




Key Performance Indicator			
			
Line Counting			
Lines of Code	481219	↗	✓
Source Lines Of Code	401326	↗	✓
Effective Lines Of Code	325196	↗	✓
Cyclomatic Complexity	12265	↗	✓
Comment Rate	16 %	↘	✗
Decision Making			
Business Value	1912	→	I
Technical Debt	3626	↗	I
Maturity Index	65 %	↘	C
Stability Index	84 %	↗	B
Reusability Index	44 %	↘	D

Square 16.3.0

Installation Checklist

Reference : SIM_QUICK_Square
Version : 16.3.0
Date : 28/04/2017

Installation Checklist

Copyright © 2017 Squoring Technologies

Table of Contents

1. Introduction	1
1.1. The Squore Ecosystem	1
2. Preparing your Machine for Squore	2
2.1. Installation Prerequisites	2
2.1.1. Supported Operating Systems	2
2.1.2. Supported Database Management Systems	2
2.1.3. Browser Compatibility	2
2.1.4. For All Systems	3
2.1.5. Prerequisites for Oracle	3
2.1.6. Obtaining a Licence File	4
2.1.7. Packages for Windows	4
2.1.8. Packages for Linux	4
2.1.9. Packages for CentOS and Red Hat Enterprise Linux	5
2.1.10. Packages for Ubuntu	7
2.2. Third-Party Plugins and Applications	8
2.3. Running Squore as a Windows Service	8
2.3.1. Services Configuration	8
2.4. Requirements for LDAP Integration	9
2.5. Requirements for TeamForge Integration	10

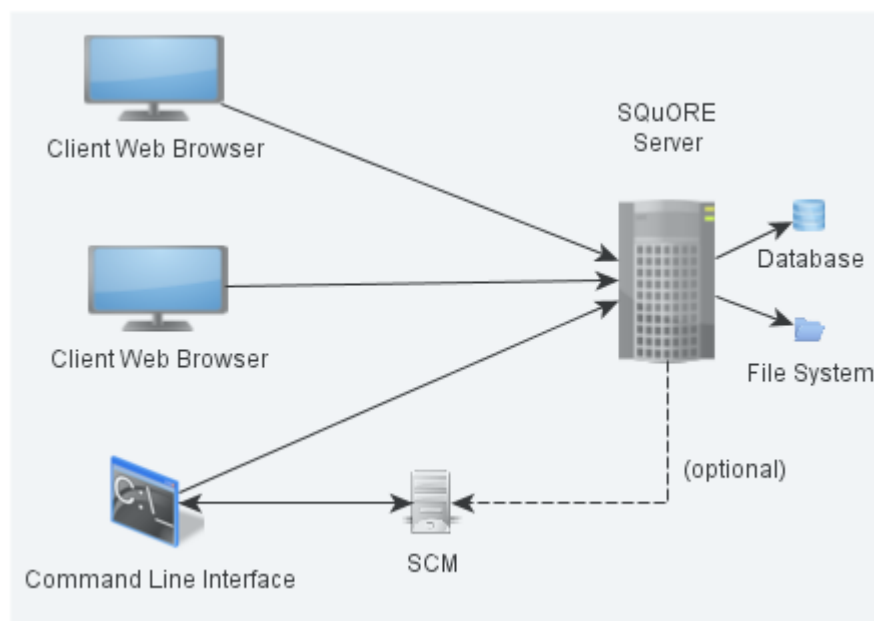
1. Introduction

This document is a checklist and FAQ sheet you can use to make sure that your environment is ready for installing Squore. The first chapter covers common pre-requisites and concepts of the Squore architecture. Each of the following chapters covers more specific topics that you only need to read about if you are interested in using the proposed solutions.

1.1. The Squore Ecosystem

Squore is based on a traditional 3-tier architecture consisting of:

- a database and a data folder for storing project data
- an application server
- a client front-end accessible through a Web Browser
- a Command Line Interface (Squore CLI) to interact with the server from a client machine



The Squore Architecture

As shown in the schema above, Squore Server can provide analysis results to clients without having access to any source code, in scenarios where the analysis is carried out on a client machine with access to the SCM repository, as is the case in most Continuous Integration environments.

If you are planning to access source code hosted in a Subversion, Git, ClearCase, CVS or Synergy repository, a command line client for this repository must be available on the machine where the Squore analysis is carried out. For complete information about all installation pre-requisites, consult Section 2.1, "Installation Prerequisites".

Squore allows analysing source code in the following programming languages: ABAP, Ada, C, COBOL, C++, C#, Fortran 77, Fortran 90, Java, JavaScript, Lustre, Mind-C, Objective-C, PHP, PL/SQL, Python, T-SQL, Visual Basic .NET, XAML.

