This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732253.

How to deploy the Q-Rapids Source Data Connectors [qr-connect]

Fraunhofer IESE

Prerequisites

The Q-Rapids source data connectors consist of a set of Apache Kafka connectors[1] that allows for storing data from various sources (Jenkins, Jira, sonarqube, and svn) into an Apache Kafka Server and from there into an Elasticsearch server. The connectors are based on the Confluent Kafka Connect Framework and are developed in Java. An already available Kafka connector for Elasticsearch[2] allows transferring the data into an Elasticsearch Cluster to allow for searching and querying the collected data. The prerequisites to deploy qr-connect are:

- A Linux Server with Oracle Java SE 8 installed
- Elasticsearch and Kibana installed (tested with version 5.4). The setup instructions for Elasticsearch are located at https://www.elastic.co/guide/en/elasticsearch/reference/current/setup.html while the instructions for Kibana are located at https://www.elastic.co/guide/en/kibana/current/setup.html.
- Confluent Open Source Kafka framework (tested with version 3.2.1). The installation instructions for the Confluent Platform are found at https://docs.confluent.io/current/installation/index.html

QR-Connect Framework

The qr-connect framework consists of an executable jar file containing the Apache Kafka Connectors, a set of configuration files that configure the options for the data collection (e.g. ip-addresses of the source systems), and a set of schema files for Elasticsearch that define the datatypes of the target indices.

Before running any connector, make sure that

- The Elasticsearch server is up and running. Check with a browser that the address `<your-elasticsearch-server-ip>:9200` shows something like

```
{
  "name" : "Mu_73pu",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "GbvewoODSAKub4595F0WHg",
  "version" : {
    "number" : "5.4.0",
    "build_hash" : "780f8c4",
    "build_date" : "2017-04-28T17:43:27.229Z",
    "build_snapshot" : false,
    "lucene_version" : "6.5.0"
  },
  "tagline" : "You Know, for Search"
}
```

- The Kibana Server is up and running Check with a browser that the Kibana GUI is displayed at `<your-elasticsearch-server-ip>:5601`

[1] https://www.confluent.io/product/connectors/
[2] https://www.elastic.co/de/products/elasticsearch

- The Kafka Server is up and running. You have to start zookeeper first:
  ```
  <Apache-Kafka-Dir>/bin/zookeeper-server-start <zookeeper-properties-file>
  <Apache-Kafka-Dir>/bin/kafka-server-start <kafka-server-properties-file>
  ```


SVN Source Connector

The svn connector periodically polls a svn repository for new commits.

The connector collects data about commits in the following format:

Attribute	Description	Example
revision	The svn revision	70
message	The commit message of the revision	Issue XXXX: Description
author	The author of the revision (svn username)	<svn username>
date	UTC date	2015-12-02T17:39Z
filename	A name of a file or directory that is added, removed, or changed with the commit	/<projectname>/trunk/…/Some.java
nodekind	Kind of element	"file" or "dir"
action	Repository action	"A" – added "D" – deleted "M" – modified

The index mapping for the svn Elasticsearch index (elasticsearch.svn.schema) has to be put into Elasticsearch before the connector is run (e.g. by using Kibana Dev Tools):

```
PUT svn
{
  "mappings": {
    "svn": {
      "properties": {
        "action":   { "type": "keyword" },
        "author":   { "type": "keyword" },
        "date":     { "type": "date" },
        "filename": { "type": "keyword" },
        "message":  { "type": "text" },
        "nodekind": { "type": "keyword" },
        "revision": { "type": "long"}
      }
    }
  }
}
```

The svn source connector configuration is stored in a properties file (connect-svn-source.properties):

Attribute	Description	Example
repository.url	The repository base URL	https://<svn-server-ip>/repo

repository.path	The path is appended to the repository URL. Concrete values depend on your repository layout	/<projectname>/trunk
repository.user	Authentication Username	<Username>
repository.pass	Authentication Password	<Password>
topic	Kafka topic name for storing the svn data	svn
name	Name of the connector	kafka-svn-source-connector
connector.class	Classname of the class implementing the connector	connect.svn.SubversionSourceConnector
poll.interval.seconds	Polling interval in seconds	60

An additional configuration file is used to define the Elasticsearch server as a data sink (connect-svn-elasticsearch.properties). Every item stored in the Kafka server is also passed to Elasticsearch.

