


Space Details

Key:	CROWD
Name:	Crowd
Description:	Single Sign-On and Identity Management
Creator (Creation Date):	justen.stepka@atlassian.com (Sep 28, 2006)
Last Modifier (Mod. Date):	justen.stepka@atlassian.com (Feb 25, 2007)

Available Pages

- Documentation 
 - 0. - Preface
 - 0.1 - Concepts
 - 0.2 - FAQ
 - 1. - Installation Guide
 - 1.1 - Release Information
 - 1.0
 - Beta Release 0.2
 - Beta Release 0.3
 - Beta Release 0.3.2
 - Beta Release 0.3.3
 - Beta Release 0.4
 - Beta Release 0.4.1
 - Beta Release 0.4.2
 - Beta Release 0.4.3
 - Beta Release 0.4.4
 - Beta Release 0.4.5
 - Release 1.0.0
 - Release 1.0.1
 - 1.2 - Requirements
 - 1.3 - Configuring Crowd
 - 1.3.1 - HSQL DB
 - 1.3.2 - MS SQL Server
 - 1.3.3 - MySQL
 - 1.3.3 - PostgreSQL
 - 1.5 - Setup Wizard
 - 1.6 - Deployment FAQ
 - Self Signed Certificate
 - 1.7 - Upgrading Crowd
 - 2. - Administration Console
 - 2.1 - About the Administration Console

- 2.2 - Options and Settings
 - 2.2.1 - Caching
 - 2.2.2 - General Options
 - 2.2.3 - Licensing
 - 2.2.4 - Mail Server
 - 2.2.5 - Mail Template
 - 2.2.6 - Session
- 2.3 - Directory Manager
 - 2.3.1 - Directory Browser
 - 2.3.2 - Adding a Directory Connector
 - 2.3.3 - Internal Directory Connectors
 - 2.3.4 - LDAP Directory Connectors
 - 2.3.5 - Custom Directory Connectors
- 2.4 - Application Manager
 - 2.4.1 - Application Browser
 - 2.4.2 - Adding an Application
 - 2.4.3 - Managing an Integrated Application
- 2.5 - Principal Manager
 - 2.5.1 - Principal Browser
 - 2.5.2 - Adding a Principal
 - 2.5.3 - Managing a Principal
- 2.6 - Group and Role Manager
 - 2.6.1 - Group and Role Browser
 - 2.6.2 - Adding a Group or Role
 - 2.6.3 - Managing a Group or Role
- 3. - Integration Guide
 - 3.1 - Integration Overview
 - 3.2 - Client Configuration
 - 3.3 - Integration Libraries
 - 3.4 - SOAP API
 - 3.5 - .NET Client
- 4. - Connectors
 - 4.1 - Supported Applications and Directories
 - 4.2 - Integrating Crowd with Apache
 - 4.3 - Integrating Crowd with Bamboo
 - 4.4 - Integrating Crowd with Fisheye
 - 4.5 - Integrating Crowd with Confluence
 - 4.6 - Integrating Crowd with JIRA
 - 4.7 - Integrating Crowd with Jive Forums
- 5. - External Resources
- Navigation
 - Blogs

Documentation

This page last changed on Mar 07, 2007 by rosie@atlassian.com.

Crowd Documentation	
<ul style="list-style-type: none">• 0. - Preface<ul style="list-style-type: none">◦ 0.1 - Concepts◦ 0.2 - FAQ• 1. - Installation Guide<ul style="list-style-type: none">◦ 1.1 - Release Information◦ 1.2 - Requirements◦ 1.3 - Configuring Crowd◦ 1.4 - Database Configuration◦ 1.5 - Setup Wizard◦ 1.6 - Deployment FAQ◦ 1.7 - Upgrading Crowd• 2. - Administration Console<ul style="list-style-type: none">◦ 2.1 - About the Administration Console◦ 2.2 - Options and Settings◦ 2.3 - Directory Manager◦ 2.4 - Application Manager◦ 2.5 - Principal Manager◦ 2.6 - Group and Role Manager• 3. - Integration Guide<ul style="list-style-type: none">◦ 3.1 - Integration Overview◦ 3.2 - Client Configuration◦ 3.3 - Integration Libraries◦ 3.4 - SOAP API◦ 3.5 - .NET Client• 4. - Connectors<ul style="list-style-type: none">◦ 4.1 - Supported Applications and Directories◦ 4.2 - Integrating Crowd with Apache◦ 4.3 - Integrating Crowd with Bamboo◦ 4.4 - Integrating Crowd with Fisheye◦ 4.5 - Integrating Crowd with Confluence◦ 4.6 - Integrating Crowd with JIRA◦ 4.7 - Integrating Crowd with Jive Forums• 5. - External Resources	<h3>About</h3> <p>Crowd is a web-based single sign-on (SSO) tool that simplifies application provisioning and identity management.</p> <p>Crowd is the perfect solution to:</p> <ul style="list-style-type: none">• Give your users the convenience of single sign-on• Manage any number of users, logins and passwords• Centralise user management for applications such as JIRA, Confluence and Bamboo• Connect to multiple LDAP servers, such as Microsoft Active Directory• Integrate or import legacy user repositories• Control access to select applications for every user and group• Easily connect Crowd's application framework to new web applications <p>If you have a suggestion for improving the Crowd documentation here, please let us know.</p> <p>If you have a question about using Crowd, please feel free to contact us at support. You may also want to check the Crowd user forum.</p>