2. Preparing your Machine for Squore

This chapter only covers how to ensure that your environment can be prepared for a Squore installation. For actual instructions on how to install Squore itself, refer to the full Installation and Administration Guide.

2.1. Installation Prerequisites

2.1.1. Supported Operating Systems

The following is a list of the officially supported and tested operating systems:

- CentOS 6
- CentOS 7
- Fedora 19
- Ubuntu Server 14.04
- Windows 7
- Windows 8
- Windows 10

The following is a list of the operating systems that are not regularly tested but are known to be working:

- RedHat EL 6
- RedHat EL 7
- SuSe Linux 11.1
- Ubuntu Server 10.04
- Ubuntu Server 16.04
- Windows Server 2008 R2
- Windows Server 2012 R2

2.1.2. Supported Database Management Systems

Squore Server can use the following database management systems to store its data:

- PostgreSQL 8.4 and up
- Oracle Database 12c Release 1

In both cases, it is possible to have database on the same machine as Squore Server or on a remote machine.

Note

When using a database backend on a remote machine, the database administrator is responsible for backing up the database. The backup scripts included in the Squore Server installation will only handle the backup of the data stored on the Squore Server file system in that case. You can find more information about backup strategies for Squore Server in the section called **Backup Tools** in the Installation and Administration Guide.

2.1.3. Browser Compatibility

Squore is compatible with many browsers. The following is the list of officially supported browsers:

- **Google Chrome 24 and up**

- **Mozilla Firefox 17 and up**
- **Apple Safari 6 and up**
- **Microsoft Internet Explorer 8 and up**
- **Microsoft Edge 20 and up**

2.1.4. For All Systems

For a successful installation of Squore, you will need:

- The latest version of the Squore Server installer, which can be downloaded from http://support.squoring.com/download_area.php
- The latest version of the Squore CLI installer, which can be downloaded from http://support.squoring.com/download_area.php for installations where only a command-line client is needed
- A user account with system administrator privileges
- The Oracle Java Development Kit version 1.7.0_51 or higher

Warning

- Java 1.8 is not supported.
- It is technically possible to run Squore using a 32-bit JDK, however this will limit the memory available to 1GB of RAM to run the application, which will result in poor performance. If you still want to attempt such an installation, consult the troubleshooting page at http://openwiki.squoring.com/index.php/Running_Squore_On_A_32-bit_Java_Installation
- The Oracle Java Runtime Environment version 1.7.0_51 or higher for Squore CLI
- At least 2 GB of space available on the disk for a full installation
- At least 8 GB of RAM on the server machine
- At least 4 GB of RAM on the client machine
- A valid Squore Server licence file (optional, since the licence file can be added after installation)
- The `java` executable should be in the machine's PATH environment variable for Squore CLI to run successfully.

Tip

Keep in mind that the requirements above are the strict minimum. In production, Squore Server generally runs on a dedicated machine with a multi-core processor and 8 to 12GB of RAM. Squore reserves 25% of the available RAM of the machine to the database and another 25% to the server. External processes (like CheckStyle or Findbugs) running on the same machine as Squore may add to the amount of RAM required for analysing source code. Linux is known to offer better performances than Windows when running Squore.

2.1.5. Prerequisites for Oracle

When using Oracle as a database backend, a database administrator must create an Oracle user before you can install Squore.

The user requires the following privileges:

- **CREATE PROCEDURE**
- **CREATE SEQUENCE**
- **CREATE SESSION**
- **CREATE TABLE**

→ **CREATE TYPE**

→ **CREATE VIEW**

→ A valid quota for the tablespace used by the user (for example **UNLIMITED**)

2.1.6. Obtaining a Licence File

Because a licence is linked to the hardware on which Squore Server is installed, Squoring usually delivers a temporary licence to try out Squore first. This give you time to send out your host-id to our team, who will then issue a permanent licence file. In order to find out what your host-id is, follow these steps:

1. Download the host-id checker from http://support.squoring.com/download_area.php
2. Open a terminal on the machine where you installed or plan to install Squore.
3. Run the command `java -jar squore-hostid.jar`.
4. Send the output to support@squoring.com.

2.1.7. Packages for Windows

A JDK is required for Squore Server. The Windows installer contains the tcl and perl runtimes as well as a portable PostgreSQL installation.

A JRE is required for Squore CLI. The Windows installer contains the tcl and perl runtimes needed. It will allow you to obtain the configuration needed to create projects from the server.