Attribute	**Description**	**Example**
name	Name of the connector	svn-elasticsearch
connector.class	Classname of the class implementing the connector	io.confluent.connect.elasticsearch.ElasticsearchSinkConnector
tasks.max	Max. number of concurrent tasks	1
topics	The Kafka Topic to read, Elasticsearch index name to store to	svn
name	Name of the connector	kafka-svn-source-connector
connection.url	URL of the Elasticsearch Server	http://<elasticsearch-server-ip>:9200
type.name	Index type name for Elasticsearch	svn

One more configuration file defines the basic connector configuration (connect-svn.properties):

Attribute	**Description**	**Example**
bootstrap.servers	Kafka Server URL	<kafka-bootstrap-server-ip>:9092

offset.storage.file.filename	Kafka offset storage, resume reading source data at the correct position	/tmp/connect-svn.offsets
rest.port	Rest port of the connector. If multiple connectors are run, this port has to be different for every instance	8085
…	Standard attributes not shown	…

Before running the connector, at least adapt the following configuration values:

- connect-svn.properties:
  - bootstrap.servers (set to your kafka server ip)
- connect-svn-source.properties:
  - repository.url (set to your svn server base url)
  - repository.path (set to a path within your repository, e.g. "/project/trunk")
  - repository.user (valid svn username)
  - repository.pass (valid password)

To run the connector for svn, use the following command within the qr-connect directory (put into one line):

```
CLASSPATH=qr-connect-0.0.2-jar-with-dependencies.jar
<kafka-install-dir>/bin/connect-standalone
<qr-connect-install-dir>/connect-svn.properties
<qr-connect-install-dir>/connect-svn-source.properties
<qr-connect-install-dir>/connect-svn-elasticsearch.properties
```


Jenkins Source Connector

The Jenkins connector periodically polls a Jenkins server for new builds of a defined list of jobs. The connector collects data about builds in the following format:

Attribute	Description	Example
created	Timestamp of data collection UTC	2017-10-12T09:38Z
jenkinsUrl	URL of the Jenkins server	http://<ip-address>:8080/
jobName	Job name	job1
jobUrl	URL of the job	http://<ip-address>:8080/job/job1/
jobClazz	Jenkins job class	hudson.model.FreeStyleProject
displayName	Jenkins job display name	job1
lastBuild	Last build number	39
lastCompletedBuild	Last complete build number	31
lastFailedBuild	Last failed build number	31
lastStableBuild	Last stable build number	39
lastSuccessfullBuild	Last successful build number	39
lastUnstableBuild	Last unstable build number	-1
lastUnsuccessfullBuild	Last unsuccessful build number	31
buildNumber	Job build number	39
buildUrl	URL of the build	http://<ip-address>:8080/job/job1/39/
buildDescription	Build description	null
duration	Build duration	13947
estimatedDuration	Estimated build duration	11983
result	build result	SUCCESS
timestamp	Timestamp of build	2017-10-10T09:15Z
testsFail	Number of failed tests (e.g. provided by JUnit test run)	0
testsPass	Number of passed tests (e.g. provided by JUnit test run)	3
testsSkip	Number of skipped tests (e.g. provided by JUnit test run)	0

testDuration	Test Duration	0.002

The index mapping for the jenkins Elasticsearch index (elasticsearch.jenkins.schema) has to be put into Elasticsearch before the connector is run (e.g. by using Kibana Dev Tools):

```
PUT jenkins
{
  "mappings": {
    "jenkins": {
      "properties": {
        "buildClazz": { "type": "keyword" },
        "buildDescription": { "type": "text" },
        "buildNumber": { "type": "integer" },
        "buildUrl": { "type": "keyword" },
        "created": { "type": "date" },
        "displayName": { "type": "keyword" },
        "duration": { "type": "long" },
        "estimatedDuration": { "type": "long" },
        "jenkinsUrl": { "type": "keyword" },
        "jobClazz": { "type": "keyword" },
        "jobName": { "type": "keyword" },
        "jobUrl": { "type": "keyword" },
        "lastBuild": { "type": "integer" },
        "lastCompletedBuild": { "type": "integer" },
        "lastFailedBuild": { "type": "integer" },
        "lastStableBuild": { "type": "integer" },
        "lastSuccessfullBuild": { "type": "integer" },
        "lastUnstableBuild": { "type": "integer" },
        "lastUnsuccessfullBuild": { "type": "integer" },
        "result": { "type": "keyword" },
        "testDuration": { "type": "double" },
        "testsFail": { "type": "long" },
        "testsPass": { "type": "long" },
        "testsSkip": { "type": "long" },
        "timestamp": { "type": "date" }
      }
    }
  }
}
```

