java:269)
    at com.sun.mail.pop3.POP3Store.protocolConnect(POP3Store.java:207)
Caused by: javax.net.ssl.SSLException: java.lang.SecurityException:
    SHA1 digest error for org/bouncycastle/jce/provider/JCEECPublicKey.class
...
```

To fix the issue, replace the library [MW_HOME]/oracle_common/modules/bcprov-jdk16-1.45.jar with the one downloaded directly from Bouncy Castle home page. Restart the application server to load the new library.

Graph run in Worker is Slow

It may be caused by slow data storage. Use vmstat, e.g. vmstat 1 30. If you see high values under io/bi or io/bo columns, it might be that case. Another tool to confirm or disconfirm slow data storage as possible cause is iotop.

Part VI. API

Chapter 36. Simple HTTP API

The Simple HTTP API is a basic Server automation tool that lets you control the Server from external applications using simple HTTP calls.

Most of operations is accessible using the HTTP GET method and return plain text. Thus, both request and response can be conveniently sent and parsed using very simple tools (wget, grep, etc.).

If global security is on (on by default), the Basic HTTP authentication is used. Authenticated operations will require valid user credentials with corresponding permissions.

Note that the Graph-related operations graph_run, graph_status and graph_kill also work for jobflows and Data Profiler jobs.

The generic pattern for a request URL:

```
http://[domain]:[port]/[context]/[servlet]/[operation]?[param1]=[value1]&[param2]=[value2]...
```

example: http://localhost:8080/clover/simpleHttpApi/help



Note

For backward compatibility, you can also use http://localhost:8080/ clover/request_processor/help.

CSRF Protection

The Simple HTTP API provides protection against Cross-Site Request Forgery (CSRF) attacks. An example of such an attack is a case where the user is logged into the Server Console, and an attacker sends him a link to the Simple HTTP API such that it runs a graph. Clicking on such a link would call the Simple HTTP API and reuse the session of the logged-in user. There are also more complex variants of the attack that are harder to detect by the user.

The protection against such an attack is that the Simple HTTP API requires the presence of the X-Requested-By header in the HTTP request. Value of the header can be arbitrary (it is not checked). Such a header cannot be set by CSRF attack vectors, i.e. by clicking on a link in an email.

Examples of calling the API with the X-Requested-By header:

curl --header "X-Requested-By: arbitrary_value" http://user:password@hostname:port/clover/simpleHttpApi/graph_run

wget --header "X-Requested-By: arbitrary_value" --user=\$USER --password=\$PASS -0 ./\$OUTPUT_FILE \$REQUEST_URL

The CSRF protection of Simple HTTP API can be disabled via the security.csrf.protection.enabled (p. 85) configuration property. It is enabled by default. If the protection is disabled, it is not necessary to set the X-Requested-By header.

The Server Console's page for testing the Simple HTTP API uses a different CSRF protection mechanism. The requests contain a csrftoken parameter. This is intended for usage only in the testing page.

List of Operations

• <u>Operation help</u> (p. 236)

- Operation graph_run (p. 236)
- Operation graph status (p. 237)
- Operation graph_kill (p. 238)
- <u>Operation server_jobs</u> (p. 239)
- Operation sandbox list (p. 239)
- Operation sandbox_content (p. 239)
- Operation executions history (p. 239)
- Operation suspend (p. 241)
- Operation resume (p. 241)
- Operation sandbox_create (p. 242)
- <u>Operation sandbox_add_location</u> (p. 242)
- Operation sandbox_remove_location (p. 242)
- Operation download_sandbox_zip (p. 243)
- <u>Operation upload_sandbox_zip</u> (p. 243)
- Operation cluster_status (p. 244)
- <u>Operation export_server_config</u> (p. 244)
- <u>Operation import_server_config</u> (p. 245)

The HTTP API is enabled by default. You can disable it with the configuration property http.api.enabled. In the Server GUI, switch to **Configuration** \rightarrow **Setup** and add the following line

http.api.enabled=false

to the properties file.

Operation help

parameters

no

returns

a list of possible operations and parameters with its descriptions

example

```
http://localhost:8080/clover/simpleHttpApi/help
```

Operation graph_run

Call this operation to start an execution of the specified job. The operation is called graph_run for backward compatibility, however it may execute a graph, jobflow or profiler job.

parameters

Name	Mandatory	Default	Description	
graphID	yes	-	A file path to the job file, relative to the sandbox root.	
sandbox	yes	-	Text ID of sandbox.	
additional job parameters	no		Any URL parameter with the param_ prefix is passed to the executed job and may be used in transformation XML as a placeholder, but without the param_ prefix. e.g. param_FILE_NAME specified in URL may be used in the XML as \${FILE_NAME}. These parameters are resolved only during loading of XML, so it cannot be pooled.	
additional config parameters	no		URL parameters prefixed with config_ can set some of the execution parameters. For graphs, the following parameters are supported:	
			• config_skipCheckConfig - when set to false, graph configuration will be checked before the execution.	
			• config_logLevel - log level of the executed graph, one of OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL.	
			• config_clearObsoleteTempFiles - when set to true, temp files of previous runs of this graph will be deleted before the execution.	
			 config_debugMode - when set to true, debug mode for a given graph will be enabled. For more information, see <u>Job</u> <u>Config Properties</u> (p. 151). 	
nodeID	no	-	In cluster mode, it is the ID of a node which should execute the job. However it is not final. If the graph is distributed or the node is disconnected, the graph may be executed on another node.	
verbose	no	MESSAGE	MESSAGE FULL - how verbose should possible error message be.	

Table 36.1. Parameters of graph_run

returns

run ID: incremental number, which identifies each execution request

example

 $\tt http://localhost:8080/clover/simpleHttpApi/graph_run?graphID=graph/graphDBExecute.grf&sandbox=mvailered and the second statement of the second sta$

Operation graph_status

Call this operation to obtain a status of a specified job execution. The operation is called graph_status for backward compatibility; however, it may return status of a graph or jobflow.

parameters

Name	Mandatory	Default	Description
runID	yes	-	Id of each graph execution
returnType	no	STATUS	STATUS STATUS_TEXT DESCRIPTION DESCRIPTION_XML
waitForStatus	no	-	Status code which we want to wait for. If it is specified, this operation will wait until the graph is in the required status.
waitTimeout	no	0	If waitForStatus is specified, it will wait only for the specified amount of milliseconds. Default 0 means forever, but it depends on an application server configuration. When the specified timeout expires and graph run still isn't in a required status, the server returns code 408 (Request Timeout). 408 code may be also returned by an application server if its HTTP request timeout expires before.
verbose	no	MESSAGE	MESSAGE FULL - how verbose should possible error message be.

Table 36.2. Parameters of graph_status

returns

Status of a specified graph. It may be a number code, text code or a complex description in dependence on the optional parameter returnType. Description is returned as a plain text with a pipe as a separator, or as XML. A schema describing XML format of the XML response is accessible on **CloverDX Server** URL: http://[host]:[port]/clover/schemas/executions.xsd Depending on the waitForStatus parameter, it may return a result immediately or wait for a specified status.

example

Operation graph_kill

Call this operation to abort/kill a job execution. The operation is called graph_kill for backward compatibility, however it may abort/kill a graph, jobflow or profiler job.

parameters

Table 36.3.	Parameters	of graph_	kill
-------------	------------	-----------	------

Name	Mandatory	Default	Description
runID	yes	-	The ID of each graph execution
returnType	no	STATUS	STATUS STATUS_TEXT DESCRIPTION
verbose	no	MESSAGE	MESSAGE FULL - how verbose should possible error message be.

returns

The status of the specified graph after an attempt to kill it. It may be a number code, text code or a complex description in dependence on optional parameter.

example

http://localhost:8080/clover/simpleHttpApi/graph_kill?runID=123456&returnType=DESCRIPTION

Operation server_jobs

parameters

no

returns

a list of runIDs of currently running jobs.

example

http://localhost:8080/clover/simpleHttpApi/server_jobs

Operation sandbox_list

parameters

no

returns

List of all sandbox text IDs. In the next versions, it will return only accessible ones.

example

```
http://localhost:8080/clover/simpleHttpApi/sandbox_list
```

Operation sandbox_content

parameters

Table 36.4. Parameters of sandbox_content

Name	Mandatory	Default	Description
sandbox	yes	-	text ID of sandbox
verbose	no	MESSAGE	MESSAGE FULL - how verbose should possible error message be.

returns

A list of all elements in the specified sandbox. Each element may be specified as a file path relative to the sandbox root.

example

http://localhost:8080/clover/simpleHttpApi/sandbox_content?sandbox=mva

Operation executions_history

parameters

Name	Mandatory	Default	Description
sandbox	yes	-	The text ID of a sandbox.
from	no		Lower datetime limit of start of execution. The operation will return only records after (and equal to) this datetime. Format: "yyyy-MM-dd HH:mm" (must be URL encoded).
to	no		The upper datetime limit of start of execution. The operation will return only records before (and equal to) this datetime. Format: "yyyy-MM-dd HH:mm" (must be URL encoded).
stopFrom	no		The lower datetime limit of stop of execution. The operation will return only records after (and equal to) this datetime. Format: "yyyy-MM-dd HH:mm" (must be URL encoded).
stopTo	no		The upper datetime limit of stop of execution. The operation will return only records before (and equal to) this datetime. Format: "yyyy-MM-dd HH:mm" (must be URL encoded).
status	no		Current execution status. The operation will return only records with specified STATUS. The values are RUNNING ABORTED FINISHED_OK ERROR
sandbox	no		Sandbox code. The operation will return only records for graphs from a specified sandbox.
graphId	no		The text Id, which is unique in a specified sandbox. The file path is relative to the sandbox root.
orderBy	no		An attribute for list ordering. Possible values: id graphId status startTime stopTime. By default, there is no ordering.
orderDescend	no	true	A switch which specifies ascending or descending ordering. If true (default), ordering is descending.
returnType	no	IDs	Possible values are: IDs DESCRIPTION DESCRIPTION_XML
index	no	0	an index of the first returned records in a whole record set. (starting from
records	no	infinite	The maximum amount of returned records.
verbose	no	MESSAGE	MESSAGE FULL - how verbose should possible error message be.

Table 36.5. Parameters of executions_history

returns

List of executions according to filter criteria.

For returnType==IDs returns a simple list of runIDs (with new line delimiter).

For returnType==DESCRIPTION returns complex response which describes current status of selected executions, their phases, nodes and ports.

```
execution | [runID] | [status] | [username] | [sandbox] | [graphID] | [startedDatetime] | [finishedDatetime] | [clusterNode] | [grap
phase | [index] | [execTimeInMilis]
node | [nodeID] | [status] | [totalCpuTime] | [totalUserTime] | [cpuUsage] | [peakCpuUsage] | [userUsage] | [peakUserUsage]
port | [portType] | [index] | [avgBytes] | [avgRows] | [peakBytes] | [peakRows] | [totalBytes] | [totalRows]
```

example of request

http://localhost:8080/clover/simpleHttpApi/executions_history ->

-> ?from=&to=2008-09-16+16%3A40&status=&sandbox=def&graphID=&index=&records=&returnType=DESCRIPTION

example of DESCRIPTION (plain text) response