Table of Contents

[0. - Preface](#)

- [0.1 - Concepts](#)
- [0.2 - FAQ](#)

[1. - Installation Guide](#)

- [1.1 - Release Information](#)
 - [1.0](#)
 - [Beta Release 0.2](#)
 - [Beta Release 0.3](#)
 - [Beta Release 0.3.2](#)
 - [Beta Release 0.3.3](#)
 - [Beta Release 0.4](#)
 - [Beta Release 0.4.1](#)
 - [Beta Release 0.4.2](#)
 - [Beta Release 0.4.3](#)
 - [Beta Release 0.4.4](#)
 - [Beta Release 0.4.5](#)
 - [Release 1.0.0](#)
 - [Release 1.0.1](#)
- [1.2 - Requirements](#)
- [1.3 - Configuring Crowd](#)
- [1.4 - Database Configuration](#)
 - [1.3.1 - HSQL DB](#)
 - [1.3.2 - MS SQL Server](#)
 - [1.3.3 - MySQL](#)
 - [1.3.3 - PostgreSQL](#)
- [1.5 - Setup Wizard](#)
- [1.6 - Deployment FAQ](#)
 - [Self Signed Certificate](#)
- [1.7 - Upgrading Crowd](#)

[2. - Administration Console](#)

- [2.1 - About the Administration Console](#)
- [2.2 - Options and Settings](#)
 - [2.2.1 - Caching](#)
 - [2.2.2 - General Options](#)
 - [2.2.3 - Licensing](#)
 - [2.2.4 - Mail Server](#)
 - [2.2.5 - Mail Template](#)
 - [2.2.6 - Session](#)
- [2.3 - Directory Manager](#)
 - [2.3.1 - Directory Browser](#)
 - [2.3.2 - Adding a Directory Connector](#)
 - [2.3.3 - Internal Directory Connectors](#)
 - [2.3.4 - LDAP Directory Connectors](#)
 - [2.3.5 - Custom Directory Connectors](#)
- [2.4 - Application Manager](#)
 - [2.4.1 - Application Browser](#)
 - [2.4.2 - Adding an Application](#)
 - [2.4.3 - Managing an Integrated Application](#)
- [2.5 - Principal Manager](#)

- [2.5.1 - Principal Browser](#)
 - [2.5.2 - Adding a Principal](#)
 - [2.5.3 - Managing a Principal](#)
- [2.6 - Group and Role Manager](#)
 - [2.6.1 - Group and Role Browser](#)
 - [2.6.2 - Adding a Group or Role](#)
 - [2.6.3 - Managing a Group or Role](#)

[3. - Integration Guide](#)

- [3.1 - Integration Overview](#)
- [3.2 - Client Configuration](#)
- [3.3 - Integration Libraries](#)
- [3.4 - SOAP API](#)
- [3.5 - .NET Client](#)

[4. - Connectors](#)

- [4.1 - Supported Applications and Directories](#)
- [4.2 - Integrating Crowd with Apache](#)
- [4.3 - Integrating Crowd with Bamboo](#)
- [4.4 - Integrating Crowd with Fisheye](#)
- [4.5 - Integrating Crowd with Confluence](#)
- [4.6 - Integrating Crowd with JIRA](#)
- [4.7 - Integrating Crowd with Jive Forums](#)

[5. - External Resources](#)

0. - Preface

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

- [0.1 - Concepts](#)
- [0.2 - FAQ](#)

0.1 - Concepts

This page last changed on Mar 07, 2007 by rosie@atlassian.com.

Crowd is an application security framework that handles authentication and authorization for your web-based applications. With Crowd you can quickly integrate web applications into a single security architecture that supports single sign-on and centralized identity management. The application is divided into two parts:

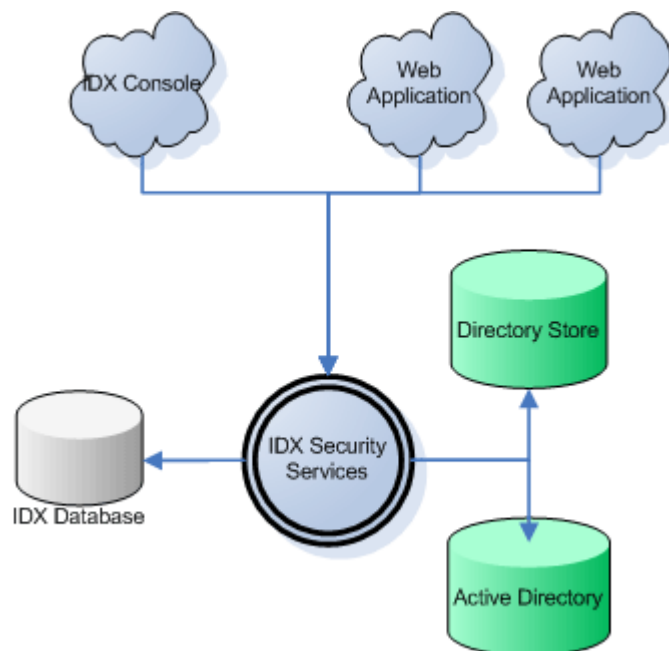
- The administration console is a clean and powerful web-interface to manage directories, users (known in Crowd as "principals") and their security rights.
- The integration API provides a platform neutral way to integrate web applications into a single security architecture. With the integration API, applications can quickly access user information or perform security checks.