The Jenkins source connector configuration is stored in a properties file (connect-jenkins-source.properties):

Attribute	**Description**	**Example**
name	Name of the connector	kafka-jenkins-source-connector
connector.class	Classname of the class implementing the connector	connect.jenkins.JenkinsSourceConnector

jenkins.url	URL of the Jenkins server	http://<jenkins-server-ip>:8080/
jenkins.user	Authentication Username	<Username>
jenkins.pass	Authentication Password	<Password>
jenkins.topic	Kafka topic name	jenkins
jenkins.jobs	List of job names to be analyzed	job1, job2
jenkins.lookback	When started the first time, fetch only this number of recent builds	100
jenkins.interval.seconds	Poll interval in seconds	60
use.preemptive.auth	Activates Nokia version with special authentication if set to true, default false.	false

An additional configuration file is used to define the Elasticsearch server as a data sink (connect-jenkins-elasticsearch.properties). Every Jenkins item stored in the Kafka server is also passed to Elasticsearch.

Attribute	**Description**	**Example**
name	Name of the connector	kafka-jenkins-elasticsearch
connector.class	Classname of the class implementing the connector	io.confluent.connect.elasticsearch. ElasticsearchSinkConnector
tasks.max	Max. number of concurrent tasks	1
topics	The Kafka Topic to read, Elasticsearch index name to store to	jenkins
connection.url	URL of the Elasticsearch Server	http://<elasticsearch-server-ip>:9200
type.name	Index type name for Elasticsearch	jenkins

One more configuration file defines the basic connector configuration (connect-jenkins.properties):

Attribute	**Description**	**Example**

bootstrap.servers	Kafka Server URL	<kafka-bootstrap-server-ip>:9092
offset.storage.file.filename	Kafka offset storage, resume reading source data at the correct position	/tmp/connect-jenkins.offsets
rest.port	Rest port of the connector. If multiple connectors are run, this port has to be different for every instance	8086
…	Standard attributes not shown	…

Before running the connector, at least adapt the following configuration values:

- connect-jenkins.properties:
  - bootstrap.servers (set to your kafka server ip)
- connect-jenkins-source.properties:
  - jenkins.url (set to your jenins server url)
  - jenkins.user (valid jenkins username)
  - jenkins.pass (valid password)
  - jenkins.jobs (list of jobs to observe)

To run the connector for svn, use the following command within the qr-connect directory (put into one line):

```
CLASSPATH=qr-connect-0.0.2-jar-with-dependencies.jar
<kafka-install-dir>/bin/connect-standalone
<qr-connect-install-dir>/connect-jenkins.properties
<qr-connect-install-dir>/connect-jenkins-source.properties
<qr-connect-install-dir>/connect-jenkins-
elasticsearch.properties
```


Jira Source Connector

The Jira connector periodically polls a Jira server for issues of a defined project. The connector collects data about issues in the following format:

Attribute	Description	Example
jiraUrl	URL of the Jira server	http://<jira-ip>:<jira-port>
projectKey	The project key	QRAP
projectName	The project name	Q-Rapids-Project
jiraIssueApi	API URL for the issue	http://<jira-ip>:<jira-port>/ rest/api/2/issue/10102
issuetype	Issue type	Bug
timespent	Time spent on issue	null
description	Description of the issue	Repair this stuff.
aggregatetimespent	Time spent on issue including child issues	null
resolution	Issue resolution	null
aggregatetimeestimate	Aggregate time estimate	null
resolutiondate	Date no resolution	null
summary	Issue summary	Always something wrong!
lastViewed	Timestamp	2017-09-13T10:40:36.572+0200
creator	User who created the issue	admin
created	Timestamp	2017-09-13T10:40:36.000+0200
reporter	User who reported the issue	admin
priority	Issue priority	Medium
timeestimate	Time estimate	null
duedate	Due date	null
assignee	Assigned to …	null
updated	Timestamp	2017-09-13T10:40:36.000+0200
status	Issue status	