```
execution | 13108 | FINISHED_OK | clover | def | test.grf | 2008-09-16 11:11:19 | 2008-09-16 11:11:58 | nodeA | 2.4
phase | 0 | 38733
node DATA_GENERATOR1 FINISHED_OK 0 0 0.0 0.0 0.0 0.0 0.0
port |Output |0 |0 |0 |0 |0 |130 |10
node | TRASH0 | FINISHED_OK | 0 | 0 | 0 . 0 | 0 . 0 | 0 . 0 | 0 . 0
port | Input | 0 | 0 | 0 | 5 | 0 | 130 | 10
node SPEED_LIMITER0 | FINISHED_OK | 0 | 0 . 0 | 0 . 0 | 0 . 0 | 0 . 0 | 0 . 0
port | Input | 0 | 0 | 0 | 0 | 0 | 130 | 10
port |Output |0 |0 |0 |5 |0 |130 |10
execution | 13107 | ABORTED | clover | def | test.grf | 2008-09-16 11:11:19 | 2008-09-16 11:11:30
phase | 0 | 11133
node | DATA_GENERATOR1 | FINISHED_OK | 0 | 0 | 0 . 0 | 0 . 0 | 0 . 0 | 0 . 0 | 0 . 0
port|Output|0|0|0|0|0|130|10
node | TRASH0 | RUNNING | 0 | 0 | 0 . 0 | 0 . 0 | 0 . 0 | 0 . 0
port | Input | 0 | 5 | 0 | 5 | 0 | 52 | 4
node SPEED_LIMITER0 RUNNING 0 0 0.0 0.0 0.0 0.0 0.0
port | Input | 0 | 0 | 0 | 0 | 0 | 130 | 10
port | Output | 0 | 5 | 0 | 5 | 0 | 52 | 4
```

For returnType==DESCRIPTION_XML returns a complex data structure describing one or more selected executions in XML format. A schema describing XML format of the XML response is accessible on **CloverDX** Server URL: http://[host]:[port]/clover/schemas/executions.xsd

Operation suspend

Suspends the Server or sandbox (if specified). No graphs may be executed on suspended Server/sandbox.

parameters

Name	Mandatory	Default	Description
sandbox	no	-	The text ID of a sandbox to suspend. If not specified, it suspends the whole Server.
atonce	no		If this param is set to true, running graphs from suspended Server (or just from sandbox) are aborted. Otherwise it can run until it is finished in standard way.

Table 36.6. Parameters of suspend

returns

Result message

Operation resume

parameters

Table 36.7. Parameters of resume

Name	Mandatory	Default	Description
sandbox	no	-	The text Id of a sandbox to resume. If not specified, the Server will be resumed.
verbose	no	MESSAGE	MESSAGE FULL - how verbose should the possible error message be.

returns

Result message

Operation sandbox_create

This operation creates a specified sandbox. If it is a sandbox of "partitioned" or "local" type, it also creates locations by "sandbox_add_location" operation.

parameters

Table 36.8.	Parameters of	of sandbox	create
-------------	---------------	------------	--------

Name	Mandatory	Default	Description
sandbox	yes	-	The text ID of a sandbox to be created.
path	no	-	A path to the sandbox root if the Server is running in a standalone mode.
type	no	shared	Sandbox type: shared partitioned local. For a standalone Server may be left empty, since the default "shared" is used.
createDirs	no	true	Switch whether to create a directory structure of the sandbox (only for a standalone Server or "shared" sandboxes in a cluster environment).
verbose	no	MESSAGE	MESSAGE FULL - how verbose should possible error message be.

returns

Result message

Operation sandbox_add_location

This operation adds a location to the specified sandbox. Can be only used with partitioned or local sandboxes.

parameters

Table 36.9. Parameters of sandbox add location

Name	Mandatory	Default	Description
sandbox	yes	-	A sandbox which we want to add a location to.
nodeId	yes	-	A location attribute - a node which has direct access to the location.
path	yes	-	A location attribute - a path to the location root on the specified node.
location	no	-	A location attribute - a location storage ID. If not specified, a new one will be generated.
verbose	no	MESSAGE	MESSAGE FULL - how verbose should possible error message be.

returns

Result message

Operation sandbox_remove_location

This operation removes a location from the specified sandbox. Only sandboxes of the partitioned or local type can have locations associated.

parameters

Name	Mandatory	Default	Description
sandbox	yes	-	Removes a specified location from its sandbox.
location	yes	-	A location storage ID. If the specified location isn't attached to the specified sandbox, the sandbox won't be changed.
verbose	no	MESSAGE	MESSAGE FULL - how verbose should possible error message be.

Table 36.10. Parameters of sandbox add location

returns

Result message

Operation download_sandbox_zip

This operation downloads the content of a specified sandbox as a ZIP archive.

parameters

Table 36.11. Parameters

Name	Mandatory	Default	Description
sandbox	yes	-	A code of the sandbox to be downloaded.

returns

a content of a specified sandbox as a ZIP archive

example

```
wget --http-user=username --http-password=password http://localhost:8080/clover/simpleHttpApi/dowpload_sandbox_zi
```

Operation upload_sandbox_zip

This operation uploads the content of a ZIP archive into a specified sandbox.

parameters

Table 36.1	2. Parameters
------------	---------------

Name	Mandatory	Default	Description
sandbox	yes	-	A code of the sandbox the ZIP file will be expanded to.
zipFile	yes	-	The ZIP archive file.
overwriteExisting	no	false	If true, the files already present in the sandbox will be overwritten.
deleteMissing	no	false	If true, the files not present in the ZIP file will be deleted from the sandbox.
fileNameEncoding	no	UTF-8	The encoding that was used to store file names in the ZIP archive.

returns

Result message

an example of request (with using curl CLI tool (http://curl.haxx.se/))

```
curl -u username:password -F "overwriteExisting=true"
    -F "zipFile=@/tmp/my-sandbox.zip"
    http://localhost:8080/clover/simpleHttpApi/upload_sandbox_zip
```

Operation cluster_status

This operation displays cluster's nodes list.

parameters

no

returns

A list of Cluster nodes with information in the following format:

```
<Node Name> | <Node HTTP URL> | <System Load Average> | <Node Status>
```

example

node01|http://localhost:8083/clover|0.3|READY

Note: The value of system load average is calculated from the minute preceding the call. If it cannot be obtained, a negative value is returned (may be caused by the calculation's unacceptable performance impact or lack of support by operating system).

Operation export_server_config

This operation exports a current server configuration in XML format.

parameters

Table 36.13. Parameters of server configuration export

Name	Mandatory	Default	Description
include	no	all	Selection of items that will be included in the exported XML file; the parameter may be specified multiple times. Possible values are:
			• all - include items of all types
			• users - include a list of users
			• userGroups - include a list of user groups
			• sandboxes - include a list of sandboxes
			• jobConfigs - include a list of job configuration parameters
			• schedules - include a list of schedules
			• eventListeners - include a list of event listeners
			• launchServices - include a list of launch services (deprecated)
			• tempSpaces - include a list of temporary spaces

returns

Current server configuration as an XML file.

example

```
wget --http-user=username --http-password=password http://localhost:8080/clover/simpleHttpApi/export_server_config
```

Operation import_server_config

This operation imports server configuration.

parameters

Table 36.14.	Parameters	of serv	er confia	uration	import
1 0.010 0011 11	i alamotoro	0, 00, 1	or oornig	anadion	

Name	Mandatory	Default	Description		
xmlFile	yes	-	An XML file with server's configuration.		
dryRun	no	true	If true, a dry run is performed with no actual changes written.		
verbose	no	MESSAGE	B MESSAGE FULL - how verbose should the response be: MESSAGE for a simple message, FULL for a full XML report.		
newOnly	no	false	If true only new items will be imported to the Server; the items already present on the Server will be left untouched.		
include	no	all	Selection of items that will be imported from the XML; the parameter may be specified multiple times. Possible values are:		
			• all - import items of all types		
			• users - import users		
			• userGroups - import user groups		
			• sandboxes - import sandboxes		
			• jobConfigs - import job configuration parameters		
			• schedules - import schedules		
			• eventListeners - import listeners		
			• launchServices - import launch services (deprecated)		
			• tempSpaces - import temporary spaces		

returns

Result message or XML report

an example of request (with using curl CLI tool (http://curl.haxx.se/))

```
curl -u username:password -F "dryRun=true" -F "verbose=FULL"
    -F "xmlFile=@/tmp/clover_configuration_2013-07-10_14-03-23+0200.xml"
    http://localhost:8080/clover/simpleHttpApi/import_server_config
```
Chapter 37. JMX mBean

The CloverDX Server JMX mBean is an API that can be used for monitoring the internal status of the Server.

MBean is registered with the name:

```
com.cloveretl.server.api.jmx:name=cloverServerJmxMBean
```

JMX Configuration

Application's JMX MBeans aren't accessible outside of JVM by default. It needs some changes in an application server configuration to make JMX Beans accessible.

This section describes how to configure a JMX Connector for development and testing. Thus authentication may be disabled. For production deployment, authentication should be enabled. For more information, see for example Password Authentication

Configurations and possible problems:

- How to configure JMX on Apache Tomcat (p. 246)
- <u>How to Configure JMX on WebSphere</u> (p. 247)
- <u>How to Configure JMX on Worker</u> (p. 248)
- Possible Problems (p. 248)

How to configure JMX on Apache Tomcat

Tomcat's JVM must be executed with these parameters:

- 1. -Dcom.sun.management.jmxremote=true
- 2. -Dcom.sun.management.jmxremote.port=8686
- 3. -Dcom.sun.management.jmxremote.ssl=false
- 4. -Dcom.sun.management.jmxremote.authenticate=false
- 5. -Djava.rmi.server.hostname=your.server.domain (necessary only for remote
 JMX connections)

On UNIX like OS set environment variable CATALINA_OPTS i.e. like this:

export CATALINA_OPTS="-Dcom.sun.management.jmxremote=true
 -Dcom.sun.management.jmxremote.port=8686
 -Dcom.sun.management.jmxremote.ssl=false
 -Dcom.sun.management.jmxremote.authenticate=false
 -Djava.rmi.server.hostname=your.server.domain.com"

File TOMCAT_HOME/bin/setenv.sh (if it does not exist, you may create it) or TOMCAT_HOME/bin/ catalina.sh

On Windows each parameter must be set separately:

```
set CATALINA_OPTS=-Dcom.sun.management.jmxremote=true
```

set CATALINA_OPTS=%CATALINA_OPTS% -Dcom.sun.management.jmxremote.port=8686

set CATALINA_OPTS=%CATALINA_OPTS% -Dcom.sun.management.jmxremote.authenticate=false

```
set CATALINA_OPTS=%CATALINA_OPTS% -Dcom.sun.management.jmxremote.ssl=false
set CATALINA_OPTS=%CATALINA_OPTS% -Djava.rmi.server.hostname=your.server.domain
```

File TOMCAT_HOME/bin/setenv.bat (if it does not exist, you may create it) or TOMCAT_HOME/bin/ catalina.bat

With these values, you can use the URL service:jmx:rmi:///jndi/rmi://localhost:8686/ jmxrmi for connection to JMX server of JVM. No user/password is needed

How to Configure JMX on WebSphere

WebSphere does not require any special configuration, but the **CloverDX** MBean is registered with a name that depends on application server configuration:

	allelandhantikada01 Call, Basélanantélat			Class
View: All tasks	en-localiostivodeorCen, Prome-promer			ciose pa
	Application servers			? -
Welcome	Application servers > server1			
Guided Activities	Use this page to configure an application	server. An application server is a server	that provides service	
Servers	required to run enterprise applications.	server an application server is a server	that provides service	
Server Types WebSphere application servers	Runtime Configuration			
WebSphere MQ servers				
	General Properties	Server messaging		
Applications	Process ID	Messaging engines		
Services	4201			
Resources	Cell name	Troubleshooting		
Security	localhostNode01Cell	Disapostis Provides con	vice	
Environment	Node name	E Diagnostic Provider ser	vice	
	localhostNode01	Additional Properties		
System administration	State			
Users and Groups	Started		. I-(
Monitoring and Tuning	Current heap size	= <u>Performance Monitorino</u>	Infrastructure (PM	
Troubleshooting	543 MB	Product Information		
Service integration	Maximum heap size			
	1024 MB			
. 0001	De de			
	Баск			

Figure 37.1. WebSphere configuration

The URL for connecting to JMX server is:

```
service:jmx:iiop://[host]:[port]/jndi/JMXConnector
```

where *host* is the host name you are connecting to and *port* is an RMI port number. If you have a default WebSphere installation, the JNDI port number will likely be 9100, depending on how many servers there are installed on one system and the specific one you want to connect to. To be sure, when starting WebSphere, check the logs for a line similar to this:

0000000a RMIConnectorC A ADMC0026I: The RMI Connector is available at port 9100

You will also need to set on the classpath the following jar files from WebSphere home directory:

runtimes/com.ibm.ws.admin.client_8.5.0.jar runtimes/com.ibm.ws.ejb.thinclient_8.5.0.jar runtimes/com.ibm.ws.orb_8.5.0.jar

How to Configure JMX on Worker

See Additional Diagnostic Tools (p. 164) for details on how to enable JMX on Worker.

- obtain jmxremote_optional-repackaged-5.0.jar you can find it in the cloverdx.server/ web/WEB-INF/lib
- 2. run VisualVM with jmxremote_optional-repackaged-5.0.jar on the classpath:

./visualvm --cp:a ../path/to/file/jmxremote_optional-repackaged-5.0.jar

3. use a URL with the JMXMP protocol, for example: service:jmx:jmxmp://172.22.0.19:10501

For more information, see JMX Monitoring and Management.

Possible Problems

• Default JMX mBean server uses RMI as a transport protocol. Sometimes RMI cannot connect remotely when one of peers uses Java version 1.6. As a solution, simply set these two system properties: -Djava.rmi.server.hostname=[hostname or IP address] - Djava.net.preferIPv4Stack=true

Operations

For details about operations, see the JavaDoc of the MBean interface:

JMX API MBean JavaDoc is accessible in the running **CloverDX Server** instance on URL: http://[host]: [port]/[contextPath]/javadoc-jmx/index.html

Chapter 38. SOAP WebService API

The **CloverDX Server** SOAP Web Service is an advanced API that provides an automation alternative to the Simple HTTP API. While most of the HTTP API operations are available in the SOAP interface too, the SOAP API provides additional operations for manipulating sandboxes, monitoring, etc.

The SOAP API service is accessible on URL:

http://[host]:[port]/clover/webservice

The SOAP API service descriptor is accessible on URL:

http://[host]:[port]/clover/webservice?wsdl

Protocol HTTP can be changed to secured HTTPS based on the web server configuration.

SOAP WS Client

Exposed service is implemented with the most common binding style "document/literal", which is widely supported by libraries in various programming languages.

To create client for this API, only WSDL document (see the URL above) is needed together with some development tools according to your programming language and development environments.

JavaDoc of the WebService interface with all related classes is accessible in a running **CloverDX Server** instance on URL http://[host]:[port]/[contextPath]/javadoc-ws/index.html

If the web server has an HTTPS connector configured, the client must also meet the security requirements according to web server configuration, i.e. client trust + key stores configured properly.

SOAP WS API Authentication/Authorization

Since exposed service is stateless, an authentication "sessionToken" has to be passed as a parameter to each operation. The client can obtain the authentication sessionToken by calling the login operation.

Chapter 39. Launch Services

Launch Services Overview (p. 250) Deploying Graph in Launch Service (p. 251) Using Dictionary in Graph/Jobflow for Launch Services (p. 251) Configuring the Job in CloverDX Server Web GUI (p. 251) Launch Services Authentication (p. 255) Sending the Data to Launch Service (p. 255) Results of the Graph Execution (p. 255)



Important

Since **4.9.0**, the **Launch Services** have been **deprecated**. We recommend to use Data Services (p. 257) instead.

The Launch Services UI is hidden by default. To make it visible, use the <u>launchservices.visible</u> (p. 89) configuration property.

Launch Services allow you to publish a transformation graph or a jobflow as a Web Service. With Launch Services, **CloverDX** transformations can be exposed to provide a request-response based data interface (e.g. searches, complex lookups, etc.) for other application or directly to users.

Launch Services Overview

The architecture of a Launch Service is layered. It follows the basic design of multi-tiered applications utilizing a web browser.

Launch Services let you build a user-friendly form that the user fills in and sends to the **CloverDX Server** for processing.



Figure 39.1. Launch Services and CloverDX Server as web application back-end

Deploying Graph in Launch Service

To prepare a graph for publishing as a Launch Service, keep this in mind during the design process:

- 1. You can define a graph/jobflow listeners to create parameterized calls. Parameters are passed to the graph as Dictionary entries design the graph so that it uses the Dictionary as input/output for parameters (e.g. file names, search terms, etc.).
- 2. The graph will need to be published in the Launch Services section, where you provide the configuration and binding for parameters to dictionary entries.

Using Dictionary in Graph/Jobflow for Launch Services

A graph or a jobflow published as a service usually means that the caller sends request data (parameters or data) and the transformation processes it and returns back the results.

In a Launch Service definition, you can bind service's parameters to Dictionary entries. These need to be predefined in the transformation.

Dictionary is a key-value temporary data interface between the running transformation and the caller. Usually, although not restricted to, Dictionary is used to pass parameters in and out the executed transformation.

For more information about Dictionary, read the Dictionary section in the CloverDX Designer User's Guide.

Passing Files to Launch Services

If a Launch Service is designed to pass an input file to a graph or jobflow, the input dictionary entry has to be of type readable.channel.

Configuring the Job in CloverDX Server Web GUI

Each Launch Service configuration is identified by its name, user and group restriction. You can create several configurations with the same name, which is valid as long as they differ in their user or group restrictions.

User restrictions can then be used to launch different jobs for different users, even though they use the same launch configuration (i.e. name). For example, developers may want to use a debug version of the job, while the end customers will want to use a production job. The user restriction can also be used to prohibit certain users from executing the launch configuration completely.

Similarly, a group restriction can be used to differentiate jobs based on the user's group membership.

If multiple configurations match the current user/group and configuration name, the most specific one is picked. (The user name has a higher priority than the group name.)

Adding New Launch Configuration

Use the **New launch configuration** button on the Launch Services tab to create a new Launch Service.

The name is the identifier for the service and will be used in the service URL. Then, select a sandbox and either a transformation graph or a jobflow that you want to publish.

Name	GetCountrySales		
Description	Get sales by country code		
Sandbox	BasicExamples	•	
Job file	graph/GetCountrySales.grf	•	

Figure 39.2. Creating a new launch configuration

Once you create the new Launch Service, you can set additional attributes like:

- 1. User and group access restrictions and additional configuration options (Edit Configuration)
- 2. Bind Launch Service parameters to Dictionary entries (Edit Parameters)

Overview	Edit parameter	s		
GetCo	ountrySa	Edit	Close	
	Name	GetCountrySales		
	Description	Get sales by country code		
	Group			
	User			
	Sandbox	BasicExamples		
	Job file	graph/GetCountrySales.grf		
S	Save run record	\bigcirc		
Display	error message detail			

Figure 39.3. Overview tab

The Overview tab shows the basic details about the launch configuration. These can be modified in the Edit Configuration tab:

Edit Configuration

Overview	Edit parameters		
GetCo	untrySales	Save	Close
Nam	e GetCountrySales		
Descriptio	n Get sales by country code		
Grou	p 👻		
Use	er 🗸		
Sandbo	X BasicExamples		
Job fil	e graph/GetCountrySales.grf	•	
	 Save run record Display error message detail 		

Figure 39.4. Edit Configuration tab

Editing configurations:

- Name The name (identifier) under which the configuration will be accessible from the web.
- Description The description of the configuration.
- *Group* Restricts the configuration to a specific group of users.
- User Restricts the configuration to a specified user.
- *Sandbox* The **CloverDX** Sandbox where the configuration will be launched.
- Job file Selects the job to run.
- *Save run record* If checked, the details about the launch configuration will be stored in the Execution History. Uncheck this if you need to increase performance storing a run record decreases response times for high frequency calls.
- Display error message detail Check this if you want to get a verbose message in case the launch fails.

Edit Parameters

The **Edit parameters** tab can be used to configure parameter mappings for the launch configuration. The mappings are required for the Launch Service to be able to correctly assign parameters values based on the values sent in the launch request.

Overview	Edit parameters					
				B Sav	ve 🛛	Close
I	Dictionary entry name	C	ountryCode (string)	•		
HTTP req	uest parameter name	C	ountryCode			
HTTP reque	st parameter required					
Pas	s HTTP request body		0			
	Trim parameter value	*				
Er	mpty parameter is null	*				
Paramet	ter value format string			0		
P	arameter value locale				•	0
Date	e parameter time zone			0		
De	efault parameter value					
Pass valu	e as graph parameter		0			

Figure 39.5. Creating new parameter

To add a new parameter binding, click on the **Add parameter** button. Every required graph/jobflow listener property defined by the job needs to be created here.

verview Edit pa	rameters							
Add parameter								
Dictionary entry name	HTTP request parameter name	HTTP request parameter required	Pass HTTP request body	Pass value as graph parameter	Default parameter value			

Figure 39.6. Edit Parameters tab

You can set the following fields for each property:

- Dictionary entry name The name of the Dictionary entry defined in the graph/jobflow that you want to bind.
- *HTTP request parameter name* The name of this property as it will be visible in the published service. This name can be different from **Name**.
- *HTTP request parameter required* If checked, the parameter is mandatory and an error will be reported if it is omitted.
- *Pass HTTP request body* If checked, the request body is set to dictionary entry as a readable channel.

- *Pass value as graph parameter* If checked, the property value will be passed to the job also as a parameter (\${ParameterName} where ParameterName is equal to Name). This lets you use the parameter anywhere in the job definition (not just places that support Dictionary). However, parameters are evaluated during job initialization. Thus, such a job cannot be pooled which decreases performance for high frequency repetitive calls to the service. In this case, consider redesigning the transformation to use Dictionary instead, allowing for pooling.
- Default parameter value The default value applied in case the parameter is omitted in the launch request.

Launch Services Authentication

If you are using Launch Services, you have two ways how to be logged in: using form-based authentication of Server console or HTTP basic authentication of Launch Services.

The form-based authentication of Server console enables the user to create or modify Launch Services. If you are logged in this way, you act as an administrator of Launch Services.

To insert data into the Launch Service form, you should be logged in using the HTTP basic authentication. Follow the link to the Launch Service form and web browser will request your credentials. If you are logged in using HTTP basic authentication, you act as an user of Launch Services forms.

Sending the Data to Launch Service

A launch request can be sent via HTTP GET or POST methods. A launch request is simply a URL which contains the values of all parameters that should be passed to the job. The request URL is composed of several parts:

(You can use a Launch Services test page, accessible from the login screen, to test drive Launch Services.)

[Clover Context]/launch/[Configuration name]?[Parameters]

- [Clover Context] is the URL to the context in which the **CloverDX Server** is running. Usually this is the full URL to the **CloverDX Server** (for example, for **CloverDX Demo Server** this would be http://server-demo.cloverdx.com:8080/clover).
- [Configuration name] is the name of the launch configuration specified when the configuration was created. In our example, this would be set to "mountains" (case-sensitive).
- [Parameters] is the list of parameters the configuration requires as a query string. It is a URL-encoded [RFC 1738] list of name=value pairs separated by the "&" character.

Based on the above, the full URL of a launch request for our example with mountains may look like this: http://example.com:8080/clover/launch/NewMountains?heightMin=4000. In the request above, the value of heightMin property is set to 4000.

Results of the Graph Execution

After the job terminates, the results are sent back to the HTTP client as content of an HTTP response.

Output parameters are defined in the job's Dictionary. Every Dictionary entry marked as "Output" is sent back as a part of the response.

Depending on the number of output parameters, the following output is sent to the HTTP client:

- *No output parameters* Only a summary page is returned. The page contains: when the job was started, when it finished, the user name, and so on. The format of the summary page cannot be customized.
- *One output parameter* In this case, the output is sent to the client as in the body of the HTTP response with its MIME content type defined by the property type in Dictionary.

• *Multiple output parameters* - In this case, each output parameter is sent to the HTTP client as a part of the multipart HTTP response. The content type of the response is either multipart/related or multipart/x-mixed-replace, depending on the HTTP client (the client detection is fully automatic). The multipart/related type is used for browsers based on Microsoft Internet Explorer and the multipart/x-mixed-replace is sent to browsers based on Gecko or Webkit.

Launch requests are recorded in the log files in the directory specified by the launch.log.dir property in the **CloverDX Server** configuration. For each launch configuration, one log file named [Configuration name]#[Launch ID].log is created. For each launch request, this file will contain only one line with following tab-delimited fields:

(If the property launch.log.dir is not specified, log files are created in the temp directory [java.io.tmpdir]/cloverlog/launch where java.io.tmpdir is system property.)

- Launch start time
- Launch end time
- Logged-in user name
- Run ID
- Execution status FINISHED_OK, ERROR or ABORTED
- IP Address of the client
- *User agent* of the HTTP client
- *Query string* passed to the Launch Service (full list of parameters of the current launch)

If the configuration is not valid, the same launch details are saved into the _no_launch_config.log file in the same directory. All unauthenticated requests are saved to the same file as well.

Chapter 40. Data Services

<u>Overview</u> (p. 257) <u>User Interface</u> (p. 258) <u>Using Data Services</u> (p. 269)

Overview

Data Services allow you to deploy a web service. The architecture of the data service is described in the documentation on **Designer**. This section describes the server-side functionality of the Data Services.



Figure 40.1. Data Services

The Data Service can be accessible via:

• HTTP (default)

Requests are accepted via HTTP protocol on the same port as the Server. This is suitable for Data Services that do not require authentication.

• HTTPS

If you need a secure connection, you should configure Data Service to listen on HTTPS: create an HTTPS Connector (p. 266) and use it in one or more Data Service endpoints. This way, you can configure a service listening on HTTPS port without restarting the Server. You can create multiple HTTPS Connectors and use it, for example, per consumer service. One Data Service endpoint can use only one HTTPS Connector.

You can publish the same data service job in multiple configurations. First, publish the job, then add an SSL Connector to it; after that, you can publish the job again with a different configuration. If a job is reconfigured to use an SSL Connector where it conflicts with another job, a failure occurs and the job has to be unpublished and republished.

Data Service can send you a notification (p. 261)in case of failure. You can set the threshold (p. 262)(number of subsequent failures or percentage) and way of notification (in the Server's UI or via email (p. 264)).

To investigate failed requests, you can use history of the particular endpoint(p. 261). Optionally, you can set the Data Service endpoint run to be recorded in Execution History (p. 264).

User Interface

Data Services user interface contains two main tabs: Endpoints (p. 258) and HTTPS Connectors (p. 266).

Endpoints

Endpoints tab consists of useful <u>buttons</u> (p. 258) in the top, <u>list of data services</u> (p. 258) and tabs with configuration of the particular data service.

rublis	h New Job	Q Search End	points	×	!
Enabled	Endpoint		ОК	FAIL	
C	Example - Create new contact - Parameters processing POST PUT /example/contact		1	1	:
	$\label{eq:Example} \begin{array}{l} \mbox{Example - Download contact VCARD - Custom response body serialization} \\ \hline \mbox{GET} & \mbox{/example/contact/vcard/{id}} \end{array}$		2	0	
	Example - Frontend pages - Use static file as response GET /example/html/{filename}		7	0	:
	Example - Import contact using VCARD - Custom parsing of request body POST PUT /example/contact/vcard		0	0	:
	Example - List contacts - Automatic JSON serialization GET /example/contacts		2	0	•••
	Example - Remove contact - Custom response status code DELETE /example/contact/{id}				•••

Figure 40.2. Endpoints

Buttons

In the top of the **Endpoints** tab, there are five control elements.



Figure 40.3. Data Service Endpoints

- 1. Creates a new Data Service job.
- 2. The three-dot-button has the following menu.

Open Endpoint Documentation Catalog displays list of data services.

Unpublish Data Services Examples.

Import Data Services Configuration.

Export Data Services Configuration.

- 3. Link to Data Services example.
- 4. Search function.
- 5. Filter to show all, or only failing and invalid Data Services.

List of Data Services

The Data Services tab contains list of all data services on the Server.

🕞 Publi	sh New Job Example page	Q, Search End	points	×	I
Enabled	Endpoint		ОК	FAIL	
	Example - Create new contact - Parameters processing POST PUT /example/contact	Ø	0	0	
€	Example - Download contact VCARD - Custom response body serialization GET /example/contact/vcard/{id}	6	0	0	:
	Example - Frontend pages - Use static file as response GET /example/html/{filename}				
	Example - Import contact using VCARD - Custom parsing of request body POST PUT /example/contact/vcard		0	0	•
	Example - List contacts - Automatic JSON serialization GET /example/contacts	8	0	0	***
	Example - Remove contact - Custom response status code DELETE /example/contact/{id}		0	0	
	new-restjob POST /new-restjob	A	2	1	

Figure 40.4. List of Data Services

- The button in the left column serves to enable () or disable () the service (e.g. temporary disable due to maintenance). A disabled Data Service returns the HTTP status code 503.
- In the second column, there are **Endpoint title**, **method**(s) and a part of **endpoint URL**.

Icon decorators indicate these endpoint states:

- - The Data Service does not require authentication.
- B The Data Service saves the job execution record in Execution History.
- ▲ The Data Service is marked as failing.
- - The Data Service is available on HTTPS.
- The third and fourth columns contain query statistics.
- The last column contains a menu with Data Service actions Unpublish and Reset Endpoint State.

Detail

The **Detail** tab contains overview of the particular endpoint. To display the **Detail** tab, click the particular line in the list of endpoints.

Detail	Testing and Documentation	State and History	Alerts and Notification	Configuration			
Example - Create new contact - Parameters processing ×							
Method	s PC						
Data Se	rvice URL /ex	ampte/contact					
Endpoir	nt URL						
🔗 htt	tp://localhost:8083/clover/data	a-service/example/con	itact				
Descrip	tion						
Enpoint 999-999	t for creating new contact. Eithe 9-999. E-mail is required and mu	er first name or last na Ist be a valid email ado	me is required. Phone numb dress. e.g. test@test.com	per is required and	must be in correct form:		
	Testing and Documentation		🛓 Swagger Defini	ition			
*	Sandbox DataServicesExamples						
	data-service/create_contact	.rjob			٢		

Figure 40.5. Data Service Detail tab

- Endpoint title is in the top of the tab's pane. It is the endpoint title specified in Designer on the Endpoint Configuration tab.
- Methods indicates endpoint methods.
- Data Service URL the configurable part of Endpoint URL. It can be set from Designer
- Endpoint URL URL of the endpoint. This URL serves the requests.

The **Copy link** (*I*) button copies the link to the clipboard.

- **Description** contain a user-defined description of the Data Service. It can be set in Designer.
- **Testing and Documentation page** links documentation of the endpoint. You can test the service there. This URL can be passed down to consumer of the service. The consumer can use the information from this URL to implement the client system.
- Swagger file allows you to download a swagger file with the definition of the Data Service.
- Sandbox sandbox containing the data service .rjob file.
- **REST job file** is a file name and path relative to the sandbox.

Testing and Documentation

The **Testing and Documentation** tab displays a user-defined documentation to the data service. The testing of the service is accessible under the **Execute** button.

Detail	Testing and Documentation	State and History	Alerts and Notification	Configuration				
GI	GET /example/contact/vcard/{id} Example - Download contact VCARD - Custom response body serialization							