Designed for ease of use, Crowd can be deployed with your existing infrastructure. Crowd supports Java, .NET and PHP. An unlimited number of directories can be configured. The directory servers can then be linked together providing applications with a single view to multiple directories.

Architectural Overview

The Crowd application is a middleware application that integrates web applications into a single security architecture that supports single sign-on and centralized identity management. The application works by dispatching authentication and authorization calls to configured directory servers. An unlimited number of directory servers may be configured and then linked to applications with an index order.

A typical deployment may be similar to the following:



When an application needs to authenticate or validate a security request, the application will make a simple API call to the Crowd framework, which will then map the call to the appropriate directory store.

Crowd supports popular directory servers such as Microsoft Active Directory, Apple Open Directory, Sun ONE and OpenLDAP. Custom directory connectors may be developed using the Crowd integration API.

0.2 - FAQ

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

- [About Crowd](#)
- [What is the difference between Authentication and Authorisation?](#)
- [What are Crowd's current integration points?](#)
- [Does the product include kerberos integration?](#)
- [Does Crowd support SAML or Liberty Alliance?](#)

Introduction

Crowd is an application security framework that handles authentication and authorisation for your web-applications. With Crowd you can quickly integrate web applications into a single security architecture that supports single sign-on and centralised identity management.

The application is divided into two parts:

- The administration console is a clean and powerful web-interface to manage directories, users and their security rights.
- The integration API provides a platform neutral way to integrate web applications into a single security architecture. With the integration API, applications can quickly access user information or perform security checks.

Designed for ease of use, Crowd can be deployed with your existing infrastructure. Crowd supports Java, .NET and PHP. An unlimited number of directories can be configured. The directory servers can then be linked together providing applications with a single view to multiple directories.

What is the difference between Authentication and Authorisation?

- Authentication: Is the act of establishing or confirming something (or someone) as authentic, that is, that claims made by or about the thing are true. Generally this is in the form of username and a password.
- Authorisation: The authorisation process is used to decide if person, program or device X is allowed to have access to data, functionality or service Y. This often comes in the form of groups, roles and permissions.

What are Crowd's current integration points?

Built In Connectors

Directory Servers

- Microsoft Active Directory
- OS X Open Directory

- SunONE
- OpenLDAP

Software

- Confluence
- JIRA
- JIVE
- Tomcat
- Webwork 1/2

Keep an eye out — more connectors to come!

Does the product include kerberos integration?

No, but we plan to add support for [kerberos](#)-based authentication for clients to authenticate versus the security framework. The current roadmap does not have this specifically stubbed out but over the next few weeks we'll be hammering out something more specific.

Currently the Crowd framework supports a generic Credentials object that can be adapted to support any number of authentication approaches such as three-factor authentication.

Does Crowd support SAML or Liberty Alliance?

[SAML](#) is a standard that was developed by several large companies for federated identity management. Similarly, [Liberty Alliance](#) is a consortium formed to develop and define federated identity management standards and protocols.

In our opinion, for the 98% of businesses who wish to enforce single sign-on, SAML specification is too complex to be truly practical. The breadth of understanding, deployment and support of these large frameworks is beyond the scope of most developers' needs or their ability to manage. Most developers and IT managers need a solution that is simple and cost effective to deploy. Crowd was developed as a practical, simple and secure alternative for identity management and single-sign on across an unlimited number of web-based applications.

1. - Installation Guide

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

- [1.1 - Release Information](#)
 - [1.0](#)
 - [Beta Release 0.2](#)
 - [Beta Release 0.3](#)
 - [Beta Release 0.3.2](#)
 - [Beta Release 0.3.3](#)
 - [Beta Release 0.4](#)
 - [Beta Release 0.4.1](#)
 - [Beta Release 0.4.2](#)
 - [Beta Release 0.4.3](#)
 - [Beta Release 0.4.4](#)
 - [Beta Release 0.4.5](#)
 - [Release 1.0.0](#)
 - [Release 1.0.1](#)
- [1.2 - Requirements](#)
- [1.3 - Configuring Crowd](#)
- [1.4 - Database Configuration](#)
 - [1.3.1 - HSQL DB](#)
 - [1.3.2 - MS SQL Server](#)
 - [1.3.3 - MySQL](#)
 - [1.3.3 - PostgreSQL](#)
- [1.5 - Setup Wizard](#)
- [1.6 - Deployment FAQ](#)
 - [Self Signed Certificate](#)
- [1.7 - Upgrading Crowd](#)

1.1 - Release Information

This page last changed on Mar 04, 2007 by justen.stepka@atlassian.com.

The latest Crowd release is version 1.0.0.

Installation

Information for installing Crowd can be found [here](#).

