

Scarab - Administration Guide

Scarab for sysadmin's

The Scarab Development Team

Scarab - Administration Guide: Scarab for sysadmin's

The Scarab Development Team

Table of Contents

1. Scarab Installation	1
General	1
How to install Scarab	1
The steps in a normal Scarab installation	1
Preparing the installation	1
Choosing an operating system	1
Choosing a database manager	1
Choosing a servlet container	2
Gathering the other elements required for the Scarab installation	3
Understanding and setting up NTLM authentication	4
Setting up and configuring your database for Scarab	6
Installing and configuring MySQL	6
Configuring PostgreSQL	11
Setting up and configuring Oracle	11
Building Scarab with the web install application	12
Step 1: Unpack the Scarab distribution	12
Step 2 : Unpack the web installer	12
Step 3: Configure and start Tomcat	12
Step 4: Configure Scarab	12
Step 5: Stop Tomcat	18
Step 6: Remove the web installer	18
Step 7: Restart Tomcat and launch Scarab	18
Building Scarab: command-line installation	18
Step 1: Unpack the Scarab distribution	18
Step 2: Migrating an old (pre b19) version of Scarab	18
Step 3: Configure Scarab	18
Step 4: Build Scarab	19
Step 5a: Create the database	19
Step 5b: Migrate your current data	20
Step 6: Start Tomcat	20
2. Scarab system administration after installation	21
Post-installation administration	21
Setting up an Apache 2 Virtual Host	21
Backups and Restores	22
Backing up Scarab information	22
Restoring Scarab information from a backup	23
Moving an existing Scarab installation to another server	23
Modifying the database when the Scarab server changes	23
3. Using Scarab with Subversion	25

Chapter 1. Scarab Installation

General

How to install Scarab

In the current state of development -- and maybe in the long term as well -- Scarab does not install fully assembled, as is usually the case with an installation program.

Scarab is built dynamically for every particular installation, according to local parameters among which the database used to store the issues and other information that Scarab needs to work.

This could make you anxious at first sight, but the process is simple and proven: you can achieve this task easily like most other people did until now.

The steps in a normal Scarab installation

A normal installation is performed in four steps :

- Preparing the installation
- Installing and configuring the database management software
- Building Scarab
- Building the Scarab database

Preparing the installation

Choosing an operating system

Scarab is theoretically portable on any system that supports Java.

Practically, Scarab has been used and intensively tested on three kinds of platforms:

- Linux: the platform on which the most experience has been gathered (and thus the system on which you will easily get support from the user community if ever you face a problem).
- Windows
- MacOS X: in fact, Scarab has been developed in part on this platform, and the first Scarab demonstrator, the "runbox", was running on this platform.

Choosing a database manager

Scarab needs a database management system to store the issues and other structured information (parameters, queries, reports, modules,...). Thus, you have to choose an RDBMS (for the database installation, see the next chapter).

Theoretically, you could use Scarab on any RDBMS that has a JDBC driver.

Practically, you are faced with some SQL compatibility issues, and moreover, you need to have correct support for this DBMS in Torque, the persistence and object-relational mapping component associated with the Turbine

framework, on which Scarab is built.

DBMS that have been tested with Scarab (and are supported)

MySQL

MySQL is the DBMS on which Scarab has been developed and fully tested. If you don't have any particular reason to choose a database or another to run Scarab, MySQL should be your first choice.

Which version to choose? Scarab will work with any 3.x or 4.x version of MySQL. It could happen in the future that the support of version 3 could be dropped to use the transactional functionalities that are available only in version 4. So, if you don't have a particular reason to choose a particular version, you may choose long term peace of mind and use a version 4.x. from the start.

MySQL is open-source software, and it is available for free on Linux, Windows and a few other platforms (commercial licences and support are available for a fee).

PostgreSQL

Scarab has been tested successfully on PostgreSQL, version 7.2.1 and the following.

PostgreSQL is open-source software, and it is available for free on Linux and Windows.

Oracle 9i

The compatibility of Scarab 0.20 with Oracle 9i has been tested

The compatibility of Scarab 0.21 has not been tested, neither with Oracle 9 nor with Oracle 10 (10g or 10 XE). The Scarab development team plans to test and improve the Oracle compatibility in the current version (TRUNK). Feel free to discuss any question you have or any issue you encounter in the mailing lists.

Because of a problem with the manner Torque manages BLOBs, using Scarab with Oracle 9i is a bit more difficult than with other DBMS (you will have to patch the code generated by Torque).

DBMS that have been tested and are not (fully) supported by Scarab

The following DBMS have their setup ready in Scarab and have been tested, but the tests have shown so far that Scarab is not fully operational on these platforms:

- Microsoft SQL Server¹
- HSQLDB
- DB2

Moreover:

- Apache Derby has been quickly tested recently; though it does not work at the moment, further testing is planned.
- Oracle 8i has been supported in the early versions of Scarab but is no longer supported (and will not be supported again)².