The index mapping for the Jira Elasticsearch index (elasticsearch.jira.schema) has to be put into Elasticsearch before the connector is run (e.g. by using Kibana Dev Tools):

```
PUT jira
{
  "mappings": {
```


```
    "jira": {
      "properties": {
        "aggregatetimeestimate": { "type": "long" },
        "aggregatetimespent": { "type": "long" },
        "assignee": { "type": "keyword" },
        "created": { "type": "date" },
        "creator": { "type": "keyword" },
        "description": { "type": "text" },
        "duedate": { "type": "date" },
        "issuetype": { "type": "keyword" },
        "issueid": { "type": "keyword" },
        "issuekey": { "type": "keyword" },
        "jiraIssueApi": { "type": "keyword" },
        "jiraUrl": { "type": "keyword" },
        "lastViewed": { "type": "date" },
        "priority": { "type": "keyword" },
        "projectKey": { "type": "keyword" },
        "projectName": { "type": "keyword" },
        "reporter": { "type": "keyword" },
        "resolution": { "type": "keyword" },
        "resolutiondate": { "type": "date" },
        "status": { "type": "keyword" },
        "summary": { "type": "text" },
        "timeestimate": { "type": "long"  },
        "timespent": { "type": "long" },
        "updated": { "type": "date" }
      }
    }
  }
}
```

The Jira source connector configuration is stored in a properties file (connect-jira-source.properties):

Attribute	**Description**	**Example**
name	Name of the connector	kafka-jira-source-connector
connector.class	Classname of the class implementing the connector	connect.jira.JiraSourceConnector
jira.url	URL of the Jira server	http://<jira-server-ip>:<port>/
jira.user	Authentication Username	<Username>
jira.pass	Authentication Password	<Password>
jira.topic	Kafka topic name	jira
jira.project	The Jira project to be analyzed	q-rapids

jira.interval.seconds	Poll interval in seconds	60
jira.created.since	Fetch issues created since this date	2017-01-01

An additional configuration file is used to define the Elasticsearch server as a data sink (connect-jira-elasticsearch.properties). Every Jenkins item stored in the Kafka server is also passed to Elasticsearch.

Attribute	**Description**	**Example**
name	Name of the connector	kafka-jira-elasticsearch
connector.class	Classname of the class implementing the connector	io.confluent.connect.elasticsearch.ElasticsearchSinkConnector
tasks.max	Max. number of concurrent tasks	1
topics	The Kafka Topic to read, Elasticsearch index name to store to	jira
connection.url	URL of the Elasticsearch Server	http://<elasticsearch-server-ip>:9200
type.name	Index type name for Elasticsearch	jira

One more configuration file defines the basic connector configuration (connect-jira.properties):

Attribute	**Description**	**Example**
bootstrap.servers	Kafka Server URL	<kafka-bootstrap-server-ip>:9092
offset.storage.file.filename	Kafka offset storage, resume reading source data at the correct position	/tmp/connect-jira.offsets
rest.port	Rest port of the connector. If multiple connectors are run, this port has to be different for every instance	8087
…	Standard attributes not shown	…

Before running the connector, at least adapt the following configuration values:

- connect-jira.properties:

    - bootstrap.servers (set to your kafka server ip)
- connect-jira-source.properties:
    - jira.url (set to your jenins server url)
    - jira.user (valid jenkins username)
    - jira.pass (valid password)
    - jira.project (projects to collect issues from)

To run the connector for jira, use the following command within the qr-connect directory (put into one line):

```
CLASSPATH=qr-connect-0.0.2-jar-with-dependencies.jar
<confluent-kafka-install-dir>/bin/connect-standalone
<qr-connect-install-dir>/connect-jira.properties
<qr-connect-install-dir>/connect-jira-source.properties
<qr-connect-install-dir>/connect-jira-elasticsearch.properties
```


Sonarqube Source Connector

The Sonarqube connector periodically polls a Sonarqube server for measures and code issues and produces two types of records:

Measures:

Attribute	Description	Example
sonarUrl	URL of the Sonarqube server	**http://<ip-address>:<port>**
snapshotDate	Date record was produced (read into kafka).	2017-09-15T11:30Z
bcId	baseComponentdId in Sonarqube	AV5_x36Jt2orMGobWbjx
bcKey	baseComponentKey in Sonarqube. This equals the "sonar.projectKey" specified in the sonar-project.properties file	tomcat:9.x
bcName	Project name in Sonarqube. This equals the "sonar.ProjectName" specified in the sonar-project.properties file	Tomcat
bcQualifier	One of: VW: view SVW: sub-view TRK: project BRC: module CLA: class UTS: unit test DIR: directory FIL: file DEV: developer	TRK
Id	Internal Sonarqube id	AV5_x4oatOvqTRshLU_S
key	Component key	tomcat:9.x:javax/servlet/jsp/tagext/TagInfo.java
name	Component name	TagInfo.java
qualifier	Component Qualifier. One of VW: view SVW: sub-view TRK: project BRC: module CLA: class UTS: unit test DIR: directory	FIL

	FIL: file DEV: developer	
language	Programming language	java
metric	Metric name	functions
value	Textual value Representation	"7"
floatValue	Float value represenatatin (if applicable)	7.0

The index mapping for the Sonarqube Measures Elasticsearch index (elasticsearch.sonarqube.measure.schema) has to be put into Elasticsearch before the connector is run (e.g. by using Kibana Dev Tools):

```
PUT sonar.measure
{
  "mappings": {
    "sonarqube": {
      "properties": {
        "Id": { "type": "keyword" },
        "bcId": { "type": "keyword" },
        "bcKey": { "type": "keyword" },
        "bcName": { "type": "keyword" },
        "bcQualifier": { "type": "keyword" },
        "floatvalue": { "type": "float" },
        "key": { "type": "keyword" },
        "language": { "type": "keyword" },
        "metric": { "type": "keyword" },
        "name": { "type": "keyword" },
        "path": { "type": "keyword" },
        "qualifier": { "type": "keyword" },
        "snapshotDate": { "type": "date" },
        "sonarUrl": { "type": "keyword" },
        "value": { "type": "keyword" }
      }
    }
  }
}
```

Sonarqube Code Issues:

Attribute	**Description**	**Example**
sonarUrl	URL of the Sonarqube server	**http://<ip-address>:<port>**
snapshotDate	Date record was produced (read into kafka).	2017-09-15T11:30Z
rule	Id of violated rule	squid:S00122
severity	Severity of violation	MINOR

component	Sonarqube component identifier	tomcat:9.x:org/apache/tomcat/util/net/NioBlockingSelector.java
componentId	Sonarqube internal component id	5045
project	Sonarqube project id. This equals the "sonar.projectKey" specified in the sonar-project.properties file	tomcat:9.x
line	Code line of violation	339
startLine	Start line of violation	339
startOffset	Character offset in startLine	0
endLine	End line of violation	java
endOffset	Character offset in endLine	60
effort	Estimated time to solve the issue	1min
debt	Issue debt	1min
author	Issue author	
creationDate	Sonarqube creation date	2017-09-14T11:43:37+0200

The index mapping for the Sonarqube Measures index (elasticsearch.sonarqube.issue.schema) has to be put into Elasticsearch before the connector is run (e.g. by using Kibana Dev Tools):

```
PUT sonarqube.issue
{
  "mappings": {
    "sonarqube": {
      "properties": {
        "author": { "type": "keyword" },
        "component": {  "type": "keyword" },
        "componentId": { "type": "integer" },
        "creationDate": { "type": "date" },
        "debt": { "type": "keyword" },
        "effort": { "type": "keyword" },
        "endLine": { "type": "integer" },
        "endOffset": { "type": "integer" },
        "key": { "type": "keyword" },
        "line": { "type": "integer" },
        "message": { "type": "text" },
        "project": { "type": "keyword" },
        "rule": { "type": "keyword" },
        "severity": { "type": "keyword" },
```


```
      "snapshotDate": { "type": "date" },
      "sonarUrl": { "type": "keyword" },
      "startLine": { "type": "integer" },
      "startOffset": { "type": "integer" },
      "status": { "type": "keyword" }
    }
   }
  }
}
```

The Sonarqube source connector configuration for both types of records is stored in a properties file (connect-sonarqube-source.properties):