Get co The jol	Get contact details in VCARD form based on contact ID. The job takes one required path parameter: id. The ID of the contact to be served.							
Param	eters							
Name	Description							
id * re	quired The ID of th	ne contact to get.						
string (path)	id - The I	O of the contact to get						
			Execute					
Respo	nses		Execute Response content type	custom	×]			
Respo	nses Descriptior		Execute Response content type	custom	×)			
Respo Code 200	nses Descriptior		Execute Response content type	custom	v			
Respo Code 200 400	nses Descriptior <i>Request</i>	validation faile	Execute Response content type	custom	× .			

Figure 40.6. Data Service - Testing and Documentation tab

If you call the service from the Server UI, the Data Service will save its run record to the Execution History (p. 199).

State and History

The **State and History** shows invocation history of the particular Data Service. It contains a summary of the endpoint state in the top and a list of query details in the bottom. If the job is configured to save a record in execution history, the list also contains link to the **Execution History**.

You can filter records based on the time interval or you can list only the failures.

Here you can reset the state of the data service. For example, the data service endpoint was failing, you fixed it and you would like to be notified if it fails again.

Detail	Testir	ng and Documentation	State and History	Alerts and Notification	Configuration	
End	lpoint stat	te: failing			F	leset Endpoint State
Filt	ter					^
Set	filter to:	all time 🔹	From:	ä	Failures only	
			То:	ė		
					Filte	r Reset Filter
HTTP S	status	Time		Detail		
0	500	2018-09-10 09:41:45		IP: 0:0:0 User: clo	:0:0:0:0:1 ver	ñ
0	200	2018-09-10 09:41:26		IP: 0:0:0 User: clo	:0:0:0:0:1 ver	<u>A</u>
0	200	2018-09-10 09:39:47		IP: 0:0:0 User: clo	:0:0:0:0:1 ver	ñ

Figure 40.7. Data Service - State and History

Alerts and Notification

No failure notification (p. 262)

Failure (p. 262)Threshold specification (p. 262)Any failure (p. 262)Threshold (p. 263)Failure Filtering (p. 263)Failure Notification (p. 263)E-mail Notification (p. 264)

The **Alerts and Notification** tab serves to set the threshold meaning the failure of the Data Service endpoint and way to notify you about it. You can set when the endpoint is marked as failing or disable this notification completely.

No failure notification

The **Never mark endpoint as failing** disables the failure notification in the Server UI. If you set this option and the request to the endpoint fails, there will be no red circle notification. Only the number of failures in activity column in the list of Data Service endpoints will be increased.

Failure

In this context, any response state from 4xx and 5xx range is considered as a failure.

Threshold specification

You can set threshold to

- any failure
- percentage of unsuccessful queries within interval
- fixed number of failures in a row

Any failure

The **Any failure will mark endpoint as failing** option considers the endpoint as failing even if a single failure occurs.

This choice is suitable for infrequently called Data Service endpoints.

Detail	Testing and Documentation	State and History	Alerts and Notification	Configuration
0	Endpoints can be marked as fa it is possible to send an email	iling. This state will be notification for failing	indicated in the endpoints Data Service endpoints.	table and
FAILURE	INDICATION			
Ne	ever mark endpoint as failing			
• An	y failure will mark endpoint as	failing		
Ma	rk endpoint as failing when:			
	Endpoint fails in %	of executions during la	seconds	~
	Endpoint fails time	s in a row		
HTTP s	tatus codes considered as endpo	pint invocation failure	404,500,501,502,504-5	i99 ()
EMAIL N	OTIFICATION			
✓ Se	end an email notification to 🛛 c	omma separated list of	email addresses	Test
SETTING	S SUMMARY			
End	point will be marked as failing a	after any failure.		
		Apply an	d Reset State Disc	ard Changes

Figure 40.8. Data Service - Alerts and Notification

Threshold

The **Mark endpoint as failing when** option sets the endpoint as failing when a threshold is reached. The threshold can be specified as a *percentage of jobs is failing* or as a *number of jobs in row is failing*.

This choice is suitable for frequently called Data Service endpoints.

Detail	Testing and Documentation	State and History	Alerts and Notification	Configuration
0	Endpoints can be marked as fa it is possible to send an email	ailing. This state will be notification for failing	indicated in the endpoints Data Service endpoints.	table and
FAILURE	INDICATION			
Ne	ever mark endpoint as failing			
Ar	ny failure will mark endpoint as	failing		
• Ma	ark endpoint as failing when:			
•	Endpoint fails in 10 % Endpoint fails 5 tim	of executions during la	seconds	Ţ
HTTP S	tatus codes considered as endp	oint invocation failure	404,500,501,502,504-	599 🚯
Setting	end an email notification to	doe.john@email.com		Test
End	point will be marked as failing	after any failure.		
		Apply an	d Reset State Disc	card Changes

Figure 40.9. Data Service - Alerts and Notification

Failure Filtering

You can also select HTTP status codes which should be considered by **CloverDX Server** as an endpoint invocation failure.

Select the HTTP status codes by entering individual codes or ranges of codes separated by commas. By default, codes 404,500,501,502,504-599 are considered as an endpoint invocation failure. Leaving the field blank means that **CloverDX Server** considers all HTTP status codes from the range 400-599 as invocation failure.



Note

By default, **CloverDX Server** does **not** consider HTTP status code 503 as a failure, because the code is returned in the case of invocation of a manually disabled endpoint.

Failure Notification

If the Data Service endpoint fails, it is shown in the list of Data Services. Additionally, the number of failing Data Service endpoints is shown in the main menu.

Ē	Ħ	Schedules	
\$	ŝ	Data Services	0 ~
i	1	Event Listeners	

Figure 40.10. Some Data Service is failing

Email Notification

You can also set an email notification. This email notification works additionally to the notification in the Server UI. It sends an email when the endpoint's state changes to *failing*. An email is also sent if the endpoint was failing and you manually reset the endpoint state.

EMAIL NOTIFICATION		
✓ Send an email notification to do	pe.john@email.com	Test

Figure 40.11. E-mail Notification

With the Test button, you can send a testing email to the addresses of the recipients.

Email notifications requires a working connection to an SMTP server.

Configuration

The **Configuration** tab allows you to disable the endpoint authentication or to enable saving records in Execution History (p. 199).

Detail	Testing and Documentation	State and Hi	istory	Alerts and Notif	ication	Configuration			
HTTP AUTHENTHICATION									
Authentication method () Do not require authentication 🔹									
Run a	0 2	clover 🔹							
SSL/TLS	SECURED TRANSPORT (HTTPS) —								
There	e are no HTTPS Connectors. Th	ey can be create	ed in Dat	a Services > HTTP	S Connec	tors.			
EXECUT	ION HISTORY								
:	Save job execution record in E	xecution History	6						
				Apply	Dis	card Changes			

Figure 40.12. Data Service

The Data Service can be configured to require credentials or not. If the Data Service does not require credentials, the user to run it should be set in its configuration with the **Run As** option.

You can configure the data service to **save job execution record in Execution History**. The saving job execution record has a performance impact. Use this option only for:

- infrequently called endpoints
- endpoints that are not in production environment
- endpoints to be debugged

Catalog of Services

The Catalog of services is a list of data services allowing the user to view the documentation and test the service.

Group Data Services by sandbox							
DataService							
GET	/mountains Get mountains						
GET	/travel						
GET	/weather-now						

Figure 40.13. Global Catalog of Services

The details can be accessed by clicking the header. The first click displays the details, the second one fold the details back.

Group Data Se	rvices by sandbox
DataService	
GET /mo	untains Get mountains
GET /tr	avel
GET /we	ather-now
You should call this ser http://172.22.0.213:808 e.g. http://172.22.0.213:808	vice with 3/clover/data-service/weather-now?location= 3/clover/data-service/weather-now?location=Aberdeen
Parameters	Try it out
Name	Description
location string (query)	undefined
Responses	Response content type application/xml v
Code	Description
200	
400	Request validation failed
500	Job failed

Figure 40.14. Global Catalog of Services

In the Catalog of Services, the end points can be grouped by sandbox or ordered by URL.

Built-in Data Service Examples

CloverDX Server contains built-in set of Data Service examples. The Data Service examples can be published from the **Data Services** tab.



Figure 40.15. Data Services - Publishing the examples

The published examples are displayed among the others in the list of Data Services.

Publish New Job		Q Search End	points	×	1
Enabled	Endpoint		ОК	FAIL	
	Example - Create new contact - Parameters processing POST PUT /example/contact		0	0	
	$\label{eq:Example-Download contact VCARD - Custom response body serialization} $$ GET / example/contact/vcard/{id} $$$		0	0	
	Example - Frontend pages - Use static file as response GET /example/html/{filename}		0	0	
	Example - Import contact using VCARD - Custom parsing of request body POST PUT /example/contact/vcard		0	0	
	Example - List contacts - Automatic JSON serialization GET /example/contacts		0	0	
	Example - Remove contact - Custom response status code DELETE /example/contact/{id}		0	0	

Figure 40.16. Data Services - Published the examples

HTTPS Connectors

The **Data Service** can be accessible via HTTPS. The configuration of HTTPS is in **Data Services** \rightarrow **HTTPS Connectors**.

Create new HTTPS Connector			bigdata-ks		Salt	:	×
Enabled	Name						
	bigdata-ks		Disabled				
	Port: 8101	:	Port	8101			
	Key store. MyProject/cens/service-z0160919.jks		Key store	Sandbox#163845#MvProject:shared/certs/service-20180919.iks			
		Ť.	Irust store				
	js-20180919						
	Port: 8100	:					
	Key store: MyProject/certs/service-20180919.jks						

Figure 40.17. HTTPS Connectors

As a key store, we support the Java key store (.jks) and PKCS 12 key store (.pl2 or .pfx) formats.

As a trust store, we support the Java key store (.jks) format.

On the left hand side, there is a list of available HTTPS Connectors. On the right, there are details of the connector selected from the list. The **New HTTPS Connector** button creates a new HTTPS Connector.

List of HTTPS Connectors

The list of HTTPS Connectors shows available connectors. You can change the order by clicking on **Name** or **Enabled** in the header.

The button in the first column enables or disables the connector. Disabling the connector that is being used by Data Service makes the Data Service invalid.

The middle column shows the connector's name, port, path to the key store and path to the trust store.

The ... button offers an option to delete the connector.

New HTTPS Connector

The **New HTTPS Connector** tab serves to create a new HTTPS Connector that can be used by one or more Data Services. One Data Service can use only one HTTPS Connector.

Create HTTPS Connecto	PL			3
Name	httpscon-js			
	HTTPS Connector enabled			
Port	8447			
Key Store File				
	 Key store is located in sand 	box		
Sandbox	MyProject	•		
Key store	certs/service-20180919.jks	•		
Key store password	•••••			
Key password		0		
Trust Store File				_
	Trust store is located in san	dbox		
Trust store path			0	
Password				
				_

Figure 40.18. HTTPS Connectors

Name	Description
Name	A name of the HTTPS Connector. The name should be unique. It is displayed in the list of HTTPS Connectors on the Endpoint's Configuration tab.
HTTPS Connector enabled	The checkbox enables or disables the HTTPS Connector to listen on the specified port. Stopping an HTTPS Connector that is being used by a Data Service makes the Data Service invalid.
Port	A TCP port used by the HTTPS Connector. The port must not be occupied by another HTTPS Connector or any other program. If the Data Service is Deployed on CloverDX Cluster , it listens on this port on all cluster nodes. If you use a firewall, set it to allow incoming connections to this port. If you use SELinux, it must be configured to allow CloverDX Server to use this TCP port.
Key store is located in sandbox	The checkbox switches between absolute paths to key store and paths relative to the Server sandbox. If selected, Sandbox and Key store items are displayed. Otherwise, you will see Key store path . The recommended way is to store the key stores out of the sandbox.
Sandbox	A sandbox with the key store.
Key store	The key store within the sandbox.
Key store path	An absolute path to the Java key store. You can use environment variables, system properties of JVM and configuration parameters of the Server as a part of the path. Usually, you will use \${sandboxes.home} here.
Key store password	The password to the Java key store.
Key password	The password to the key in the key store.
Trust store is located in sandbox	The checkbox switches between absolute paths to trust store and paths relative to the Server sandbox. If selected, you can enter Sandbox and Trust store options. Otherwise, you will see Trust store path .
Sandbox	The sandbox containing the trust store.
Trust store	The trust store within the sandbox.
Trust store path	An absolute path to the trust store. You can use environment variables, system properties of JVM and configuration parameters of the Server as a part of the path. Usually, you will use \${sandboxes.home} here.
Password	The password to the trust store.

Table 40.1. Attributes in Create HTTPS Connector dialog

To create a data service listening on HTTPS, you need a keystore with a server certificate. You can create one with the following command.

keytool -keystore service.jks -genkey -keyalg rsa -keysize 3072 -alias serverName

As a key store, we support the Java key store (.jks) and PKCS 12 key store (.pl2 or .pfx) formats.

As a trust store, we support the Java key store (.jks) format.

For security reasons, we recommend you to put the keystore outside the Server sandbox.

Using Data Services

Deploying Data Service (p. 269) Publishing and Unpublishing Data Service from Sandbox (p. 270) Publishing Data Service Examples (p. 270) Changing Data Service to Anonymous (p. 270) Running Data Service on HTTPS (p. 271) Running Data Service on HTTPS on Cluster (p. 273) Monitoring Data Service (p. 273) Testing Data Service (p. 273) Testing Data Service (p. 273) Performance Tuning (p. 273) Exporting Data Service Configuration (p. 273) Importing Data Service Configuration (p. 273) Avoiding Premature Marking of Data Service as Failing (p. 274) Looking up Particular Data Service (p. 274) Resetting State of Failing Data Service Endpoint (p. 274)

Deploying Data Service

To deploy **Data Service** from the Server, go to **Data Services** tab, click **Publish Data Service job** and choose a sandbox and .rjob file.

Publish Data Service J	ob	×
Sandbox Data Service job	DataServicesExamples • data-service/get_contacts.rjob • Image: Contacts.rjob •	
	Publish another Publish Cancel	

Figure 40.19. Publishing Data Service job

You can choose between Data Service with or without required authentication. In the latter case, the Data Service will run under the specified account.

Publish Data Service Job)	ж				
Sandbox	DataServicesExamples •					
Data Service job	data-service/get_contacts.rjob	•				
	Require authentication 0					
Run as 🚯	clover -					
	Publish another Publish C	ancel				

Figure 40.20. Publishing Data Service job that does not require authentication

Publishing Multiple Jobs

To deploy multiple jobs, tick the **Publish another** checkbox. After deploying one job, the dialog for publishing Data Service is displayed again to let you enter the next one.

Publishing and Unpublishing Data Service from Sandbox

You can deploy Data Service directly from a sandbox. To do so, you need read access to the sandbox and **List Data Services** and **Manage Data Services** privileges.

In the **Sandboxes** section of the Server GUI, select a data service to be published/unpublished. In the top right corner of the overview, there are options for publishing or unpublishing the data service, as well as editing, showing data service in Execution History, downloading/downloading as ZIP and deleting the data service.

Publishing Data Service Examples

CloverDX Server contains a built-in set of Data Service examples. These examples are not deployed by default.

The Data Service examples can be deployed directly from the **Data Services** tab. If you do not have any Data Service deployed, click the **Publish Data service Examples** link.

Data Service Endpoints	
There are no published Data Service jobs.	
Publish New Job	

Figure 40.21. Publishing Data Service examples

If there is an existing Data Service, the button to publish examples is in the menu accessible under the three-dotbutton.



Figure 40.22. Publishing Data Service examples - II

See Built-in Data Service Examples (p. 265).

Changing Data Service to Anonymous

By default, the Data Service requires a client to send the credentials. To create the Data Service that does not require authentication, switch to the **Configuration** tab in the Data Service **Detail** pane and change **Authentication method** to *Do not require authentication*. The Data service runs with privileges of an existing user; therefore, you should set the *Run as* field to the suitable user. This user should have permissions necessary to run the Data Service.

Detait	Testing and Documentation	State and History	Alerts and Notification	Configuration
HTTP A	UTHENTHICATION			
Authe	entication method ()	Do not require authent	ication 👻	
Run a	1 21	clover 👻		
SSL/TLS	SECURED TRANSPORT (HTTPS)			
There	e are no HTTPS Connectors. The	ey can be created in Da	ata Services > HTTPS Conne	ctors.
EXECUT	ION HISTORY			
EXECUT	ION HISTORYSave job execution record in Ex	ecution History ()		
EXECUT	ION HISTORY	ecution History ()		
EXECUT	ION HISTORY	ecution History 🜖		
EXECUT	ION HISTORY	ecution History 🜒		

Figure 40.23. Configuring Anonymous Data Service

In the list of Data Services, the Data Service that does not require credentials is indicated by unlocked padlock icon.



Figure 40.24. Data Service without authentication

Running Data Service on HTTPS

By default, the Data Service runs on HTTP and you can configure it to run on HTTPS.

To run Data Service on HTTPS, create a new HTTPS Connector.



Figure 40.25. Creating a new Data Service Connector

Enter a name, port, keystore path, and keystore and key passwords.

Create HTTPS Connecto	Dr			3
Name	httpscon-js			
	✓ HTTPS Connector enabled			
Port	8447			
Key Store File				
	 Key store is located in sand 	box		
Sandbox	MyProject	•		
Key store	certs/service-20180919.jks	•		
Key store password	•••••			
Key password		0		
Trust Store File				
	Trust store is located in san	dbox		
Trust store path			8	
Password				
		D C		neel

Figure 40.26. Creating a new Data Service Connector II

In **Data Services** \rightarrow **Endpoints**, select the Data Service to be running on HTTPS and switch to the **Configuration** tab.

Select the HTTPS connector from the combo box and click the **Apply** button. Now, the Data Service runs on HTTPS.

Detail	Testing and Documentati	on State and History	Alerts and Notification	Configuration
Exan	nple - Remove contact	- Custom response st	atus code	×
HTTP A	UTHENTHICATION			
Authe	entication method 🕕	Basic authentication	-	
SSL/TL	S SECURED TRANSPORT (HTT	PS)		
нтте	S connector	js-20180919 (port: 810	0) 👻	
EXECU	TION HISTORY			
	Save job execution record in	Execution History 0		
			Analy	and Changes

Figure 40.27. Using the HTTPS Connector in Data Service endpoint

You can have more independent HTTPS contexts running on one Server. There can be multiple Data Services running on the same HTTPS context.

Running Data Service on HTTPS on Cluster

This case extends the case of <u>Running Data Service on HTTPS</u>(p. 271). Different cluster nodes have different domain names, but the Java key store has to have one certificate. There are two way to solve the problem with certificates.

- Use a wildcard certificate. The key store file should be placed on the shared file system.
- Use different certificates for each cluster node. The keystores with the certificates must be on the same path on all cluster nodes.

Monitoring Data Service

To see the activity of Data Service, use the list of Data Services. There you can see the main overview of data services.

The state of a particular Data Service is on the State and History tab.

Testing Data Service

To test the Data Service, select the Data Service in the list, switch to the **Testing and Documentation** tab and click the **Execute** button. Results appears in the **Responses** part of the tab.

Performance Tuning

To improve performance, do not save job execution records in Execution History (p. 199). To do so, do not tick *Save job execution records in Execution History* on *Configuration* tab.

Exporting Data Service Configuration

You can export the Data Service configuration from **Data Services** \rightarrow **Endpoints** tab. Click the three-dot-button and select **Export Data Services Configuration** from context menu.



Figure 40.28. Data Services - Export

The Data Services configuration will be exported.

You can also export Data Service configuration in **Configuration** \rightarrow **Export** See <u>Server Configuration</u> <u>Export</u> (p. 157).

Importing Data Service Configuration

You can import the Data Service configuration from **Endpoints**. Click the three-dot-button and select **Import Data Services Configuration** from the context menu.



Figure 40.29. Data Services - Import

You can also import the Data Service configuration directly in **Configuration** \rightarrow **Import** See <u>Server Configuration</u> <u>Import</u> (p. 158).

Avoiding Premature Marking of Data Service as Failing

Data Service might prematurely switch to a failing state if the failure indication is set up to switch to a failing state after a given percentage of executions fails in a given time window. E.g. First execution fails.

To avoid this, you can set the minimum number of events necessary to be taken into account when calculating the change of Data Service state. It can be set with the dataservice.failure.ratio.min.record.count configuration property. The default value is 10 executions.

It can be set in **Configuration** \rightarrow **Setup** \rightarrow **Configuration File**. Add a line containing