2.1.8. Packages for Linux

On Linux platforms, the following must be installed before installing Squore:

→ **Perl** version 5.10.1 or greater including the following extra-modules:

→ Mandatory packages:

- **Algorithm::Diff** [module details] [<http://search.cpan.org/~nedkonz/Algorithm-Diff/lib/Algorithm/Diff.pm>]
- **Archive::Zip** [module details] [<http://search.cpan.org/~phred/Archive-Zip/lib/Archive/Zip.pm>]
- **Date::Calc** [module details] [<http://search.cpan.org/~stbey/Date-Calc/lib/Date/Calc.pod>]
- (Squore Server only) **DBD::Pg** (unless you use a Oracle as your database backend) [module details] [<http://search.cpan.org/~turnstep/DBD-Pg/Pg.pm>]
- (Squore Server only) **DBI** (unless you use a Oracle as your database backend) [module details] [<http://search.cpan.org/~timb/DBI/DBI.pm>]
- **Digest::SHA** [module details] [<http://search.cpan.org/~mshelord/Digest-SHA/lib/Digest/SHA.pm>]
- **HTTP::Request** [module details] [<http://search.cpan.org/~gaas/HTTP-Message/lib/HTTP/Request.pm>]
- **JSON** [module details] [<http://search.cpan.org/~makamaka/JSON/lib/JSON.pm>]
- **LWP** [module details] [<http://search.cpan.org/~ether/libwww-perl/lib/LWP.pm>]
- **LWP::UserAgent** [module details] [<http://search.cpan.org/~gaas/libwww-perl/lib/LWP/UserAgent.pm>]
- **Time::HiRes** [module details] [<http://search.cpan.org/~zefram/Time-HiRes/HiRes.pm>]
- **XML::Parser** [module details] [<http://search.cpan.org/~toddr/XML-Parser/Parser.pm>]

→ Optional packages for working with Microsoft Excel:

- **HTML::Entities** [module details] [<http://search.cpan.org/dist/HTML-Parser/lib/HTML/Entities.pm>]
- **Spreadsheet::BasicRead** [module details] [<http://search.cpan.org/~gng/Spreadsheet-BasicRead/BasicRead.pm>]
- Optional packages for working with OSLC systems:
 - **Date::Parse** [module details] [<http://search.cpan.org/~gbarr/TimeDate/lib/Date/Parse.pm>]
 - **WWW::Mechanize** [module details] [<http://search.cpan.org/~ether/WWW-Mechanize/lib/WWW/Mechanize.pm>]
 - **XML::LibXML** [module details] [<http://search.cpan.org/~shlomif/XML-LibXML/LibXML.pod>]
- Optional packages for working with GitHub systems:
 - **Date::Parse** [module details] [<http://search.cpan.org/~gbarr/TimeDate/lib/Date/Parse.pm>]
 - **Mail::Box::Manager** [module details] [<http://search.cpan.org/~markov/Mail-Box/lib/Mail/Box/Manager.pod>]
 - **Mail::Message::Body::Lines** [module details] [<http://search.cpan.org/~markov/Mail-Box/lib/Mail/Message/Body/Lines.pod>]
 - **Mail::Message::Construct** [module details] [<http://search.cpan.org/~markov/Mail-Box/lib/Mail/Message/Construct.pod>]
 - **Mail::Mbox::MessageParser** [module details] [<http://search.cpan.org/~dcoppit/Mail-Mbox-MessageParser/lib/Mail/Mbox/MessageParser.pm>]
 - **Net::GitHub** [module details] [<http://search.cpan.org/~fayland/Net-GitHub/lib/Net/GitHub.pm>]
- Optional packages for working with Semios/Prometil systems:
 - **File::Slurp** [module details] [<http://search.cpan.org/~uri/File-Slurp/lib/File/Slurp.pm>]
- Optional packages for Advanced CSV Export Management:
 - **Text::CSV** [module details] [<http://search.cpan.org/~makamaka/Text-CSV-1.33/lib/Text/CSV.pm>]

Tip

If some of these modules are not available as packages on your operating system, use your perl installation's cpan to install the modules. Using the OS packages is recommended, as it avoids having to reinstall via cpan after upgrading your version of perl.

- **Tcl** version 8.5 or greater,
- (Squore Server only) **PostgreSQL** version 8.4 (unless you use a RDBMS running on another system) including at least the **server** component, and optionally, the **pgAdmin** utility. Note that your system must use a UTF-8 locale for the database creation to be carried out successfully. You can force this by running `export LANG=en_US.UTF-8` or `export LANG=fr_FR.UTF-8` according to what is available on your system before installing Squore.
- (Squore Server only) The **rsync** utility

If you are running on a headless Squore Server, java-1.6.0-openjdk may not be sufficient, as it lacks some fonts to render graphics. This is why using Oracle's JDK is recommended.