Choosing a servlet container

Scarab runs on any Servlet 2.3-compatible servlet engine; in fact, a patch could make Scarab compatible with 2.2 servlet engines, among which WebSphere Application Server 4.0 -- but this has not been tested for a very long time.
Problems remain with the use of the SQL CONCAT function and in the mapping of certain types by Torque.
²Unless someone volunteers but it is a tough problem with *right outer joins*.

Practically, most Scarab users use it on Tomcat; in fact, the Scarab installation process builds a "light" Tomcat server, the "sandbox", which you can use to test or run Scarab once it has been built.

You may want to deploy Scarab on an existing Tomcat installation; Resin has also been successfully tested as Scarab engine.

Deploying Scarab on "full" J2EE application servers has usually been tricky and has rarely worked. Recent tests have shown at least that it does not work "out of the box" neither on JBoss3 nor on WebLogic Server.

Gathering the other elements required for the Scarab installation

To perform the Scarab installation, you will need three more things: a Java SDK, Ant and -- of course -- Scarab itself.

Download and install a Java SDK

If you don't have a Java SDK handy (pre-installed on some systems such as MacOS X or certain Linux distributions), you need to download and install one. Be careful: you need a Java 2 SDK and not only a JRE (Java Runtime Environment) because you will need to compile Scarab.

The SDK version must be at least 1.4.

You may download a Java SDK from <http://java.sun.com>.

Make sure the JAVA_HOME environment variable correctly points to the directory in which your SDK is installed.

Download and install Ant

To build Scarab, you will need the Ant build system.

You can download Ant from <http://ant.apache.org>.

The Ant version should be 1.5.1 or greater.

Make sure the ANT_HOME environment variable correctly points to the directory in which Ant is installed



Caution

The version of ant that is supplied by RedHat with RHEL3 (latest update is ant-1.5.2-23) throws an exception when running the create-db target.

The exception is:

```
BUILD                                                                    FAILED
file:/export/home/admin/scarab/target/scarab/WEB-INF/conf/runtime-
torque.xml:72:                                                            taskdef                    class
org.apache.torque.task.TorqueDocumentationTask cannot be found
```

Work Around: use the version of ant that comes bundled with SCARAB:

- Copy it from `$$SCARAB_ROOT/www/repository/ant/jars/ant-1.5.3-1.jar` to `$$SCARAB_ROOT/build`
- From `$$SCARAB_ROOT/build` run `java -jar ant-1.5.3-1.jar create-db`

³Most probably there is at least a problem with the log4j settings.

Download Scarab

You need of course to download Scarab itself. You will find the latest version (1.0-b21) at the following URL: <http://scarab.tigris.org/servlets/ProjectDocumentList>.

Understanding and setting up NTLM authentication

Abstract

NTLM authentication provides automatic login (Single Sign-On) in Windows NT domains. It is only useful in carefully controlled and administered environments, it requires some additional setup, so if you don't plan to use this feature, you may (you should) skip this section.

Understanding NTLM authentication in Scarab

This development is intended to provide the means for automatic login on Windows networks. Initially based on code provided by Peter Nei (thanks!), current implementation is mostly based on JCIFS (see note at the end of this doc).

It's implemented as a new valve in the pipeline (NtlmLoginValve), that will try to get the credentials from the browser using the ntlm challenge protocol.

The credentials sent by the browser are not trusted directly, and they're authenticated in a NTLM domain.

Given that feature is not automatically cross-browser (it needs some config tweaks in Firefox, and it's not guaranteed to work on every browser), it might only makes sense in controlled network environments.

Behaviour

The NTLM login will be inactive by default, and it will be activated using two Scarab properties:

```
scarab.login.ntlm.active=true|false
```

The setting of this property to `true` will require the authentication domain to be defined.

```
scarab.login.ntlm.domain=mydomaincontroller
```

When browsing, the NTLMLoginValve will try to get the credentials and validate them against the NTLM domain. If they are valid, it will try logging in the user into Scarab. **For this to succeed, the NTLM username must match an Scarab username.** If the user gets logged in, a message will be displayed, such as

```
You've been automatically logged in using your user 'YourUser' from
```

If anyone from the NTLM credentials or its equivalent Scarab username are invalid, the valve will just 'pass' and the rest of the login system will proceed (Login form / Anonymous).

When an authenticated user (even Anonymous) click on the 'Logout/Login' link, the session will be terminated and the Login.vm screen will be displayed, so any user (NTLM or not) can try logging in the system with other credentials.

Single-Sign-On behaviour

User are still able to login using the old form, and this causes a problem because Scarab and NTLM passwords will not be synchronized.

Possible solutions:

1. Disable form-login: This would be a problem if some users are not to be in the Windows domain.

2. Use a UserPreference to choose between NTLM and form login for every user. This would effectively restrict any user to only one authentication method, be Scarab or NTLM, but it would add another administration task. password in Scarab, the NTLM PASSWORD WILL NOT CHANGE, and neither . Similarly, the Windows domain can revoke or

Login precedences

The different login methods allowed will be selected in this order:

1. Login form will always take precedence on others.
2. If the login-form has not been used, the system will try NTLM authentication (if activated!)
3. Anonymous login will be granted if allowed.