```
dataservice.failure.ratio.min.record.count=10
```

to the configuration file.

You can set it to any reasonable positive integer. This configuration is valid for all Data Services available on the Server.

See also Chapter 15, List of Configuration Properties (p. 82).

Looking up Particular Data Service

If you have multiple Data Services available, you can search for a specific Data Service:

If you know the endpoint name, you can look it up. Enter the text into the **Search endpoints** field and click the **Refresh** button. The Data Services will be filtered.

The entered text will be searched in the title of the Data Service, in the name of the request method, in the name of .rjob file and in the path that the Data Service uses.

If you would like to see invalid endpoints only, click the **failing and invalid only** icon. The both filters can be combined.

To switch off the filters, click the **Show All** button.

Resetting State of Failing Data Service Endpoint

If the Data Service endpoint is in the failing state and the problem has been fixed, you can reset the endpoint state manually.

To reset the state, open the details of the endpoint, switch to the **Alerts and Notification** tab and click the **Apply and Reset State** button.

If the endpoint has an email address set, a notification email will be sent to this address.

Custom HTTP Headers

Data Services accept a custom HTTP header X-Clover-Save-Run-Record. The possible values of the header are TRUE and FALSE. **CloverDX Server** accepts them case insensitively.

This header overrides the endpoint's configuration to save the run record or not.

Testing and Documentation page now automatically sends the header with value set to true. This means that all invocations from Testing and Documentation page are saved to the Execution History.

Testing Data Service from Designer creates a record in Execution History regardless of it being published or not.

Data Services on Cluster

Data Service jobs can run on Cluster in the same way as they run on **CloverDX Server**. Parallel run of one Data Service job on multiple cluster nodes in not supported.

Part VII. Cluster

Chapter 41. Sandboxes in Cluster

There are three sandbox types in total - shared sandboxes, and partitioned and local sandboxes (introduced in 3.0) which are vital for parallel data processing..

Shared Sandbox

This type of sandbox must be used for all data which is supposed to be accessible on all cluster nodes. This includes all graphs, jobflows, metadata, connections, classes and input/output data for graphs which should support HA. All shared sandboxes reside in the directory, which must be properly shared among all cluster nodes. You can use a suitable sharing/replicating tool according to the operating system and filesystem.

Create new sandbox		ж
Sandbox type	Shared -	
Name	dataFlow	
ID	dataFlow	
Root file system	\${sandboxes.home}/dataFlow	
Paul	Sandbox root path must be valid on all cluster nodes. You can also use system properties (e.g. user.home) or CloverDX Server property "sandboxes.home" as placeholders.	
Create sandbox root path and sub- directories	✓	
	Create	I

Figure 41.1. Dialog form for creating a new shared sandbox

As you can see in the screenshot above, you can specify the root path on the filesystem and you can use placeholders or absolute path. Placeholders available are environment variables, system properties or **CloverDX Server** configuration property intended for this use: sandboxes.home. Default path is set as [user.data.home]/CloverDX/sandboxes/[sandboxID] where the sandboxID is an ID specified by the user. The user.data.home placeholder refers to the home directory of the user running the JVM process (/home subdirectory on Unix-like OS); it is determined as the first writable directory selected from the following values:

- USERPROFILE environment variable on Windows OS
- user.home system property (user home directory)
- user.dir system property (JVM process working directory)
- java.io.tmpdir system property (JVM process temporary directory)

Note that the path must be valid on all cluster nodes. Not just nodes currently connected to the cluster, but also on nodes that may be connected later. Thus when the placeholders are resolved on a node, the path must exist on the node and it must be readable/writable for the JVM process.

Local Sandbox

This sandbox type is intended for data, which is accessible only by certain cluster nodes. It may include massive input/output files. The purpose being, that any cluster node may access content of this type of sandbox, but only one has local (fast) access and this node must be up and running to provide data. The graph may use resources from multiple sandboxes which are physically stored on different nodes since cluster nodes are able to create network streams transparently as if the resources were a local file. For details, see <u>Using a Sandbox Resource as a Component Data Source</u> (p. 279).

Do not use a local sandbox for common project data (graphs, metadata, connections, lookups, properties files, etc.). It would cause odd behavior. Use shared sandboxes instead.

Create new san	ndbox	ж
Sandbox type	Local	
Name	uploadedData	
ID	uploadedData	
	Sandbox locations	
Cluster node ID	Sandbox locations Root path	
Cluster node ID virt-alpha ▼	Sandbox locations Root path \${sandboxes.home.local}/uploadedData	Delete
Cluster node ID virt-alpha ◄	Sandbox locations Root path \${sandboxes.home.locaf}/uploadedData	Delete

Figure 41.2. Dialog form for creating a new local sandbox

The sandbox location path is pre-filled with the sandboxes.home.local placeholder which, by default, points to [user.data.home]/CloverDX/sandboxes-local. The placeholder can be configured as any other CloverDX configuration property.

Partitioned Sandbox

This type of sandbox is an abstract wrapper for physical locations existing typically on different cluster nodes. However, there may be multiple locations on the same node. A partitioned sandbox has two purposes related to parallel data processing:

1. node allocation specification

Locations of a partitioned sandbox define the workers which will run the graph or its parts. Each physical location causes a single worker to run without the need to store any data on its location. In other words, it tells the **CloverDX Server**: to execute this part of the graph in parallel on these nodes.

2. storage for part of the data

During parallel data processing, each physical location contains only part of the data. Typically, input data is split in more input files, so each file is put into a different location and each worker processes its own file.

Create new san	Create new sandbox			
Sandbox type	Partitioned -			
Name	uploadedData			
ID	uploadedData			
	Sandbox locations			
Cluster node ID	Root path			
virt-alpha 👻	\${sandboxes.home.partitioned}/uploadedData	Delete		
virt-gray 👻	\${sandboxes.home.partitioned}/uploadedData	Delete		
	Add location			
	Create	Cancel		

Figure 41.3. Dialog form for creating a new partitioned sandbox

As you can see on the screenshot above, for a partitioned sandbox, you can specify one or more physical locations on different cluster nodes.

The sandbox location path is pre-filled with the sandboxes.home.partitioned placeholder which, by default, points to [user.data.home]/CloverDX/sandboxes-paritioned. The sandboxes.home.partitioned config property may be configured as any other **CloverDX Server** configuration property. Note that the directory must be readable/writable for the user running JVM process.

Do not use a partitioned sandbox for common project data (graphs, metadata, connections, lookups, properties files, etc.). It would cause odd behavior. Use shared sandboxes instead.

Using a Sandbox Resource as a Component Data Source

A sandbox resource, whether it is a shared, local or partitioned sandbox (or ordinary sandbox on standalone server), is specified in the graph under the **fileURL** attributes as a so called sandbox URL like this:

sandbox://data/path/to/file/file.dat

where data is a code for the sandbox and path/to/file/file.dat is the path to the resource from the sandbox root. The URL is evaluated by **CloverDX Server** during job execution and a component (reader or writer) obtains the opened stream from the Server. This may be a stream to a local file or to some other remote resource. Thus, a job does not have to run on the node which has local access to the resource. There may be more sandbox resources used in the job and each of them may be on a different node.

The sandbox URL has a specific use for parallel data processing. When the sandbox URL with the resource in a *partitioned sandbox* is used, that part of the graph/phase runs in parallel, according to the node allocation specified by the list of partitioned sandbox locations. Thus, each worker has its own local sandbox resource. **CloverDX Server** evaluates the sandbox URL on each worker and provides an open stream to a local resource to the component.

The sandbox URL may be used on the standalone Server as well. It is an excellent choice when graph references some resources from different sandboxes. It may be metadata, lookup definition or input/output data. A referenced sandbox must be accessible for the user who executes the graph.

Remote Edges

Data transfer between graphs running on different nodes is performed by a special type of edge - remote edge. The edge utilizes buffers for sending data in fixed-sized chunks. Each chunk has a unique number; therefore, in case of an I/O error, the last chunk sent can be re-requested.

You can set up values for various remote edge parameters via configuration properties. For list of properties, their meaning and default values, see <u>Optional Remote Edge Properties</u> (p. 285).

The following figure shows how nodes in a cluster communicate and transfer data - the client (graph running on Node 2) issues an HTTP request to Node 1 where a servlet accepts the request and checks the status of the source buffer. The source buffer is the buffer filled by the component writing to the left side of the remote edge. If the buffer is full, its content is transmitted to the Node 2, otherwise the servlet waits for configurable time interval for the buffer to become full. If the interval has elapsed without data being ready for download, the servlet finishes the request and Node 2 will re-issue the request at later time. Once the data chunk is downloaded, it is made available via the target buffer for the component reading from the right side of the remote edge. When the target buffer is emptied by the reading component, Node 2 issues new HTTP request to fetch the next data chunk.

This communication protocol and its implementation have consequences for the memory consumption of remote edges. A single remote edge will consume 3 x chunk size (1.5MB by default) of memory on the node that is the source side of the edge and 1 x chunk size (512KB by default) on the node that is the target of the edge. A smaller chunk size will save memory; however, more HTTP requests will be needed to transfer the data and the network latency will lower the throughput. Large data chunks will improve the edge throughput at the cost of higher memory consumption.



Figure 41.4. Remote Edge Implementation

Chapter 42. Cluster Configuration

Cluster can work properly only if each node is properly configured. Clustering must be enabled, nodeID must be unique on each node, all nodes must have access to shared DB (direct connection or proxied by another Cluster node) and shared sandboxes, and all properties for inter-node cooperation must be set according to network environment.

Properties and possible configuration are the following:

- Mandatory Cluster Properties (p. 282)
- Optional Cluster Properties (p. 283)
- Example of 2 Node Cluster Configuration (p. 286)
- Jobs Load Balancing Properties (p. 292)
Mandatory Cluster Properties

Besides mandatory Cluster properties, you need to set other necessary properties which are not specifically related to the Cluster environment. Database connection must be also configured; however, besides direct connection, it is alternatively possible to configure proxing using another Cluster node/nodes. For details, see the property cluster.datasource.type (p. 284).

Mandatory properties

These properties must be properly set on each node of the Cluster.

Table 42.1. l	Mandatory	Cluster	properties
---------------	-----------	---------	------------

Name	Туре	Description	Default	
cluster.enabled	boolean	Switch whether the Server should start in the standalone or Cluster node mode. The property isn't set at all (empty value) by default, which means that the mode is chosen according to the loaded license. It is strongly recommended to set the property to true if the other Cluster properties are configured, as well. Thus the Cluster node will be initialized regardless of the license.		
cluster.node.id	String	Each Cluster node must have a unique ID.	node01	
cluster.jgroups.bind_addres	sString, IP address	An IP address of the ethernet interface which is used for communication with another Cluster nodes. Necessary for inter-node messaging.	127.0.0.1	
cluster.jgroups.start_port	int, port	Port where a jGroups server listens for inter-node messages.	7800	
cluster.http.url	String, URL	A URL of the CloverDX Cluster node. It must be an HTTP/HTTPS URL to the root of a web application. Typically it would be http://[hostname]:[port]/ clover. Primarily, it is used for synchronous inter- node communication from other Cluster nodes. It is recommended to use a fully qualified hostname or IP address, so it is accessible from a client browser or CloverDX Designer.	http:// localhost:8 clover	3080
cluster.jgroups.tcpping.init	a <mark>8_thiog</mark> tsin format: "IPaddress	List of IP addresses (with ports) where we expect running and listening nodes. It is related to other nodes I'[pint1]].port! properties. Necessary for inter-node messaging.	127.0.0.1[7800

Optional Cluster Properties

Optional General Properties (p. 283) Optional Remote Edge Properties (p. 285)

Optional General Properties

These properties are not vital for Cluster configuration - default values are sufficient.

Table 42.2.	Optional	general	properties
-------------	----------	---------	------------

Name	Туре	Description	Default
cluster.jgroups.external_ad	d Setsi ng, IP address	An IP address of the Cluster node. Configure this only if the Cluster nodes are on different sub-nets, so the IP address of the network interface isn't directly accessible from the other Cluster nodes.	
cluster.jgroups.external_po	r í nt, port	A port for asynchronous messaging. Configure this only if the Cluster nodes are on different sub-nets and the port opened on the IP address is different than the port opened on the node's network interface IP address.	
cluster.jgroups.protocol.NA	MACK.gc	Agnumber of delivered messages kept in the sent messages buffer of each jGroups view member. Messages are kept in a sender cache even though they were reported as delivered by existing view members, because there may be some other member temporarily not in the view. The higher the number, the higher the chance of reliable messages delivery in an unreliable network environment. However the messages consume memory: approximately 4kB for each message.	10000
cluster.jgroups.protocol.NA	KACK.xm	iHaw lkomb (in the illine county) time keep obsolete member in the xmit-table. It is necessary for recognition of member temporarily unaccessible and removed from the view. With previous NAKACK implementation, the member removed from the view was also automatically removed from xmit- table, so it appeared as a new member when it re-joined the view. With current modified implementation the member is kept in the xmit-table for a configured interval longer, so when it re-joins the view, it is a known member and undelivered messages may be re-delivered to it. A member in the xmit-table isn't consuming memory.	3600000
cluster.jgroups.protocol.AU	J SHing lue	String used by a jgroups member to authenticate to the group. Must be the same on all Cluster nodes. It is a protection against fake messages.	
sandboxes.home.partitioned	1String	Intended as a placeholder in the location path. So the sandbox path is specified with the placeholder and it is resolved to the real path just before it is used. For backward compatibility, the default value uses the <u>clover.home</u> (p. 82) configuration property.	\${clover.home sandboxes- partitioned
sandboxes.home.local	String	Intended as a placeholder in the location path. So the sandbox path is specified with the placeholder and it is resolved to the real path just before it is used. For backward compatibility, the default value uses the <u>clover.home</u> (p. 82) configuration property.	\${clover.home sandboxes- local

Name	Туре	Description	Default
cluster.shared_sandboxes_	p &tr ing	This property is deprecated. This property still works but is used only when a shared sandbox doesn't have its own path specified. It is just for backward compatibility and it is not recommended for new deployments. Since 3.5, we recommend to specify the sandbox path explicitly and use the sandboxes.home property/placeholder.	
cluster.node.sendinfo.inter	väht	A time interval in milliseconds. Each node sends a heart- beat with information about itself to another nodes. This interval specifies how often the information is sent under common circumstances.	2000
cluster.node.sendinfo.clust	eiimtode.sen	diAfformerin_tentent/ah milliseconds. A specified minimum interval between two heart-beats. A heart- beat may be send more often than specified by cluster.node.sendinfo.interval, e.g. when jobs start or finish. However the interval will never be shorter then this minimum.	500
cluster.node.sendinfo.histo	rýrinterval	A time interval in milliseconds, for which each node stores a heart-beat in the memory. It is used for rendering figures in the web GUI-monitoring section.	240000
cluster.node.remove.interv	alint	A time interval in milliseconds. If no node info comes in this interval, the node is considered as lost and it is removed from the Cluster.	50000
cluster.max_allowed_time_	simift_betw	effinenondesimum allowed time shift between nodes. All nodes must have system time synchronized, otherwise the Cluster may not work properly. So if this threshold is exceeded, the node will be set as invalid.	2000
cluster.group.name	String	Each Cluster has its unique group name. If you need 2 Clusters in the same network environment, each of them would have its own group name.	cloverCluster
cluster.jgroups.protocol.AU	J SHing lue	An authentication string/password used for verification Cluster nodes accessing the group. If this property is not specified, the Cluster should be protected by firewall settings.	
cluster.datasource.type	String	Change this property to remote if the node doesn't have a direct connection to the CloverDX Server database, so it has to use some other Cluster node as proxy to handle persistent operations. In such a case, the cluster.datasource.delegate.nodeIds property must be properly configured, as well. Properties jdbc.* will be ignored. Note that scheduler is active only on nodes with a direct connection.	local
cluster.datasource.delegate	nstatienIgIs	A list of Cluster node IDs (separated by a comma) which this node may use as a proxy to handle persistent operations. At least one of the listed node IDs must be running, otherwise this node will fail. All listed node IDs must have a direct connection to CloverDX Server database properly configured. Property cluster.datasource.delegate.nodeIds is ignored by default. Property cluster.datasource.type must be set to remote to enable the feature.	

Optional Remote Edge Properties

Below is a list of names and default values of properties used to configure remote edges in a Clustered environment.

Table 42.3. Optional remote edge properties

Name	Description	Default
cluster.edge.chunkSize	Specifies the size of a chunk created by the right side of a remote edge (in bytes).	524288
cluster.edge.chunkWaitTimeo	Suppose the server wait for a next chunk to become available (in milliseconds).	60000
cluster.edge.connectTimeout	Specifies a socket connection timeout when fetching a chunk (in milliseconds).	30000
cluster.edge.readTimeout	Specifies a socket read timeout when fetching a chunk (in milliseconds).	90000
cluster.edge.handshakeTimeo	uspecifies how long should the client wait until a remote edge is registered by a data producing job (in milliseconds).	120000
cluster.edge.chunkReadRetrie	Specifies how many times should be a chunk fetch re-attempted before reporting an error to the consumer.	2
cluster.edge.disableChunkPro	t Disd bles the chunked data transfer protocol, switching to the old implementation.	false
cluster.ssl.disableCertificateV	dDistablus validation of certificates in HTTPS connections of remote edges. Disabling the validation affects jobs run on both Worker and Server Core.	false

Example of 2 Node Cluster Configuration

Basic 2-nodes Cluster Configuration (p. 286) 2-nodes Cluster with Proxied Access to Database (p. 287) 2-nodes Cluster with Load Balancer (p. 288)

This section contains examples of **CloverDX** Cluster nodes configuration. We assume that the user "clover" is running the JVM process and the license will be uploaded manually in the web GUI. In addition it is necessary to configure:

- sharing or replication of file system directory which the property "sandboxes.home" is pointing to. E.g. on Unixlike systems it would be typically /home/[username]/CloverDX/sandboxes.
- · connection to the same database from both nodes

Basic 2-nodes Cluster Configuration

This example describes a simple Cluster: each node has a direct connection to a database.



Figure 42.1. Configuration of 2-nodes Cluster, each node has access to a database

Configuration of Node 1 on 192.168.1.131

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://192.168.1.200/clover_db?charSet=UTF-8
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
jdbc.username=clover
jdbc.password=clover
cluster.enabled=true
cluster.node.id=node01
cluster.http.url=http://192.168.1.131:8080/clover
cluster.jgroups.bind_address=192.168.1.131
cluster.jgroups.start_port=7800
cluster.jgroups.tcpping.initial_hosts=192.168.1.132[7800]
cluster.group.name=TheCloverCluster1
sandboxes.home=/home/clover/shared_sandboxes
```