2.1.9. Packages for CentOS and Red Hat Enterprise Linux

On Red Hat Enterprise Linux and CentOS (6.5 and 7.1), the dependencies are satisfied by the following packages:

Mandatory packages:

- **java-1.7.0-openjdk**

- **perl**
- **perl-Algorithm-Diff**
- **perl-Archive-Zip**
- **perl-Date-Calc**
- **perl-Digest-SHA**
- **perl-JSON**
- **perl-libwww-perl**
- **perl-Time-HiRes**
- **perl-XML-Parser**
- (Squore Server only) **postgresql-server** (unless you use a RDBMS running on another system)
- (Squore Server only) **rsync**
- **tcl**

Optional packages for working with Microsoft Excel:

- **perl-HTML-Parser**
- **perl-CPAN** (CPAN utility requirement)
- **perl-Spreadsheet-ParseExcel** (available in the EPEL repository)
- **perl-Spreadsheet-XLSX** (available in the EPEL repository)

Warning

The module **Spreadsheet::BasicRead** is not available as a package and must therefore be installed using cpan (make sure cpan is properly configured, by running **cpan** without arguments first):

```
sudo cpan -i Spreadsheet::BasicRead
```

Optional packages for working with OSLC systems:

- **perl-TimeDate**
- **perl-WWW-Mechanize** (available in the EPEL repository)
- **perl-XML-LibXML**

Optional packages for working with GitHub systems:

- **perl-TimeDate**
- **perl-Mail-Box** (available in the EPEL repository)
- **perl-Mail-Mbox-MessageParser** (available in the EPEL repository)
- **perl-Net-GitHub** (available in the EPEL repository)

Optional packages for working with Semios/Prometil systems:

- **perl-File-Slurp**

Optional packages for Advanced CSV Export Management:

- **perl-Text-CSV** (available in the EPEL repository)

For more information about how to install the Extra Packages for Enterprise Linux (EPEL) repository, consult <https://fedoraproject.org/wiki/EPEL>.

2.1.10. Packages for Ubuntu

On Ubuntu 14.04.3 LTS, the dependencies are satisfied by the following packages:

Mandatory packages:

- **libalgorithm-diff-perl**
- **libarchive-zip-perl**
- **libdate-calc-perl**
- (Squore Server only) **libdbd-pg-perl** (unless you use a Oracle as your database backend)
- (Squore Server only) **libdbi-perl** (unless you use a Oracle as your database backend)
- **libhttp-message-perl**
- **libjson-perl**
- **libwww-perl**
- **libxml-parser-perl**
- **openjdk-7-jdk**
- **perl**
- (Squore Server only) **postgresql** (unless you use a RDBMS running on another system)
- (Squore Server only) **rsync**
- **tcl**

Optional packages for working with Microsoft Excel:

- **make** (CPAN utility requirement)
- **libhtml-parser-perl**
- **libspreadsheet-parseexcel-perl**
- **libspreadsheet-xlsx-perl**

Warning

The module **Spreadsheet::BasicRead** is not available as a package and must therefore be installed using **cpan** (make sure **cpan** is properly configured, by running **cpan** without arguments first):

```
sudo cpan -i Spreadsheet::BasicRead
```

Optional packages for working with OSLC systems:

- **libtimedate-perl**
- **libwww-mechanize-perl**
- **libxml-libxml-perl**

Optional packages for working with GitHub systems:

- **libtimedate-perl**
- **libmail-box-perl**
- **libmail-mbox-messageparser-perl**
- **libnet-github-perl**

Optional packages for working with Semios/Prometil systems:

→ **libfile-slurp-perl**

Optional packages for Advanced CSV Export Management:

→ **libtext-csv-perl**

2.2. Third-Party Plugins and Applications

End users can run third-party static code analysers or rule checkers that are not shipped with the Squore installer for licencing reasons. In this case, it is necessary to download the extra binaries from http://support.squoring.com/download_area.php and deploy them on the server.

The list of third party plugins to be downloaded separately is as follows:

- Checkstyle 5.6
- CPD 4.2.6
- FindBugs 2.0 or 3.0
- Cppcheck 1.61
- PMD 5.0.5
- Polyspace Export
- Stylecop 4.7

Here is a full example of how to deploy Checkstyle into Squore

1. Download the Checkstyle binary from http://support.squoring.com/download_area.php.
2. Extract the contents of the zip file onto Squore Server.
3. Copy the extracted checkstyle-5.6 folder into <SQUORE_HOME>/addons/tools/CheckStyle_auto.
4. Instruct all client installations to synchronise with the server so that they get the newly deployed third-party binaries.