Firefox

Firefox users will be by default prompted with a dialog box asking for username and password. To login with Windows domain credentials, the user will have to fill them up using the DOMAIN\User format.

Example: Username: "MyDomain\mywindowsuser", Password: myplainpassword

If Firefox users want to automatically (no dialog!) get the credentials fed from the underlying Windows operating system, they have to:

- Type "about:config" in the URL box.
- Search for the 'network.automatic-ntlm-auth.trusted-uris' configuration value.
- Add the name of the Scarab host to that value.

Other browsers

The feature has not been tested on other browsers. At least, they would probably offer the dialog asking for username and password, but it may vary.

Preparing Scarab for NTLM authentication

Abstract

You have to perform some library setup before building Scarab; that is why this lengthy thread about NTLM authentication took place here.

jcifs

To perform NTLM authentication, Scarab relies on the jcifs library, part of the Samba project (<http://jcifs.samba.org>).

The license of jcifs is LGPL, so the jar cannot be included in the Scarab distribution.

Some Apache projects (Lenya) have been using jcifs, providing a **mock jar** to allow compiling and asking the administrator to manually download the (real) jar. This method fits Scarab too, so you will have to download jcifs yourself from one of these URLs:

- <http://jcifs.samba.org/>

- <http://www.ibiblio.org/maven/org.samba.jcifs/jars/jcifs-1.2.6.jar>

Once you have got the jar file, you must replace the jar that was delivered with Scarab. Be aware that if you are using Maven, you have possibly already got the mock version in your local Maven repository (usually at `$HOME/.maven/repository/org.samba.jcifs/jars`).



Warning

You must replace the (mock) jar with the real jar you have downloaded.

Setting up and configuring your database for Scarab

Installing and configuring MySQL

Guide for Linux with RPM packages

Grab the MySQL distribution

You will find an appropriate distribution through the MySQL website [<http://www.mysql.com>].

You can follow the instructions from the MySQL documentation. We shall give here summary info only.

Install the distribution

For example if you have grabbed version 4.0.22, you can install MySQL as follows:

```
rpm -i MySQL-server-4.0.22-0.i386.rpm MySQL-shared-4.0.22-0.i386.rpm MySQL
```

Initialize and setup your database server

Install the root database first and start the `mysql-daemon`. This step is apparently not required, because installing the rpm's already processes these steps. But just in case they don't, you can do manually:

```
mysql_install_db  
safe_mysqld
```

Configuring MySQL for Scarab

The following cookbook approach summarizes the process of tracking down connection problems with MySQL __and__ Scarab which almost ever turn out to be a matter of incorrectly configured privileges. I assume you have a clean MySQL installation on your system and you want to run the MySQL database and scarab on the same host namely `localhost`. Each step below goes into some degree of detail. Maybe not all described steps are necessary in your environment.

Step I (unix only) - Check the `/etc/hosts` file

If you plan to run MySQL on `localhost`, you must check the corresponding entry in your `/etc/hosts` file. We have seen problems when the `localhost` entry is defined as in the following example:

```
127.0.0.1 localhost.localdomain localhost (seen on debian)
```

```
127.0.0.1    linux localhost                (seen on Suse)
```

If you see similar entries, revert the order so that "localhost" appears first:

```
127.0.0.1    localhost localhost.localdomain    (seen on debian)
127.0.0.1    localhost linux                (seen on Suse)
```

You may need to reboot(!) although I am not sure about this. After reboot, ensure that your operating system has not manipulated the entry in your /etc/hosts file. I have no idea what to do if this happens... (FIXME: add a recipe for this situation)

Step II - Ensure MySQL is up and running

I assume you have just installed an instance of MySQL. MySQL comes with an administration user named "root" and with no password at all. Yes, this is highly insecure. However it helps getting started with MySQL quickly. We will come back to security in the next step.

For now just try to enter the database administration. You may do so from a commandline (dos box or windows shell):

```
mysql -u root mysql
```

Now you should have access to the administration database of your MySQL instance. If you are rejected, something out of the trail happened. Either your database is corrupted or someone else has installed the database and has set up security. In that case you may need to ask your administrator to give you access to the "mysql" database in order to proceed with the next step.

Step III - Secure your MySQL instance (optional)

Once you are logged in to the database either as root, or as some administrator (depending on your site) you can check the grants:

```
select host, user, password from user;
```

```
+-----+-----+-----+
| host      | user  | password |
+-----+-----+-----+
| localhost | root  |          |
+-----+-----+-----+
1 row in set (0.01 sec)
```

Now let's secure it. First, we want to secure the root user. Thus we give him/her a password, we choose "secret" here.



Note

This step is NOT necessary to get Scarab up and running, so if you don't want to secure your database now, just don't execute the following statement.

```
grant all on *.* to root@localhost identified by 'secret';
```

From now on you MUST remember your password for ever! The user table now looks as follows:

```
select host, user, password from user;
+-----+-----+-----+
| host      | user  | password |
+-----+-----+-----+
| localhost | root  | *14E65567ABDB5135D0CFD9A70B3032C179A49EE7 |
+-----+-----+-----+
1 row in set (0.00 sec)
```

Step IV - Add the admin user for Scarab

We choose *admin* as administrator using the password *secret* and furthermore we choose the database name to be *scarab*. Of course you can take your own choice.