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://192.168.1.200/clover_db?charSet=UTF-8
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
jdbc.username=clover
jdbc.password=clover
```



The configuration is done in a **properties file**. The file can be placed either on a default (p. 51) or specified (p. 50) location.

2-nodes Cluster with Proxied Access to Database

This Cluster configuration is similar to the previous one, but only one node has direct access to a database. The node2 has to use node1 as a proxy.



Figure 42.2. Configuration of 2-nodes Cluster, one node without direct access to database

Configuration of Node 1 on 192.168.1.131

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://192.168.1.200/clover_db?charSet=UTF-8
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
jdbc.username=clover
jdbc.password=clover
cluster.enabled=true
cluster.node.id=node01
cluster.http.url=http://192.168.1.131:8080/clover
cluster.jgroups.bind_address=192.168.1.131
cluster.jgroups.start_port=7800
cluster.group.name=TheCloverCluster2
sandboxes.home=/home/clover/shared_sandboxes
```

```
cluster.datasource.type=remote (1)
cluster.datasource.delegate.nodeIds=node01
cluster.enabled=true
cluster.node.id=node02
cluster.http.url=http://192.168.1.132:8080/clover
cluster.jgroups.bind_address=192.168.1.132
```

```
cluster.jgroups.start_port=7800
cluster.jgroups.tcpping.initial_hosts=192.168.1.131[7800]
cluster.group.name=TheCloverCluster2
sandboxes.home=/home/clover/shared_sandboxes
```