Attribute	**Description**	**Example**
name	Name of the connector	kafka-sonar-source-connector
connector.class	Classname of the class implementing the connector	connect.sonarqube. SonarqubeSourceConnector
sonar.url	URL of the Sonarqube server	http://<sonar-ip>:<port>/
sonar.user	Authentication Username	<Username>
sonar.pass	Authentication Password	<Password>
sonar.basecomponent.key	ComponentID for analysis as defined "sonar.projectKey" specified in the sonar-project.properties file	
sonar.componentroots.key	ComponentID for analysis	
sonar.measure.topic	Kafka topic name for measures. Also, name for Elasticsearch index.	sonarqube.measure
sonar.issue.topic	Kafka topic name for issues. Also, name for Elasticsearch index.	sonarqube.issue
sonar.metric.keys	Metric keys to be collected	ncloc,lines,comment_lines,complexity, …

		See https://docs.sonarqube.org/display/SONAR/Metric+Definitions for full list
sonar.interval.seconds	Polling interval. At every poll, a new set of records is collected, regardless if data has changed or not. The interval has to be in sync with the actual sonarqube metric collection.	86400 (one day)

An additional configuration file is used to define the Elasticsearch server as a data sink (connect-sonarqube-elasticsearch.properties). Every Sonarqube item stored in the Kafka server is also passed to Elasticsearch.

Attribute	**Description**	**Example**
name	Name of the connector	kafka-sonarqube-elasticsearch
connector.class	Classname of the class implementing the connector	io.confluent.connect.elasticsearch.ElasticsearchSinkConnector
tasks.max	Max. number of concurrent tasks	1
topics	The Kafka Topic to read, Elasticsearch index name to store to	sonarqube.measure,sonarqube.issue
connection.url	URL of the Elasticsearch Server	http://<elasticsearch-server-ip>:9200
type.name	Index type name for Elasticsearch	sonarqube

One more configuration file defines the basic connector configuration (connect-sonarqube.properties):

Attribute	**Description**	**Example**
bootstrap.servers	Kafka Server URL	<kafka-bootstrap-server-ip>:9092
offset.storage.file.filename	Kafka offset storage, resume reading source data at the correct position	/tmp/connect-sonarqube.offsets
rest.port	Rest port of the connector. If multiple connectors are run, this port has to be	8088

	different for every instance	
…	Standard attributes not shown	…

Before running the connector, at least adapt the following configuration values:

- connect-sonarqube.properties:
  - bootstrap.servers (set to your kafka server ip)
- connect-sonarqube-source.properties:
  - sonarqube.url (set to your sonarqube server url)
  - sonarqube.user (valid jenkins username)
  - sonarqube.pass (valid password)
  - jira.project (projects to collect issues from)

To run the connector for sonarqube, use the following command within the qr-connect directory (put into one line):

```
CLASSPATH=qr-connect-0.0.2-jar-with-dependencies.jar
<confluent-kafka-install-dir>/bin/connect-standalone
<qr-connect-install-dir>/connect-sonarqube.properties
<qr-connect-install-dir>/connect-sonarqube-source.properties
<qr-connect-install-dir>/connect-sonarqube-
elasticsearch.properties
```


1 Redmine Source Connector

The Redmine connector periodically polls a Redmine API for new and updated issues:

Measures:

Attribute	Description	Example
redmineURL	URL of the Redmine server	http://<ip-address>:<port>
project_id	Project id (from database)	10
project	Redmine project name	MyProject
issue_id	Issue id	20
tracker	One of "Issue", "Evolution", "Assistance" etc.	Issue
tracker_id	Tracker id (from database)	99
status	Issue status. One of "New", "Assigned", "Resovlved", , "Feedback", etc.	New
status_id	Status id (from database)	100
priority	Issue priority (e.g. "Low", "Medium", "High")	Medium
priority_id	Priority id (from database)	101
author	Issue author	JohnDoe
author_id	Author id (from database)	101
assigned_to	Issue assigned to person	JohnDoe
assigned_to_id	Assigned to id from database	102
fixed_version	Version where issue was fixed	Version-1.3.5
fixed_version_id	Fixed version id (from database)	105
subject	Issue subject	Lore ipsum …
start_date	Start date	2017-10-10
done_ratio	Percentage issue done	50
created_on	Creation timestamp	2014-06-02T07:09:53Z
updated_on	Last update timestamp	2014-09-01T07:52:54Z