Note

You will need ALL choices above later to add them to the `{{build.properties}}` of Scarab !

```
grant all on scarab.* to admin@localhost identified by 'secret';
grant grant option on scarab.* to admin@localhost;
grant all on mysql.* to admin@localhost;
```



Note

You need the second statement because the admin user will eventually create the runtime database user! (see below) The term `grant grant option` above is definitely correct! You also will need the third statement because your scarab user uses a password for identification. In order to be able to make the appropriate settings, the admin needs access to the mysql control tables.

This is how the user table looks like now:

```
select host, user, password from user;
+-----+-----+-----+
| host      | user  | password |
+-----+-----+-----+
| localhost | root  | *14E65567ABDB5135D0CFD9A70B3032C179A49EE7 |
| localhost | admin | *14E65567ABDB5135D0CFD9A70B3032C179A49EE7 |
+-----+-----+-----+
```

Step V - Test the settings



Caution

For safety keep the current shell open and DO NOT QUIT the mysql session. That helps if something went wrong ;-)

OK, now open a new shell and try to create a database from the admin user:

```
mysqladmin -u admin -h localhost -p create scarab
Enter password: *****
```

Now you have created your scarab database, though it's empty. If the database was already there you would have received an appropriate error message! In that case don't proceed until you have checked where this database came from!



Note

This step is NOT required for the Scarab setup, but doing so ensures that your privilege settings for the Scarab administration user are correct and work as needed.

Step VI - Configure scarab

Make a copy of minimal.properties (or project.properties) and name it build.properties. You will need to define a runtime database user for your instance now. Let's choose "scarab" as username and "baracs" as password. These two properties will be needed now in addition to the admin user created in step IV above (just keep them in mind for a minute)

Now edit build.properties. About the database settings, please get hands on these properties:

```
scarab.database.username=scarab      #changed from **GENERATED**
scarab.database.password=baracs      #changed from **GENERATED**
scarab.database.admin.username=admin  #changed from **GENERATED**
scarab.database.admin.password=secret #changed from **GENERATED**
scarab.database.host=localhost       #unchanged
scarab.database.name=scarab          #unchanged
```

Just to be complete: there are other relevant properties, but usually they are generated from the above settings. The only two additional properties you should take care of are:

```
scarab.database.type=mysql           #unchanged
scarab.database.port=**GENERATED**  #unchanged
```

Where the first one is set to mysql by default and the last one is set to the standard port 3306 used by MySQL for the connection peer.



Note

The database.username and password are later used to automatically create(!) an additional user 'scarab' with password 'baracs'. So don't bother yourself with setting up other users except the admin user !

Step VII - Build scarab

Go to the build directory and launch ant

```
ant
ant create-db
```

This should execute without any error now. You should see something like this in the shell:

```
[echo]
[echo]           R U N T I M E   B U I L D   ____
[echo]  /-----/-----\_____|_
[echo] \_____ \ /  ___ \ \ \  _ \  \ | _ \
```


low TCP/IP connections from the host where your Java code is running to the MySQL server on the port that MySQL is listening to (by default, 3306).

Configuring PostgreSQL

Due to a problem in the database persistence framework we use, Scarab will not work "out-of-the-box" with PostgreSQL. Fortunately, there is a simple script that you can run which will create a work-around.

This script must be run by a user with sufficient privileges, which generally means the user that originally created and initialised the PostgreSQL database. Typically this user will be 'postgres'.

The script can be found at `src/sql/postgresql/hack-db.sql` in the distribution, and it can be executed in different ways, for example from the command line:

```
postgres> psql scarab < hack-db.sql
```

where 'scarab' is the name of the Scarab database, and 'postgres' is the "system" PostgreSQL user. Alternatively, from the psql interactive console:

```
scarab=> \i hack-db.sql
```

Setting up and configuring Oracle

Getting the JDBC drivers

For licence reasons, the jar containing the Oracle JDBC driver is not included in the Scarab distribution.

You will have to get it yourself from your Oracle distribution (or from Oracle Technology Network: http://www.oracle.com/technology/software/tech/java/sqlj_jdbc/index.html). Unless you have a compelling reason to do otherwise, you should take the latest drivers: they are compatible with older releases of Oracle but are much faster than the older drivers.

The jar you need is probably named `classes12.jar` (for JDK 1.2 and 1.3) or `ojdbc14.jar` (for JDK 1.4 and 1.5). Copy this jar to the `$SCARAB/lib` directory (i.e., the lib directory directly at the root of your Scarab distribution. In a fresh installation of Scarab, this directory is probably empty, except for a README file. This is normal, it is populated during the build process.