<http://confluence.atlassian.com/display/CROWD/Documentation>

Handy Links

- [Documentation Home](#)
- [JIRA Issue Tracker for Crowd](#)
- [Javadoc Home](#)
- Forums and Mailing List
 - [Crowd Announcements](#)
 - [Crowd General Forum](#)
 - [Crowd Developers Forum](#)

Release Notes

- [Release 1.0.0](#)
- [Beta Release 0.4.5](#)
- [Beta Release 0.4.4](#)
- [Beta Release 0.4.3](#)
- [Beta Release 0.4.2](#)
- [Beta Release 0.4.1](#)
- [Beta Release 0.4](#)
- [Beta Release 0.3.3](#)
- [Beta Release 0.3.2](#)
- [Beta Release 0.3](#)
- [Beta Release 0.2](#)

1.0

This page last changed on Jan 03, 2007 by [justin](#).

- All LDAP configuration now need to have filters set
- If you are using PostgreSQL you need to change the column name `attributevalue.attributevalueid` to `attributevalue.ATTRIBUTEVALUEID` (make it uppercase).

Beta Release 0.2

This page last changed on Dec 10, 2006 by justen.stepka@atlassian.com.

Crowd 0.2

- [Standalone version - Tomcat 5.5 with HSQL - .zip](#) (59.5Mbs)
- [Standalone version - Tomcat 5.5 with HSQL - .tar.gz](#) (59.7Mbs)

Points of Interest

- There is an error when unzipping on the Windows platform, the archive integrity is fine and this will be fixed for the 0.3 release.
- The focus of this distribution is for JIRA and Confluence integration. Performance enhancements will be added for the 0.3 release which will allow large user-databases to be integrated.

Beta Release 0.3

This page last changed on Dec 10, 2006 by justen.stepka@atlassian.com.

Crowd 0.3

- [Standalone version - Tomcat 5.5 with HSQL - .zip](#) (65.3 Mbs)
- [Standalone version - Tomcat 5.5 with HSQL - .tar.gz](#) (64.7 Mbs)

Points of Interest

- The focus of this distribution is on performance for a large number of users and groups when integrating JIRA, Confluence and Bamboo integration.

Beta Release 0.3.2

This page last changed on Dec 14, 2006 by justen.stepka@atlassian.com.

The Crowd development team has released a new version of Crowd - 0.3.2.

This release addresses a Seraph SSO issue when integrating JIRA, Confluence and Bamboo.

<http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12540>

You can now download Crowd from <http://www.atlassian.com/Crowd>

Cheers,

The Atlassian Crowd Development Team

Beta Release 0.3.3

This page last changed on Jan 24, 2007 by [ssmith](#).

The Crowd development team has released a new version of Crowd - 0.3.3.

This release addresses the following:

- Upgrade from Webwork 1 to Webwork 2
- Workaround for Active Directory to support CN forwards.

CRITICAL POSTGRES UPGRADE NOTES: <http://jira.atlassian.com/browse/CWD-71>

We started testing on IE7 and have noticed the CSS bugs and will work to get this addressed for the next build.

<http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12544>

You can now download Crowd from <http://www.atlassian.com/Crowd>

Cheers,

The Atlassian Crowd Development Team

Beta Release 0.4

This page last changed on Jan 24, 2007 by justen.stepka@atlassian.com.

The Crowd development team has released a new version of Crowd - 0.4.

This release addresses several critical issues:

- Seraph Logout code fails to logout the user in Confluence, Bamboo and JIRA.
- Unable to search for a Principal by email address.
- Accept header authentication factor unreliable with Mozilla based browsers.
- Default 'localhost' configuration not added valid IP address of 127.0.0.1.

New features include:

- Allow all to authenticate.
- New LDAP connectors build off Spring LDAP Template with better performance enhancements.
- Support for LDAP filters

All Postgres DB will need to have the following command ran:

```
alter table "APPLICATIONDIRECTORIES" add column "ALLOWALLTOAUTHENTICATE" boolean;
```

<http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12266>

You can now download Crowd from <http://www.atlassian.com/Crowd>

Cheers,

The Atlassian Crowd Development Team

Beta Release 0.4.1

This page last changed on Jan 16, 2007 by justen.stepka@atlassian.com.

The Crowd development team has released a new version of Crowd - 0.4.1.

This addresses bugs which can be viewed through our JIRA issue tracker:

<http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12600>

You can now download Crowd from <http://www.atlassian.com/Crowd>

Cheers,

The Atlassian Crowd Development Team

Beta Release 0.4.2

This page last changed on Jan 24, 2007 by justen.stepka@atlassian.com.

The Crowd development team has released a new version of Crowd - 0.4.2.

This addresses bugs which can be viewed through our JIRA issue tracker:

<http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12623>

You can now download Crowd from <http://www.atlassian.com/Crowd>

Cheers,

The Atlassian Crowd Development Team

Beta Release 0.4.3

This page last changed on Jan 31, 2007 by justen.stepka@atlassian.com.

The Crowd development team has released a new version of Crowd - 0.4.3.

This addresses bugs which can be viewed through our JIRA issue tracker:

<http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12267>

- Support for AD when there are more than 999 records in a search result.
- Reduced the number of necessary libs for a client application.
- Improved the 'build.properties' file configuration.

You can now download Crowd from <http://www.atlassian.com/Crowd>

Cheers,

The Atlassian Crowd Development Team

Beta Release 0.4.4

This page last changed on Feb 06, 2007 by justen.stepka@atlassian.com.