If you have deployed some third-party tools on Squore Server, they will automatically be downloaded to your client when you launch the client synchronisation script.

Tip

AntiC and Cppcheck on Linux also require special attention: Cppcheck must be installed and available in the path, and antiC must be compiled with the command:

```
# cd <SQUORE_HOME>/addons/Antic_auto/bin/ && gcc antic.c -o antic
```

For more information, refer to the Command Line Interface Manual, which contains the full details about special installation procedures for Data Providers and Repository Connectors.

2.3. Running Squore as a Windows Service

During the installation of Squore on Windows, the installer provides the option to register two services to control Squore's startup and shutdown.

2.3.1. Services Configuration

After installing Squore Server and the Windows services, you should follow these steps on your server:

1. Configure both services to run with a dedicated user instead of the built-in LocalSystem

2. Ensure that the dedicated user you assigned to run the services is explicitly granted the **Full Control** permission on the following folders:
 - <SQUORE_HOME>
 - <SQUORE_DATA>\cluster
 - <SQUORE_DATA>\backup
 - <SQUORE_DATA>\projects
 - <SQUORE_DATA>\temp
 - <SQUORE_DATA>\temp\sources

By explicitly granting this permission instead of inheriting it from a group, you ensure that you will not run into issues with folder and file permissions, especially if User Account Control is enabled on your server.
3. Set the startup type for both services to **Run automatically**

Tip

You can manage and configure the two new services in the Windows Services console or from the command line. If you want to change their display names for example, you can use the following Windows command as administrator:

```
sc config "JBAS71SVC" DisplayName= "Squore Web"
sc config "PG84SVC" DisplayName= "Squore DB"
```

2.4. Requirements for LDAP Integration

In order to configure Squore to integrate with your LDAP Server, you should make sure that you have access to the following information:

- The address of the LDAP server you want to connect to Squore.
- The section(s) of the directory that contain the users that should be allowed to log into Squore.
- The login and password of a user account allowed to browse the section(s) of the directory mentioned above.
- Basic knowledge of your directory structure. Note that Squore was tested with Microsoft Active Directory on Windows Server 2008 and OpenLDAP on Ubuntu 12.04.

If you need to obtain this information to a system administrator, ask for these details:

- **java.naming.provider.url**: The URL of the directory server.
- **baseCtxDN**: The fixed DN of the context to start the user search from.
- **bindDN**: The DN used to bind against the ldap server for the user and roles queries. This is some DN with read/search permissions on the baseCtxDN and rolesCtxDN values.
- **bindCredential**: The password for the bindDN
- **baseFilter**: The search query sent by Squore to the LDAP server when authenticating. If the password is correct and the search returns true, the user is allowed to log into Squore. The default query checks that the login exists, but you can change it to check that the login is valid and that the user is part of a specific group for example, using the syntax `&((condition1)(condition2))`. For more information about LDAP query syntax, refer to [https://technet.microsoft.com/en-us/library/aa996205\(v=exchg.65\).aspx](https://technet.microsoft.com/en-us/library/aa996205(v=exchg.65).aspx). Note that the **&** characters must be written as an entity (**&**) in the settings file.
- **rolesCtxDN**: The fixed DN of the context to search for user roles. This is required to exist, even though it is not used by Squore at the moment.
- **userCompositeName (optional)**: the field in the LDAP account that Squore will import and use as the user's full name.

- **userMail (optional)**: the field in the LDAP account that Squore will import and use as the user's e-mail address.
- **userOrganizationUnit (optional)**: the field in the LDAP account that Squore will import and use as the user's department.
- **userId (optional)**: the field in the LDAP account that Squore will use as the final user login to create the account or log into the application.

2.5. Requirements for TeamForge Integration

In order to configure Squore to integrate with TeamForge, you will need to provide the following TeamForge details:

- **TeamForge Server URL (without / at the end)**, for example `http://localhost`.
- **TeamForge Server Name (will appear in Squore export format list)**, for example `TeamForge`.
- **TeamForge artifact default status at export**, for example `Open`.
- **TeamForge artifact default priority at export**, for example `4`.
- **SvnEdge Main Repository URL**, for example `http://localhost/svn`.
- **SvnEdge Viewer (viewvc) URL**, for example `http://localhost/viewvc`.
- **External System Id of SvnEdge (for ex: exsy1001)**, for example `exsy1011`.

The rest of the configuration is carried out on the TeamForge side with data obtained from Squore after you create some projects. You can see this information in the full Installation and Administration Guide