Configuring access to your Oracle database

In the `build.properties` file you will prepare in Step 3 below, you will need to configure the access to your Oracle database.

```
scarab.database.host=localhost
scarab.database.port=
scarab.database.url=
scarab.database.admin.url=
scarab.database.type=oracle
scarab.database.jdbc.driver=oracle.jdbc.driver.OracleDriver
scarab.database.username=scarab
scarab.database.password=secret
scarab.database.admin.username=scott
scarab.database.admin.password=tiger
```

Solving a problem with LOBs

To use Large Objects (LOBs), such as BLOBs (Binary Large Objects) and CLOBs (Character Large Objects) you need to instantiate Oracle classes (not JDBC generic classes) and the current version of the persistence layer

we use, Torque, does not allow to do that at the moment. A workaround is to use large VARCHARs instead. To change this, you will need to patch the `webapps/scarab/WEB-INF/sql/turbine.sql` after building Scarab (see next section), replacing `BLOB` by `VARCHAR2(4000)`⁴

Using Oracle 10g XE on the same machine as Scarab

A word of caution if you plan to use Scarab on the same machine as Oracle 10g XE. Both Scarab and the Oracle administration web app listen by default on the same port (8080), which will cause a conflict.

To configure Scarab to listen on a different port, change the `scarab.http.port` property to a value you choose and edit the `./tomcat/conf.server.xml` file to configure Tomcat to listen to a different port.

To change the port on which the Oracle administration web app listens, consult the Oracle 10g XE documentation.

Building Scarab with the web install application

For a quick install, a test install or if you just want to experiment with Scarab, you will find the web installation simpler and much more user-friendly. This simplified process allows you to define the most current parameters and gives you short explanations and even some advice.

If you are a power user and want a fine-tuned installation or if your environment does not allow for the web installer, skip this section and install Scarab from the command line.

Step 1: Unpack the Scarab distribution

The Scarab release can be downloaded as either a `.zip` or a `.tar.gz` file. Unpack either one of these into a directory that you have write access to.

Step 2 : Unpack the web installer

Download the install application from the Scarab web site and unpack the web application at the same place you unpacked the Scarab distribution. You may be asked whether you accept to overwrite `$(SCARAB)/tomcat/conf/web.xml` : accept it.

Step 3: Configure and start Tomcat

With the default settings, Tomcat will start listening on the 8080 port. If this is a problem (e.g., because another program -- such as the Oracle XE admin application -- uses this port), edit the `$(SCARAB)/tomcat/conf/server.xml` file and change the port (line 3).

To start Tomcat, go to the `$(SCARAB)/tomcat/bin` directory and launch:

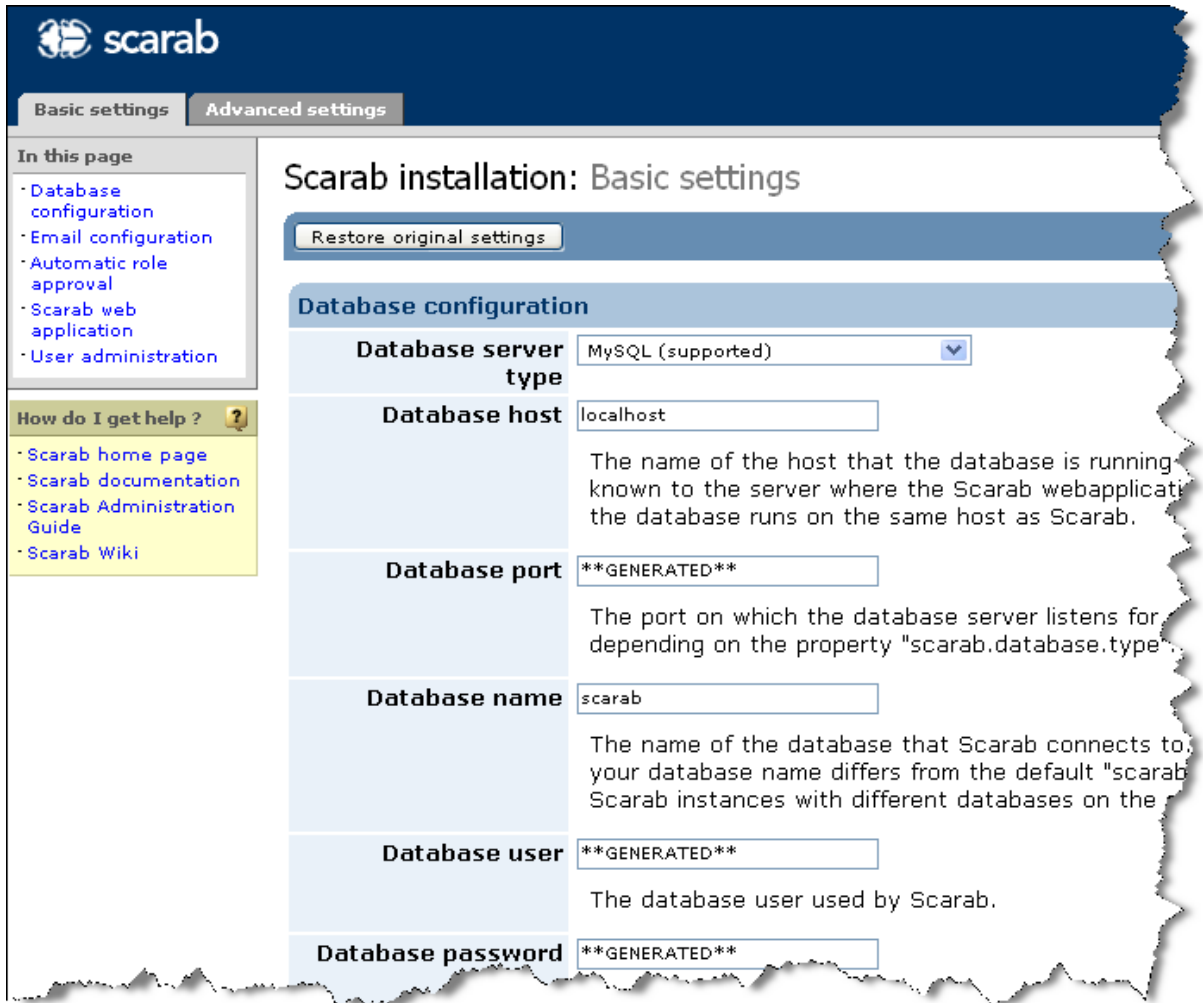
- `startup.sh` (Unix)
- `startup.bat` (Windows)