The Crowd development team has released a new version of Crowd - 0.4.4.

This addresses bugs which can be viewed through our JIRA issue tracker:

<http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12642>

- Caching improvement for Confluence.
- Removed an additional attribute that was causing integration problems with SOAP services when using Active Directory.

You can now download Crowd from <http://www.atlassian.com/Crowd>

Cheers,

The Atlassian Crowd Development Team

Beta Release 0.4.5

This page last changed on Feb 19, 2007 by justen.stepka@atlassian.com.

The Crowd development team has released a new version of Crowd - 0.4.5.

This addresses bugs which can be viewed through our JIRA issue tracker:

<http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12652>

- Improved Active Directory LDAP attribute filtering.
- UI improvements with new screen layouts.
- Spring TX management.

You can now download Crowd from <http://www.atlassian.com/Crowd>

Cheers,

The Atlassian Crowd Development Team

Release 1.0.0





















This page last changed on Mar 04, 2007 by justen.stepka@atlassian.com.

The Crowd development team has released Crowd 1.0.

This addresses bugs which can be viewed through our JIRA issue tracker:

- UI improvements with new screen layouts.
- Import and Export process for XML.
- LDAP Fixes for OpenLDAP and Microsoft Active Directory.
- Improved error reporting.
- Apache / Subversion support.

You can now download Crowd from <http://www.atlassian.com/Crowd>

Atlassian JIRA (10 issues)			
Key	Summary	Pr	Status
CWD-173	Implement an import and export function in Crowd		 Resolved
CWD-188	License update (when invalid) page should detail current license details.		 Resolved
CWD-184	Make Crowd's internal exception extend NestableException from commons-lang		 Resolved
CWD-180	Schema violation with LDAP and Groups/Roles		 Resolved
CWD-178	LDAP flags are incorrect for Active Directory/LDAP (Win2k3 domain)		 Resolved
CWD-150	Build fails		 Resolved
CWD-101	Unable to upgrade from 0.2 to 0.3.3		 Closed
CWD-97	Apache mod Crowd integration		 Resolved
CWD-90	sso support for fisheye		 Resolved
CWD-62	500 page.		 Resolved

Cheers,

The Atlassian Crowd Development Team

Release 1.0.1







This page last changed on Mar 05, 2007 by justen.stepka@atlassian.com.

The Crowd development team has released Crowd 1.0.1.

This addresses 3 critical bugs which can be viewed through our JIRA issue tracker:

- Create new group/role broken using OpenLDAP.
- XFireFault exception: "No write method for property".
- Single sign on Seraph authentication fails when the host on a domain is not the same.

You can now download Crowd from <http://www.atlassian.com/Crowd>

Atlassian JIRA (3 issues)			
Key	Summary	Pr	Status
CWD-190	XFireFault exception: "No write method for property".		 Resolved
CWD-189	Create new group/role broken using OpenLDAP		 Resolved
CWD-82	Single sign on Seraph authentication fails when the host on a domain is not the same.		 Resolved

Cheers,

The Atlassian Crowd Development Team

1.2 - Requirements

This page last changed on Feb 28, 2007 by [shamid](#).

Crowd Standalone ships with a built-in database, however, for production environments we recommend configuring Crowd Standalone to use an external database. Crowd supports most relational database servers and therefore we suggest using the one you are most comfortable administering. If you are looking for a low cost solution, please consider using [MySQL](#) or [PostgreSQL](#) as both of these are open source (free) software.

Software Requirements

- J2EE 1.4 application server or a Servlet 2.3 web container.
- JDBC compliant database that is supported by Hibernate.
- Sun JDK 1.4 (1.5 or higher is preferred).

Hardware Requirements

The hardware required to run Crowd heavily depends on the number of applications and users that your installation will have, as well as the maximum number of concurrent requests that the system will experience during peak hours.

During evaluation Crowd will run well on any reasonably fast workstation computer (eg. 1.5+Ghz processor). Memory requirements depend on how many projects and issues you will store, but 256MB is enough for most evaluation purposes.

Most users start by downloading Crowd Standalone, and running it on their local computer. It is easy to move Crowd to a central server later.

We would appreciate if you let us know what hardware configuration works for you. Please create a support request in [JIRA](#) with your hardware specification and mention the number of users and issues in your Crowd installation.

Supported Databases

Crowd requires a database for storage of data, the following database servers are supported:

- DB2
- Firebird
- Frontbase
- HypersonicSQL
- Informix
- Ingres
- Interbase
- Pointbase
- PostgreSQL
- Mckoi SQL

- Microsoft SQL Server
- MySQL
- Oracle
- Pointbase
- SAP DB
- Sybase

Supported J2EE servers

The following J2EE servers are supported:

- Borland
- JBoss
- JOTM
- JOnAS
- JRun
- Orion
- Resin
- Tomcat
- Weblogic
- WebSphere

1.3 - Configuring Crowd

This page last changed on Mar 06, 2007 by dhardiker@adaptavist.com.