1 These two lines describe access to database via another node.

2-nodes Cluster with Load Balancer

If you use any external load balancer, the configuration of CloverDX Cluster will be same as in the first example.





The cluster.http.url and cluster.jgroups.bind_address are URLs of particular Cluster nodes even if you use a load balancer.

Configuration of Node 1 on 192.168.1.131

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://192.168.1.200/clover_db?charSet=UTF-8
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
jdbc.username=clover
jdbc.password=clover
cluster.enabled=true
cluster.node.id=node01
cluster.http.url=http://192.168.1.131:8080/clover
cluster.jgroups.bind_address=192.168.1.131
cluster.jgroups.start_port=7800
cluster.jgroups.tcpping.initial_hosts=192.168.1.132[7800]
cluster.group.name=TheCloverCluster3
sandboxes.home=/home/clover/shared_sandboxes
```

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://192.168.1.200/clover_db?charSet=UTF-8
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
jdbc.username=clover
jdbc.password=clover
cluster.enabled=true
cluster.node.id=node02
cluster.http.url=http://192.168.1.132:8080/clover
cluster.jgroups.bind_address=192.168.1.132
```

cluster.jgroups.start_port=7800
cluster.jgroups.tcpping.initial_hosts=192.168.1.131[7800]

cluster.group.name=TheCloverCluster3

sandboxes.home=/home/clover/shared_sandboxes

Example of 3 Node Cluster Configuration

Basic 3-nodes Cluster Configuration

This example describes a Cluster with three nodes where each node has a direct connection to a database.





Configuration of Node 1 on 192.168.1.131

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://192.168.1.200/clover_db?charSet=UTF-8
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
jdbc.username=clover
jdbc.password=clover
cluster.enabled=true
cluster.node.id=node01
cluster.http.url=http://192.168.1.131:8080/clover
cluster.jgroups.bind_address=192.168.1.131
cluster.jgroups.start_port=7800
cluster.jgroups.tcpping.initial_hosts=192.168.1.132[7800],192.168.1.133[7800]
cluster.group.name=TheCloverCluster4
sandboxes.home=/home/clover/shared_sandboxes
```

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://192.168.1.200/clover_db?charSet=UTF-8
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
jdbc.username=clover
jdbc.password=clover
cluster.enabled=true
cluster.node.id=node02
cluster.http.url=http://192.168.1.132:8080/clover
cluster.jgroups.bind_address=192.168.1.132
cluster.jgroups.start_port=7800
cluster.jgroups.tcpping.initial_hosts=192.168.1.131[7800],192.168.1.133[7800]
cluster.group.name=TheCloverCluster4
sandboxes.home=/home/clover/shared_sandboxes
```

Configuration of Node 3 on 192.168.1.133

```
jdbc.driverClassName=org.postgresql.Driver
jdbc.url=jdbc:postgresql://192.168.1.200/clover_db?charSet=UTF-8
jdbc.dialect=org.hibernate.dialect.PostgreSQLDialect
jdbc.username=clover
jdbc.password=clover
cluster.enabled=true
cluster.node.id=node03
cluster.http.url=http://192.168.1.133:8080/clover
cluster.jgroups.bind_address=192.168.1.133
cluster.jgroups.start_port=7800
cluster.jgroups.tcpping.initial_hosts=192.168.1.131[7800],192.168.1.132[7800]
cluster.group.name=TheCloverCluster4
```

sandboxes.home=/home/clover/shared_sandboxes

Jobs Load Balancing Properties

Multiplicators of load balancing criteria. A load balancer decides which Cluster node executes the graph. It means, that any node may process a request for execution, but a graph may be executed on the same or on different node according to current load of the nodes and according to these multiplicators.

The higher the number, the higher the relevance for decision. All multiplicators must be greater than 0.

Each node of the Cluster may have different load balancing properties. Any node may process incoming requests for transformation execution and each may apply criteria for loadbalancing in a different way according to its own configuration.

These properties aren't vital for Cluster configuration - default values are sufficient

Name	Туре	Default	Description
cluster.lb.balance.run	nfh <u>gat</u> graph	s 3	Specify importance of running graphs for load balancing.
cluster.lb.balance.mer	nfiload	0.5	Specify importance of used memory for load balancing.
cluster.lb.balance.cpu	sfloat	1.5	Specify importance of number of CPUs for load balancing.
cluster.lb.balance.req	u£kot <u>a</u> bonus	2	Specify importance of the fact that the node is the same which processes the request for execution. The same node which decides where to execute the graph. If you specify this multiplicator great enough, the graph will be always executed on the same node which processes the request for execution.
cluster.lb.balance.nod	ef <u>l</u> bøtnus	1	Overall ratio bonus for configured node. Values greater then "1" increase probability of the node to be chosen by the loadbalancer. Value "1" means no bonus or penalty. "0" means that the node will never be chosen by the loadbalancer; however, it still may execute graphs, e.g. when there is no other node in the cluster or when the graph is designed to run on the node.

Table 42.4. Load balancing properties

Running More Clusters

If you run more clusters, each Cluster has to have its own unique name. If the name is not unique, the Cluster nodes of different clusters may consider foreign Cluster nodes as part of the same cluster. The Cluster name is configured using cluster.group.name option. See <u>Optional Cluster Properties</u> (p. 283).

Chapter 43. Recommendations for Cluster Deployment

- 1. All nodes in the cluster should have a synchronized system date-time.
- 2. All nodes share sandboxes stored on a shared or replicated filesystem. The filesystem shared among all nodes is a single point of failure. Thus, the use of a replicated filesystem is strongly recommended.
- 3. All nodes share a DB, thus it must support transactions. I.e. The MySQL table engine, MyISAM, may cause unusual behavior because it is not transactional.
- 4. All nodes share a DB, which is a single point of failure. Use of a clustered DB is strongly recommended.
- 5. Configure the license by license.file property or upload it in the Web GUI, so it is stored in the database. Do not use clover-license.war.

Chapter 44. Troubleshooting

Cluster Reliability in Unreliable Network Environment

CloverDX Server instances must cooperate with each other to form a Cluster together. If the connection between nodes doesn't work at all, or if it is not configured, Cluster can't work properly. This chapter describes Cluster nodes behavior in an environment where the connection between nodes is somehow unreliable.

Nodes use three channels to exchange status info or data

1. synchronous calls (via HTTP/HTTPS)

Typically NodeA requests some operation on NodeB, e.g. job execution. HTTP/HTTPS is also used for streaming data between workers of parallel execution

2. asynchronous messaging (TCP connection on port 7800 by default)

Typically heart-beat or events, e.g. job started or finished.

3. shared database - each node must be able to create DB connection

Shared configuration data, execution history, etc.

Following scenarios are described below one by one, however they may occur together:

- NodeA Cannot Establish HTTP Connection to NodeB (p. 295)
- NodeA Cannot Establish TCP Connection (Port 7800 by Default) to NodeB (p. 296)
- NodeB is Killed or It Cannot Connect to the Database (p. 296)
- <u>Auto-Resuming in Unreliable Network</u> (p. 297)
- Long-Term Network Malfunction May Cause Jobs to Hang on (p. 297)

NodeA Cannot Establish HTTP Connection to NodeB

When HTTP request can't be established between nodes, jobs which are delegated between nodes or jobs running in parallel on more nodes will fail. The error is visible in the Execution History. Each node periodically executes a check-task which checks the HTTP connection to other nodes. If the problem is detected, one of the nodes is suspended, since they can't cooperate with each other.

Time-line describing the scenario:

- Os network connection between NodeA and NodeB is down
- 0-40s a check-task running on NodeA can't establish HTTP connection to NodeB; check may last for 30s until it times-out; there is no re-try, if connection fails even just once, it is considered as unreliable, so the nodes can't cooperate.
- status of NodeA or NodeB (the one with shorter uptime) is changed to "suspended"

The following configuration properties set the time intervals mentioned above:

cluster.node.check.checkMiPeriodicity of Cluster node checks, in milliseconds.

Default: 20000

cluster.sync.connection.read HintPowonnection response timeout, in milliseconds.

Default: 30000

cluster.sync.connection.comstablishing de BEEP connection timeout, in milliseconds.

Default: 7000

NodeA Cannot Establish TCP Connection (Port 7800 by Default) to NodeB

TCP connection is used for asynchronous messaging. When the NodeB can't send/receive asynchronous messages, the other nodes aren't notified about started/finished jobs, so a parent jobflow running on NodeA keeps waiting for the event from NodeB. A heart-beat is vital for meaningful load-balancing, the same check-task mentioned above also checks a heart-beat from all Cluster nodes.

Time-line describing the scenario:

- Os network connection between NodeA and NodeB is down
- 60s NodeA uses the last available NodeB heart-beat
- 0-40s check-task running on NodeA detects missing heart-beat from NodeB
- status of NodeA or NodeB (the one with shorter uptime) is changed to suspended

The following configuration properties set the time intervals mentioned above:

cluster.node.check.checkMiPeriodicityaof Cluster node checks, in milliseconds.

Default: 40000

cluster.node.sendinfo.interevisidicity of heart-beat messages, in milliseconds.

Default: 2000

cluster.node.sendinfo.min_Antwartwheat may occasionally be sent more often than specified by cluster.node.sendinfo.interval. This property specifies the minimum interval in milliseconds.

Default: 500

cluster.node.remove.intervEble maximum interval for missing a heart-beat, in milliseconds.

Default: 50000

NodeB is Killed or It Cannot Connect to the Database

Access to a database is vital for running jobs, running scheduler and cooperation with other nodes. Touching a database is also used for detection of dead process. When the JVM process of NodeB is killed, it stops touching the database and the other nodes may detect it.

Time-line describing the scenario:

- 0s-30s last touch on DB
- NodeB or its connection to the database is down
- 90s NodeA sees the last touch
- 0-40s check-task running on NodeA detects obsolete touch from NodeB
- status of NodeB is changed to stopped, jobs running on the NodeB are solved, which means that their status is changed to UNKNOWN and the event is dispatched among the Cluster nodes. The job result is considered as error.

The following configuration properties set the time intervals mentioned above:

cluster.node.touch.intervaleriodicity of a database touch, in milliseconds.

Default: 20000

cluster.node.touch.forced_Astripperivaltementate other nodes accept the last touch, in milliseconds.