Step 4: Configure Scarab

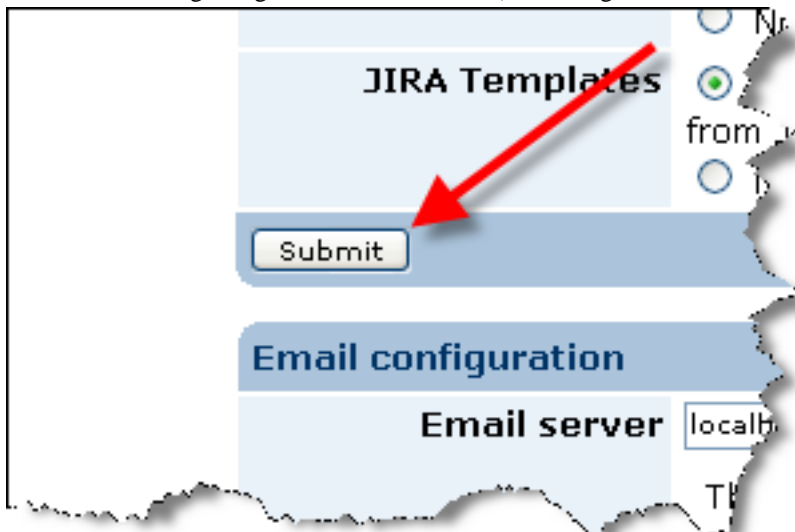
Now launch your favorite web browser and go to the following URL: `http://localhost:8080/ScarabWebInstaller`

You'll see the "Basic settings" page..

⁴You can as well edit `webapps/scarab/WEB-INF/src/torque/templates/sql/base/oracle/db.props` and change `VARBINARY = BLOB` to `VARBINARY = VARCHAR2(4000)` -- that is a slightly more persistent modification.



This web page has four forms. If you want to modify the default values for a parameter in a form, do not forget to save this setting using the "Submit" button (the settings are saved one block at a time and not page by page).



You may navigate back and forth between the "Basic settings" and "Advanced settings" tabs. When you are finished configuring the application as you wish, scroll down to the bottom of a page and click *Build Scarab now !*




The application writes down your parameters in the `build.properties` file, located in the root directory of your installation, generates a shell script to invoke Ant and executes this script.

The result appears on the next page:

If this worked as expected, you can proceed and create the Scarab database using the "Build the Scarab database" button.

```
BUILD SUCCESSFUL
Total time: 4 seconds
```

 L'opération a réussi



[carab documentation](#) | [Scarab Administration Guide](#)

ris.org

The application writes a second shell script to invoke Ant and launches this script. The result appears on the last page.

If this last operation was a success, the installation is almost over, you have just some cleanup to perform. In fact, Scarab may have been launched already by Tomcat but you really should stop, purge and restart Tomcat.

Step 5: Stop Tomcat

To stop Tomcat, go to the `$SCARAB/tomcat/bin` directory and execute:

- `shutdown.sh` (Unix)
- `shutdown.bat` (Windows)

Step 6: Remove the web installer

The web installer is unprotected and leaving it there would certainly create a security issue. If you want to leave your Scarab installation in a clean state, uninstall the web installer by removing the `ScarabWebInstaller` directory under `$SCARAB/tomcat/webapps`.

(If you are a paranoid admin, you have probably kept the `$SCARAB/tomcat/conf/web.xml` file around. Otherwise you still may retrieve an original version from the Scarab distribution. But this is not strictly required.)

Step 7: Restart Tomcat and launch Scarab

You may now restart Tomcat as has been explained in the Step 3 above.

Building Scarab: command-line installation



Note

This section is a formatted copy of the `INSTALL` file at the root of the Scarab distribution.

Step 1: Unpack the Scarab distribution

The Scarab release can be downloaded as either a `.zip` or a `.tar.gz` file. Unpack either one of these into a directory that you have write access to.

Step 2: Migrating an old (pre b19) version of Scarab

Please, read the 2.1 section of the `MIGRATION` file for instructions on the migration of your current settings.