When first configuring the Crowd application the `build.xml` and `build.properties` file are of primary concern. The `build.xml` is an ant script that loads properties from the `build.properties` configuration file. This script can be used to quickly adjust and configure various deployment properties of Crowd and the demo application.

```
jstepka-osx:~/atlassian-crowd-0.1-SNAPSHOT $ ls -l
total 48
drwxr-xr-x  13 jstepka  jstepka   442 Nov 20 13:57 apache-tomcat-5.5.20
-rw-r--r--   1 jstepka  jstepka   259 Nov 17 14:16 build.bat
-rw-r--r--   1 jstepka  jstepka   559 Nov 17 14:16 build.properties
-rwxr-xr-x   1 jstepka  jstepka   565 Nov 17 14:16 build.sh
-rw-r--r--   1 jstepka  jstepka  1961 Nov 17 14:16 build.xml
drwxr-xr-x   7 jstepka  jstepka   238 Nov 21 12:50 crowd-webapp
drwxr-xr-x   4 jstepka  jstepka   136 Nov 21 12:52 database
drwxr-xr-x   9 jstepka  jstepka   306 Nov 21 12:50 demo-src
drwxr-xr-x  12 jstepka  jstepka   408 Nov 21 12:50 demo-webapp
drwxr-xr-x   5 jstepka  jstepka   170 Nov 21 12:50 etc
-rw-r--r--   1 jstepka  jstepka   274 Nov 17 14:16 start_crowd.bat
-rwxr-xr-x   1 jstepka  jstepka   183 Nov 20 13:34 start_crowd.sh
```

The default `build.properties` will look similar to the following:

```
# Modify the attributes of this file to quickly adjust the deployment values of Crowd.

# The Hibernate database dialect to use. See .\etc\hibernate.properties for a list of supported
dialects.
hibernate.dialect=org.hibernate.dialect.HSQLDialect

# The Hibernate transaction factory to use. See .\etc\hibernate.properties for a list of
supported dialects.
hibernate.transaction.factory_class=org.hibernate.transaction.JTATransactionFactory

# Crowd context root
crowd.url=http://localhost:8080/crowd

# Demo context root
crowd.url=http://localhost:8080/demo
```

Build Properties (`build.properties`)

Parameter	Description
<code>hibernate.dialect</code>	This parameter controls the database dialect the Hibernate persistence system will use when executing commands versus your database server.
<code>hibernate.transaction.factory_class</code>	<p>This parameter controls the transaction factory to use when executing transactions at run-time: Hibernate provides two generic options, additional application server specific options are available:</p> <ul style="list-style-type: none"><code>org.hibernate.transaction.JDBCTransactionFactory</code> delegates to database (JDBC) transactions (default).<code>org.hibernate.transaction.JTATransactionFactory</code> delegates to JTA (if an existing transaction is under way, the work performed is done in

	that context. Otherwise a new transaction is started).
crowd.url	The path and port for the root of the Crowd web-application.
demo.url	The path and port for the door of the Crowd demo web-application

Hibernate SQL Dialects (`hibernate.dialect`)

RDBMS	Dialect
DB2	<code>org.hibernate.dialect.DB2Dialect</code>
DB2 AS/400	<code>org.hibernate.dialect.DB2400Dialect</code>
DB2 OS390	<code>org.hibernate.dialect.DB2390Dialect</code>
Firebird	<code>org.hibernate.dialect.FirebirdDialect</code>
FrontBase	<code>org.hibernate.dialect.FrontbaseDialect</code>
HypersonicSQL	<code>org.hibernate.dialect.HSQLDialect</code>
Informix	<code>org.hibernate.dialect.InformixDialect</code>
Ingres	<code>org.hibernate.dialect.IngresDialect</code>
Interbase	<code>org.hibernate.dialect.InterbaseDialect</code>
Mckoi SQL	<code>org.hibernate.dialect.MckoiDialect</code>
Microsoft SQL Server	<code>org.hibernate.dialect.SQLServerDialect</code>
MySQL	<code>org.hibernate.dialect.MySQLDialect</code>
MySQL with InnoDB	<code>org.hibernate.dialect.MySQLInnoDBDialect</code>
MySQL with MyISAM	<code>org.hibernate.dialect.MySQLMyISAMDDialect</code>
Oracle (any version)	<code>org.hibernate.dialect.OracleDialect</code>
Oracle 9i/10g	<code>org.hibernate.dialect.Oracle9Dialect</code>
Pointbase	<code>org.hibernate.dialect.PointbaseDialect</code>
PostgreSQL	<code>org.hibernate.dialect.PostgreSQLDialect</code>
Progress	<code>org.hibernate.dialect.PosgressDialect</code>
SAP DB	<code>org.hibernate.dialect.SAPDBDialect</code>
Sybase	<code>org.hibernate.dialect.SybaseDialect</code>
Sybase Anywhere	<code>org.hibernate.dialect.SybaseAnywhereDialect</code>

Hiberante Transaction Factories (`hibernate.transaction.factory_class`)

J2EE Server	Dialect
Borland ES	<code>org.hibernate.transaction.BESTransactionManagerLookup</code>

JBoss	org.hibernate.transaction.JBossTransactionManagerLookup
JOnAS	org.hibernate.transaction.JOnASTransactionManagerLookup
JOTM	org.hibernate.transaction.JOTMTransactionManagerLookup
JRun4	org.hibernate.transaction.JRun4TransactionManagerLookup
Orion	org.hibernate.transaction.OrionTransactionManagerLookup
Resin	org.hibernate.transaction.ResinTransactionManagerLookup
Weblogic	org.hibernate.transaction.WeblogicTransactionManagerLookup
WebSphere	org.hibernate.transaction.WebSphereTransactionManagerLookup