Default: 60000

cluster.node.check.checkMiPeriodicityaof Cluster node checks, in milliseconds.

Default: 40000

cluster.node.touch.forced_Ast**bqolean**lwalueuwhichgcajolsswitchatbdesdolving of running jobs mentioned above.

Auto-Resuming in Unreliable Network

In version 4.4, auto-resuming of suspended nodes was introduced.

Time-line describing the scenario:

- NodeB is suspended after connection loss
- 0s NodeA successfully reestablishes connection to NodeB
- 120s NodeA changes NodeB status to forced_resume
- NodeB attempts to resume itself if the maximum auto-resume count is not reached.
- If the connection is lost again, the cycle repeats; if the maximum auto-resume count is exceeded, the node will remain suspended until the counter is reset, to prevent suspend-resume cycles.
- 240m auto-resume counter is reset

The following configuration properties set the time intervals mentioned above:

cluster.node.check.intervallibre from the base to be forcibly resumed, in milliseconds.

Default: 120000

cluster.node.check.maxAutoHersumangotimets a node may try to auto-resume itself.

Default: 3

cluster.node.check.intervallinesterfore thereasones actuation will be reset, in minutes.

Default: 240

Long-Term Network Malfunction May Cause Jobs to Hang on

Jobflow or master execution executing child jobs on another Cluster nodes must be notified about status changes of their child jobs. When the asynchronous messaging doesn't work, events from the child jobs aren't delivered, so parent jobs keep running. When the network works again, the child job events may be re-transmitted, so hung parent job may be finished. However, the network malfunction may be so long, that the event can't be re-transmitted.

See following time-line to consider proper configuration:

- job A running on NodeA executes job B running on NodeB
- network between NodeA and NodeB is down from some reason
- job B finishes and sends the finished event, however it can't be delivered to NodeA the event is stored in the sent events buffer
- Since the network is down, a heart-beat can't be delivered as well and maybe HTTP connections can't be established, the Cluster reacts as described in the sections above. Even though the nodes may be suspended, parent job A keeps waiting for the event from job B.
- now, there are 3 possibilities:

- a. Network finally starts working and since all undelivered events are in the sent events buffer, they are re-transmitted and all of them are finally delivered. Parent job A is notified and proceeds. It may fail later, since some Cluster nodes may be suspended.
- b. Network finally starts working, but the number of the events sent during the malfunction exceeded the sent events buffer limit size. So some messages are lost and won't be re-transmitted. Thus the buffer size limit should be higher in the environment with unreliable network. Default buffer size limit is 10,000 events. It should be sufficient for thousands of simple job executions; basically, it depends on number of job phases. Each job execution produces at least 3 events (job started, phase finished, job finished). Please note that there are also other events fired occasionally (configuration changes, suspending, resuming, cache invalidation). Also messaging layer itself stores own messages to the buffer, but the number is negligible (tens of messages per hour). Heart-beat is not stored in the buffer.

There is also an inbound events buffer used as a temporary storage for events, so events may be delivered in correct order when some events can't be delivered at the moment. When the Cluster node is inaccessible, the inbound buffer is released after timeout, which is set to 1 hour, by default.

c. Node B is restarted, so all undelivered events in the buffer are lost.

The following configuration properties set the time intervals mentioned above:

cluster.jgroups.protocol.N2iNAISMhgsiz& aof the sent events buffer; Note that each stored message takes 2kB of heap memory.

Default: 10000

cluster.jgroups.protocol.NAKACKyuzahbutfferabileeuobbsanhecessihkenClesterionkeout

Part VIII. Security

Chapter 45. Security Recommendations for CloverDX Server

To improve security of CloverDX Server, you should:

- Change the default password for **clover** user. Without changing the password, everybody would be able to log in as **clover**. See <u>Change Users Password</u> (p. 129).
- Create a user different from **clover** and add it to the **admin** group. If there are more administrators, create a user account for each. See <u>Users</u> (p. 129).
- Set the **master password**. Without the master password, you cannot use secure parameters. See Chapter 20, <u>Secure Parameters</u> (p. 120).
- Run CloverDX Server with privileges of an ordinary user, e.g. create a system account clover used only for running CloverDX Server. Do not run CloverDX Server under the root account.
- Communication with system database may be unencrypted. Consider encrypting the connection to system database too.
- If database provides you with a root/admin account, do not use this account for **CloverDX Server**. Create a separate database user account, e.g. **clover**.
- Run CloverDX Server on HTTPS. If you communicate over HTTP, your data is sent unencrypted and eavesdroppers can easily see it.
- Disable the HTTP API if you do not need it. See Chapter 36, Simple HTTP API (p. 235).
- In Data Services, put keystores outside a sandbox and run the service on HTTPS. If you have a keystore in a sandbox, a user with write permissions could replace it with another key store. <u>HTTPS Connectors</u> (p. 266).
- Enable **user lockout** after repeated failed login attempts. If you use this feature in Cluster, make sure that all cluster nodes have the same lockout configuration. See <u>User Lockout</u> (p. 143)

List of Figures

1.1. CloverDX Server User Interface	2
2.1. System Architecture	3
7.1. setenv.sh edited in Linux.	13
7.2. setenv.bat edited in Windows	. 14
7.3. Adjusting Maximum heap size limit	22
7.4. Login page of CloverDX Server without license	32
7.5. Add new license form	. 33
7.6. Update license form	34
10.1. CloverDX Server as the only running application on IBM WebSphere	. 44
11.1. CloverDX Server's System Database Configuration	. 49
13.1. Setup GUI with decorators	56
13.2. Example of the Server Configuration file	56
13.3. The License tab	57
13.4. Database connection configuration for JDBC (left) and JNDI (right)	58
13.5. The Worker tab	. 58
13.6. Sandbox path configuration with clustering disabled (left) and enabled (right)	. 59
13.7. Encryption configuration	. 59
13.8. E-mail configuration	60
13.9. LDAP configuration	. 61
13.10. Cluster configuration	62
15.1. MBean for a JNDI datasource in jconsole	92
18.1. Standalone server detail	108
18.2. Resource Utilization	109
18.3. Worker	109
18.4. System	109
18.5. Performance	109
18.6. CPU Load	110
18.7. Running jobs	110
18.8. Status History	110
18.9. Users' Accesses panel	111
18.10. Status	111
18.11. Status	112
18.12. Heartbeat	112
18.13. Threads	112
18.14. Quartz	112
18.15. Cluster overview	113
18.16. Node detail	114
18.17. Server Logs	115
19.1. Configured temp spaces overview - one default temp space on each cluster node	117
20.1. Master password initialization	120
20.2. Graph parameters tab with initialized master password	120
22.1. Sandboxes Section in CloverDX Server Web GUI	145
22.2. Sandbox Permissions in CloverDX Server Web GUI	148
22.3. GUI - Sandboxes context menu	149
22.4. Job config properties	153
23.1. Server Configuration Export screen	157
23.2. Server Configuration Import screen	158
23.3. Updated User Groups - User added to a group	159
28.1. Web GUI - send email	174
28.2. Web GUI - shell command	176
28.3. Web GUI - Graph execution task	180
28.4. Web GUI - Jobflow execution task	183
28.5. Web GUI - Profiler job execution task	184
28.6. Web GUI - "Abort job"	185
28.7. Web GUI - archive records	187

28.8. Web GUI - Task JMS message editor	189
29.1. Web GUI - "Manual task execution" form	192
30.1. Web GUI - Scheduling	193
30.2. Web GUI - onetime schedule form	194
30.3. Web GUI - periodical schedule form	195
30.4. Cron periodical schedule form	196
30.5. Schedule allocation - One ore more specific nodes	197
31.1. Execution History - executions table	199
31.2. Execution History - overall perspective	201
31.3. Execution History - Tracking	202
31.4. Execution History - Tracking	203
32.1. Listeners	204
32.2. The event source graph isn't specified, thus the listener works for all graphs in the specified sandbox 2	208
32.3. Web GUI - email notification about graph failure	209
32.4. Web GUI - email notification about graph success	210
32.5 Web GUI - backup of data processed by graph	211
32.6 Web GUI - creating a File Event listener	221
32.7. File available on local file system	221
32.8 File available on remote location	223
32.0. Web GUIL creating a Task Failure listener	223
32.9. Web OUI - cleaning a Task Fahule listener	229
20.1. Loursh Services and ClauerDV Service as web application hads and	247
20.2. Creating a new lower as well application back-end	250
39.2. Creating a new launch configuration	252
39.3. Overview tab	252
39.4. Edit Configuration tab	253
39.5. Creating new parameter	254
39.6. Edit Parameters tab	254
40.1. Data Services	257
40.2. Endpoints	258
40.3. Data Service Endpoints	258
40.4. List of Data Services	259
40.5. Data Service Detail tab	260
40.6. Data Service - Testing and Documentation tab	261
40.7. Data Service - State and History	261
40.8. Data Service - Alerts and Notification	262
40.9. Data Service - Alerts and Notification	263
40.10. Some Data Service is failing	263
40.11. E-mail Notification	264
40.12. Data Service	264
40.13. Global Catalog of Services	265
40.14. Global Catalog of Services	265
40.15. Data Services - Publishing the examples	266
40.16. Data Services - Published the examples	266
40 17 HTTPS Connectors	266
40.18 HTTPS Connectors	267
40.19 Publishing Data Service job	269
40.20 Publishing Data Service job that does not require authentication	269
40.21. Publishing Data Service examples	20)
40.22 Publishing Data Service examples	270
40.22. Fublishing Data Service examples - II	270
40.24 Data Service without authentication	2/1 271
40.25. Creating a new Data Service Converter	2/1
40.26 Creating a new Data Service Connector	271
40.27 Using a new Data Service Connector II	272
40.27. Using the HTTPS Connector in Data Service endpoint	272
40.28. Data Services - Export	273
40.29. Data Services - Import	274
41.1. Dialog form for creating a new shared sandbox	277
41.2. Dialog form for creating a new local sandbox	278

41.3. Dialog form for creating a new partitioned sandbox	278
41.4. Remote Edge Implementation	280
42.1. Configuration of 2-nodes Cluster, each node has access to a database	286
42.2. Configuration of 2-nodes Cluster, one node without direct access to database	. 287
42.3. Configuration of 2-nodes Cluster with load balancer	288
42.4. Configuration of 3-nodes cluster, each node has access to a database	. 290

List of Tables

6.1. Hardware requirements of CloverDX Server	9
6.2. Compatibility Matrix	. 10
8.1. JVM Memory Structure	. 37
8.2. Recommended Heap Memory	38
8.3. Firewall Exceptions	. 39
15.1. General configuration	. 82
15.2. Server - Worker configuration	. 90
15.3. Properties for JDBC JNDI Resources in Worker	. 93
15.4. Properties for JMS JNDI Resources in Worker	. 94
15.5. Properties for SSL communication in Worker	. 95
15.6. List of Properties on Worker's Command Line	. 96
15.7. Defaults for job execution configuration - see Job Config Properties for details	. 98
16.1. Parameters	102
20.1. Secure parameters configuration parameters	122
21.1. Admin user	129
21.2. User attributes	129
21.3. Default groups created during installation	131
21.4 User lockout parameters	143
22.1 Sandbox attributes	146
22.2. Sandbox autorities	148
22.2. Sundova permissions	140
22.4. Job configuration properties	151
22.4. sob configuration properties	151
27.1. Defaults for graph execution configuration	168
27.2. persed perspectate	160
27.2. passed parameters	109
27.5. passed parameters	170
27.4. passed parameters	170
28.1. Altributes of Send e-mail task	175
28.2. Placeholders useful in email templates	175
28.3. Attributes of Execute shell command task	170
28.4. Parameters of Execute shell command	170
28.5. Attributes of "Graph execution" task	179
28.6. Additional parameters available in Event Listeners	181
28.7. Attributes of "Jobflow execution" task	182
28.8. Attributes of "Abort job" task	185
28.9. Attributes of "Archivator" task	186
28.10. Attributes of JMS message task	188
28.11. Parameters of "Send a JMS Message"	189
28.12. List of variables available in Groovy code	190
30.1. Onetime schedule attributes	194
30.2. Periodical schedule attributes	195
30.3. Cron periodical schedule attributes	196
31.1. Persistent run record attributes	200
31.2. Tracking table information	202
32.1. Event Listeners Table Description	204
32.2. Attributes of JMS message task	214
32.3. Variables accessible in groovy code	216
32.4. Properties Elements	217
32.5. "Data" elements	217
32.6. Attributes of Universal message task	219
32.7. Variables accessible in Groovy code	220
32.8. Parameters passed from the listener to the task	222
32.9. Parameters usable in task failure email template	230
32.10. File Event Listener specific parameters usable in task failure email template	230
36.1. Parameters of graph_run	237

36.2. Parameters of graph_status	238
36.3. Parameters of graph_kill	238
36.4. Parameters of sandbox_content	239
36.5. Parameters of executions_history	240
36.6. Parameters of suspend	241
36.7. Parameters of resume	241
36.8. Parameters of sandbox create	242
36.9. Parameters of sandbox add location	242
36.10. Parameters of sandbox add location	243
36.11. Parameters	243
36.12. Parameters	243
36.13. Parameters of server configuration export	244
36.14. Parameters of server configuration import	245
40.1. Attributes in Create HTTPS Connector dialog	268
42.1. Mandatory Cluster properties	282
42.2. Optional general properties	283
42.3. Optional remote edge properties	285
42.4. Load balancing properties	292