If you don't have an existing pre-b19 Scarab installation, or you don't want to keep your old settings and data, then you can skip this step.

Step 3: Configure Scarab

If you already have an old `build.properties` file from a previous version of Scarab, please copy this `build.properties` file to the `$SCARAB_ROOT` directory (NOT the build directory) of your new installation.

If you don't already have an old `build.properties` file, create one as follows:

1. Go to your `$SCARAB_HOME` directory
2. copy (not rename!) one of the following two files to `build.properties`:
 - `project.properties` (full customization property set)

minimal.properties (minimal customization property set)

You will choose project.properties as your starting point, if you are a developer who is deeply involved in- to the Scarab development. In all other cases minimal.properties is your best choice (believe us!). So most of you will be doing this:

```
cd $SCARAB_HOME  
  
cp minimal.properties build.properties
```

3. Scroll through your build.properties and be sure to follow the notes. If you really miss a property which is not yet available in your build.properties just add it to your build.properties!

Step 4: Build Scarab

Go to the \$SCARAB_ROOT/build directory.

Launch ant::

```
ant
```



Note

If you are a developer, particularly a Scarab developer, you may also build Scarab with Maven 1.

This will give you additional features and options, such as running unit tests.

For detailed information, see the chapter 3 in the Developer's Guide.

Step 5a: Create the database



Warning

ONLY IF YOU'RE NOT MIGRATING AN EXISTING INSTANCE OF SCARAB !!

Read the 'MIGRATION' file if you need to preserve your data !!

Now you can create and populate the Scarab database.

Just run one of the commands bellow:

If you are using ant:

```
ant create-db
```

or

```
ant -Dskip.seed.data=true create-db
```

The second command avoids the population of your database with the Scarab sample data. If you are just testing Scarab out, then we recommend that you use the first option and load the sample data into the database.

If you are using maven, similar commands exist:

```
maven scarab:create-db
```

```
maven -Dskip.seed.data=true scarab:create-db
```

JIRA templates

By default, a module is created with issue-types that match the structure of JIRA, enabling transfer of issues from JIRA to Scarab. If you don't want this extra module in your issue tracker, then you can pass the option "-Dskip.jira.templates.data=true" to ant or maven, e.g.

```
ant -Dskip.jira.templates.data=true create-db
```

or

```
maven -Dskip.jira.templates.data=true scarab:create-db
```



Note

This option is independent of 'skip.seed.data', so either or both may be specified.

Step 5b: Migrate your current data



Note

ONLY IF YOU ARE MIGRATING AN EXISTING INSTANCE OF SCARAB !!

Go to MIGRATION file for instructions on migrating your existing data.

Step 6: Start Tomcat

On Windows, double-click on the file `$$SCARAB_HOME/tomcat/bin/startup.bat`

On Unix/Linux, run the following shell script: `$$SCARAB_HOME/tomcat/bin/startup.sh`

The webapplication will be available as reflected by the settings of the following properties (see step 3):

- `scarab.http.scheme`
- `scarab.http.domain`
- `scarab.http.port`
- `scarab.context`

In general your Scarab will be available from a browser on the URL: `/${scarab.http.scheme}://${scarab.http.domain}:${scarab.http.port}/${scarab.context}`

Consequently in case you did NOT change the default properties, you now will find Scarab by pointing your Browser to: `http://localhost:8080/scarab`

If your setup is ok, you will be presented with the Scarab login page after one initial redirect. If that happens, congratulations! You now have Scarab installed and running on your system.

If any of the above steps fail, then please read the full README.txt file. If you are still having problems then please subscribe to the Scarab users mailing list at: `users-subscribe@scarab.tigris.org`

Once you are subscribed, send a message to `users@scarab.tigris.org` detailing your problem.