The index mapping for the Redmine Elasticsearch index (elasticsearch.redmine.schema) has to be put into Elasticsearch before the connector is run (e.g. by using Kibana Dev Tools):

```
PUT redmine
{
  "mappings": {
```


```
    "redmine": {
      "properties": {
        "redmineURL": { "type": "keyword" },
        "project_id": { "type": "long" },
        "project": { "type": "keyword" },
        "issue_id": { "type": "long" },
        "tracker": { "type": "keyword" },
        "tracker_id": { "type": "long" },
        "status": { "type": "keyword" },
        "status_id": { "type": "long" },
        "priority": { "type": "keyword" },
        "priority_id": { "type": "long" },
        "author": { "type": "keyword" },
        "author_id": { "type": "long" },
        "assigned_to": { "type": "keyword" },
        "assigned_to_id": { "type": "long" },
        "fixed_version": { "type": "keyword" },
        "fixed_version_id": { "type": "long" },
        "subject": { "type": "text" },
        "start_date": { "type": "keyword" },
        "done_ratio": { "type": "date" },
        "created_on": { "type": "keyword" },
        "updated_on": { "type": "text" }
      }
    }
  }
}
```

The Redmine source connector configuration is stored in a properties file (connect-redmine-source.properties):

Attribute	**Description**	**Example**
name	Name of the connector	kafka-sonar-source-connector
connector.class	Classname of the class implementing the connector	connect.redmine. RedmineSourceConnector
redmine.url	URL of the Redmine server	http://<sonar-ip>:<port>
redmine.user	Authentication Username, leave blank for no authentication	<Username>
redmine.pass	Authentication Password	<Password>

redmine.created.since	Read only issues created after this date	2000-01-01
redmine.topic	Kafka topic name for Redmine issues. Also, name for Elasticsearch index.	redmine
sonar.metric.keys	Metric keys to be collected	ncloc,lines,comment_lines,complexity, … See https://docs.sonarqube.org/display/SONAR/Metric+Definitions for full list
redmine.interval.seconds	Polling interval. At every poll, all new and updated issues are collected. or not. The interval has to be in sync with the actual sonarqube metric collection.	3600 (one hour)

An additional configuration file is used to define the Elasticsearch server as a data sink (connect-sonarqube-elasticsearch.properties). Every Sonarqube item stored in the Kafka server is also passed to Elasticsearch.

Attribute	**Description**	**Example**
name	Name of the connector	kafka-redmine-elasticsearch
connector.class	Classname of the class implementing the connector	io.confluent.connect.elasticsearch.ElasticsearchSinkConnector
tasks.max	Max. number of concurrent tasks	1
topics	The Kafka Topic to read, Elasticsearch index name to store to	redmine
connection.url	URL of the Elasticsearch Server	http://<elasticsearch-server-ip>:9200
type.name	Index type name for Elasticsearch	redmine

One more configuration file defines the basic connector configuration (connect-sonarqube.properties):

Attribute	**Description**	**Example**

bootstrap.servers	Kafka Server URL	<kafka-bootstrap-server-ip>:9092
offset.storage.file.filename	Kafka offset storage, resume reading source data at the correct position	/tmp/connect-sonarqube.offsets
rest.port	Rest port of the connector. If multiple connectors are run, this port has to be different for every instance	8088
…	Standard attributes not shown	…

Before running the connector, at least adapt the following configuration values:

- connect-redmine.properties:
  - bootstrap.servers (set to your kafka server ip)
- connect-redmine-source.properties:
  - redmine.url (set to your Redmine server url)
  - redmine.user (valid jenkins username, or empty if authentication is not needed)
  - redmine.pass (valid password)

To run the connector for sonarqube, use the following command within the qr-connect directory (put into one line):

```
CLASSPATH=qr-connect-0.0.2-jar-with-dependencies.jar
<confluent-kafka-install-dir>/bin/connect-standalone
<qr-connect-install-dir>/connect-redmine.properties
<qr-connect-install-dir>/connect-redmine-source.properties
<qr-connect-install-dir>/connect-redmine-
elasticsearch.properties
```