If configuring Crowd and/or the demo application to run on a port and context path other than the default, you will then need to run the command `./build.sh` verses the `build.xml` configuration file. This process will then edit all of the necessary Crowd configuration files for your deployment.

```
jstepka-osx:~/atlassian-crowd-0.1-SNAPSHOT $ ./build.sh
Buildfile: build.xml

init:

assistant:
    [echo] Configuring the Crowd Console
    [echo] Copying crowd.properties to: crowd-webapp/WEB-INF/classes
    [copy] Copying 1 file to
/Users/jstepka/Projects/crowd/releases/atlassian-crowd-0.1-SNAPSHOT/crowd-webapp/WEB-INF/classes
    [echo] Configuring the Crowd hibernate configuration
    [echo] Copying hibernate.properties to: crowd-webapp/WEB-INF/classes
    [copy] Copying 1 file to
/Users/jstepka/Projects/crowd/releases/atlassian-crowd-0.1-SNAPSHOT/crowd-webapp/WEB-INF/classes
    [echo] Configuring the demo application
    [echo] Renaming and copying demo.properties to:
demo-webapp/WEB-INF/classes/crowd.properties
    [copy] Copying 1 file to
/Users/jstepka/Projects/crowd/releases/atlassian-crowd-0.1-SNAPSHOT/demo-webapp/WEB-INF/classes

BUILD SUCCESSFUL
Total time: 0 seconds
```

With the configuration files correctly configured, you may now run the application server.

Tomcat Standalone

The standalone Tomcat distribution uses an in-memory HSQL database engine. This JNDI reference (CrowdDS can be adjusted by editing the `crowd.xml` deployment description to use your custom database and driver. To start Crowd, perform the following:

- Unzip the download archive (avoid directories with spaces).
- Run the start-up script: `start_crowd.bat` for Windows or `start_crowd.sh` in a Unix environment.
- Point a web browser at <http://localhost:8095/> where you will see the introduction page for configuring the Crowd security server.

Custom Containers

Configuring the Crowd security server to run with your existing application server is dependent on the

deployment process of the chosen server. The web-application is located in the `crowd` directory of the downloaded archive. Regardless of the application deployment process, a JNDI data-source will need to be configured. The name of the JNDI data-source must be `jdbc/CrowdDS`.

1.4 - Database Configuration

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

By default Crowd standalone is shipped with a [HSQL](#) preconfigured. This is fine for evaluation purposes, but for production installations, you should connect Crowd to an enterprise database. This also lets you take advantage of existing database backup and recovery procedures.

Below is a list of supported databases and their Hibernate configurations.

Hibernate SQL Dialects (`hibernate.dialect`)

RDBMS	Dialect
HypersonicSQL	<code>org.hibernate.dialect.HSQLDialect</code>
Microsoft SQL Server	<code>org.hibernate.dialect.SQLServerDialect</code>
MySQL	<code>org.hibernate.dialect.MySQLDialect</code>
MySQL with InnoDB	<code>org.hibernate.dialect.MySQLInnoDBDialect</code>
MySQL with MyISAM	<code>org.hibernate.dialect.MySQLMyISAMDDialect</code>
PostgreSQL	<code>org.hibernate.dialect.PostgreSQLDialect</code>

The following instructions will allow you to configure Crowd to an external database:

- [1.3.1 - HSQL DB](#)
- [1.3.2 - MS SQL Server](#)
- [1.3.3 - MySQL](#)
- [1.3.3 - PostgreSQL](#)

1.3.1 - HSQL DB

This page last changed on Feb 21, 2007 by justen.stepka@atlassian.com.

The default version of Crowd uses an embedded HSQL DB

Also see <http://hsqldb.sourceforge.net/doc/guide/ch01.html#N101C2> .

HSQL DB periodically must update its files to represent changes made in the database. In doing so, it must delete the current `crowddb.data` file on the filesystem (beneath `/database` folder) and replace it with a new one.

If an administrator issues a shutdown on Crowd in this period, data can be lost, and is typically noticed by all configuration data for your Crowd server being lost.



HSQLDB should not be used as a production database. It is included for evaluation purposes only.

1.3.2 - MS SQL Server

This page last changed on Feb 28, 2007 by [justin](#).

1. Configure SQL Server

1. Create a database user which Crowd will connect as (e.g. crowduser).



In SQL Server, the database user (crowduser above) should not be the database owner, but should be in the `db_owner` role.

2. Create a database for Crowd to store data in (e.g. crowddb).
3. Ensure that the user has permission to connect to the database, and create and populate tables

2. Copy the SQL Server driver to your application server

1. Download the SQL Server JDBC driver from [JTDS](#) (recommended, assumed below), or [I-net software](#) (commercial).



Microsoft have their own JDBC driver but we strongly recommend avoiding it after our JIRA customers have reported various connection errors ([JIRA-5760](#), [JIRA-6872](#) | <http://jira.atlassian.com/browse/JIRA-6872>), workflow problems ([JIRA-8443](#)) and Chinese character problems ([JIRA-5054](#)).