Chapter 2. Scarab system administration after installation

Post-installation administration

Setting up an Apache 2 Virtual Host

The following example (with inline explanations) should help you to set up an Apache 2 Virtual Host in front of Scarab.

```
# Scarab virtual host configuration for use with Apache2
#
# S.P.James mailto:steATcpanDOTorg
#
# This script is released under the same terms as Scarab itself
# See http://scarab.tigris.org/LICENSE
#
# On Debian Sarge, put this script here:
#
#   /etc/apache2/sites-enabled/020-scarab
#
# This is a Scarab virtual host configuration for use with Apache2. Assuming a
# typical Scarab application running on a Tomcat instance, its URL would be,
# say, http://scarab:8080/scarab/issues. This vhost configuration allows the
# shorter URL http://scarab to be used instead. It also enables the short-
# cut URL like this http://scarab/PVS42 to refer to the given issue ID.
# What's more, you can use Apache's access control mechanisms.
#
# I've used the name "scarab" for the hostname in this example. Make sure
# you have forward and reverse DNS records for whatever virtual host name
# you choose to use.
#
# See also the Apache HTTP Server Version 2.0 manual, "Name-based Virtual
# Host Support" section.

<VirtualHost *>
    # Fully qualified name for this virtual server
    ServerName scarab.beachsolutions.plus.com
    # Alternative name(s) for this server
    ServerAlias scarab

    ServerAdmin scarab@beachsolutions.plus.com

    # Log errors here
    ErrorLog /var/log/apache2/scarab.error.log

    # Possible values include:
    #   debug, info, notice, warn, error, crit, alert, emerg.
    LogLevel warn

    DocumentRoot /var/local/scarab

    ProxyRequests Off

    # This section defines the external location /scarab as proxy
    # to the Scarab instance running on a Tomcat server
    <Location /scarab>
        # Configure access to /scarab. Permit our network and local
        Order deny,allow
        Deny from all
        Allow from 127.0.0.1
        Allow from .beachsolutions.plus.com
```

```
Allow from ::1/128

# Pass this location to the Scarab instance running on port 8080
ProxyPass          http://scarab.beachsolutions.plus.com:8080/scarab
ProxyPassReverse   http://scarab.beachsolutions.plus.com:8080/scarab

</Location>

RewriteEngine On
# This enables a visitor to use http://scarab/SCB42 to get to an issue
RewriteRule ^/([A-Za-z]{1,4}\d+)$ scarab/issues/id/$1 [last,redirect]
# This enables a visitor to use just http://scarab to begin
RewriteRule ^/$ scarab/issues [last,redirect]

</VirtualHost>
```

Backups and Restores

To restore a Scarab installation, if the server fails for instance, you must perform backups on a regular basis.

You will have to backup the data contained in the Scarab database, but also the directories that contain the attachments, the indices used by Lucene for full-text search and the objects serialized by Intake.

Backing up Scarab information

Backing up the database

MySQL

On the command line, type for instance:

```
mysqldump scarab > filename.sql
```

If your MySQL server requires authentication, you will have to supply the appropriate arguments on the command line, for instance:

```
mysqldump scarab --user=scarab --password > filename.sql
```

Backing up other information

The directories used to store attachments and Lucene indices, and the file containing the objects serialized by Intake have been defined at build time. The corresponding properties are:

- `scarab.attachments.path`
- `scarab.lucene.index.path`
- `scarab.intake.serialize.file`

You need to back up these two directories and this file as well as the database information.

If you hadn't redefined the three parameters above when you built Scarab, all these objects are in the Scarab file hierarchy under `target/webapps/scarab/WEB-INF`.

On Windows, the simplest solution is probably to copy the two directories and the Intake file in a zip archive.

On Linux, you can do the same with the following command (for those not familiar with unix commands):

```
tar zcf scarab.tgz attachments index intake-xml.ser
```

If one or several of the parameters has/have been redefined, you will need to find where the directories and the file are to be able to archive them.

Restoring Scarab information from a backup

Restoring the database

MySQL

Connect to MySQL, for instance:

```
mysql
```

You may have to specify a user name and/or password if your MySQL server requires authentication:

```
mysql --user=name --password
```

If necessary, you may have to delete a corrupted Scarab database:

```
drop database scarab;
```

Create a new Scarab database and select it:

```
create database scarab;
```

```
use scarab;
```

Restore the data you had backed up with mysqldump:

```
source filename.sql;
```

That's it!

Restoring other information

Uncompress the archive you made previously with attachments, Lucene indices and objects serialized by Intake, in the directories defined when Scarab was built, as explained above.

Your Scarab installation is now ready to restart.

Moving an existing Scarab installation to another server

If your Scarab server fails -- or for any other valid admin reason -- you may want to move an existing Scarab installation to another server.

Unless the settings and the network name of the new machine are strictly identical, you will need to perform a new Scarab installation from scratch, tuning a few configuration parameters.

If the network name of the target machine differs from the original one, you will also need to perform three modifications in the database, even if the database server itself does not change.

Modifying the database when the Scarab server changes

Modifying the SCARAB_GLOBAL_PARAMETER table

If you change any of the following parameters on the new installation: `scarab.http.domain`, `scarab.http.scheme`, `scarab.http.scriptname` or `scarab.http.port`, you will need to update the corresponding entries in the `SCARAB_GLOBAL_PARAMETER` table. For instance:

```
update SCARAB_GLOBAL_PARAMETER set VALUE='newserver.example.com' where  
NAME='scarab.http.domain';
```

```
update SCARAB_GLOBAL_PARAMETER set VALUE='8081' where  
NAME='scarab.http.port';
```

Modifying the SCARAB_MODULE table

You may need to update the domain (server) name in the SCARAB_MODULE table:

```
update SCARAB_MODULE set DOMAIN='newserver.example.com';
```

Modifying the SCARAB_ISSUE table

You may need to update the domain (server) name in the SCARAB_ISSUE table:

```
update SCARAB_ISSUE set ID_DOMAIN='newserver.example.com';
```

Chapter 3. Using Scarab with Subversion

Abstract

Integrating an issue tracking tool (Scarab) and a configuration management tool (Subversion) is a quest for the Grail for most developers.

Well, the good news is that it's definitely possible to make these work together.

Until this chapter is written, you may refer to the documentation written by Mick Wever and available at this URL: <http://www.wever.org/java/space/java/Integrating+Subversion+and+Scarab>