2. Add the SQL Server JDBC driver jar (jtds-[version].jar) to the `common/lib` directory.

3. Configure your application server to connect to SQL Server

1. Edit the `conf/Catalina/localhost/crowd.xml` and customise the username, password, `driverClassName` and url parameters for the Datasource.

```
<Context path="/crowd" docBase="../../crowd-webapp" debug="0">

  <Resource name="jdbc/CrowdDS" auth="Container" type="javax.sql.DataSource"
    username="[enter db username here]"
    password="[enter db password here]"
    driverClassName="net.sourceforge.jtds.jdbc.Driver"
    url="jdbc:jtds:sqlserver://localhost:1433/crowddb"
    [ delete the minEvictableIdleTimeMillis, timeBetweenEvictionRunsMillis and
maxActive params here ]
  />

  <Manager className="org.apache.catalina.session.PersistentManager" saveOnRestart="false"/>
```

```
</Context>
```

2. Delete the `minEvictableIdleTimeMillis`, `timeBetweenEvictionRunsMillis` and `maxActive` attributes (which are only needed for HSQL, and degrade performance otherwise).

4. Configure Crowd to use MS SQL Server

1. Edit the `build.properties` file located in the root of the standalone release and modify the `hibernate.dialect` to the following

```
hibernate.dialect=org.hibernate.dialect.SQLServerDialect
```

2. Then run the `./build.sh` or `build.bat`, this will configure crowd to use the MS SQL Server dialect.

If you do not wish to edit this file and run the build script, you can edit the `jdbc.properties` (which the above script modifies) directly. The `jdbc.properties` file is located here:

`crowd-webapp\WEB-INF\classes\jdbc.properties`, modify the file to the following:

```
# - Crowd Configuration Options

hibernate.connection.datasource=java\:comp/env/jdbc/CrowdDS
hibernate.dialect=org.hibernate.dialect.SQLServerDialect
hibernate.transaction.factory_class=org.hibernate.transaction.JDBCTransactionFactory

...
```

Next steps

You should now have an application server configured to connect to a database, and Crowd configured to use the correct database. Now start up Crowd and watch the logs for any errors.

1.3.3 - MySQL

This page last changed on Feb 28, 2007 by [justin](#).

1. Configure MySQL

1. Create a database user which Crowd will connect as (e.g. crowduser).
2. Create a database for Crowd to store data in (e.g. crowddb).
3. Ensure that the user has permission to connect to the database, and create and populate tables

2. Copy the MySQL driver to your application server

1. Download the latest [MySQL Connector/J JDBC driver](#).
2. Add the MySQL JDBC driver jar (mysql-connector-java-3.x.x-bin.jar) to the `common/lib/` directory.
NOTE: Do not place the Debug Driver (mysql-connector-java-3.x.x-bin-g.jar) on the `CLASSPATH` as this can cause issues ([JRA-8674](#)).

3. Configure your application server to connect to MySQL

1. Edit the `conf/Catalina/localhost/crowd.xml` and customise the username, password, `driverClassName` and `url` parameters for the Datasource.

```
<Context path="/crowd" docBase="../../crowd-webapp" debug="0">

  <Resource name="jdbc/CrowdDS" auth="Container" type="javax.sql.DataSource"
    username="[enter db username here]"
    password="[enter db password here]"
    driverClassName="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost/crowddb?autoReconnect=true&useUnicode=true&characterEncoding=utf8;delete the minEvictableIdleTimeMillis, timeBetweenEvictionRunsMillis and
maxActive params here ]
    />

  <Manager className="org.apache.catalina.session.PersistentManager" saveOnRestart="false"/>
</Context>
```

The URL above assumes a UTF-8 database - ie. created with `create database crowddb character set utf8;`. If you don't specify character set `utf8` you risk getting 'Data truncation: Data too long for column' errors.



MySQL closes idle connection after 8 hours, so the `autoReconnect=true` is necessary to tell the driver to reconnect

2. Delete the `minEvictableIdleTimeMillis`, `timeBetweenEvictionRunsMillis` and `maxActive` attributes (which are only needed for HSQL, and degrade performance otherwise).

4. Configure Crowd to use MySQL

1. Edit the build.properties file located in the root of the standalone release and modify the hibernate.dialect to the following, please only choose one of the 3 available options depending on how you have configured your database server.

```
*For MySQL set:*
hibernate.dialect=org.hibernate.dialect.MySQLDialect
*For MySQL with InnoDB set:*
hibernate.dialect=org.hibernate.dialect.MySQLInnoDBDialect
*For MySQL with MyISAM set:*
hibernate.dialect=org.hibernate.dialect.MySQLMyISAMDDialect
```

2. Then run the ./build.sh or build.bat, this will configure crowd to use the MySQL dialect.

If you do not wish to edit this file and run the build script, you can edit the jdbc.properties (which the above script modifies) directly. The jdbc.properties file is located here:

crowd-webapp\WEB-INF\classes\jdbc.properties, modify the file to the following:

```
# - Crowd Configuration Options

hibernate.connection.datasource=java\:comp/env/jdbc/CrowdDS
hibernate.dialect=org.hibernate.dialect.MySQLDialect
hibernate.transaction.factory_class=org.hibernate.transaction.JDBCTransactionFactory

...
```

Next steps

You should now have an application server configured to connect to a database, and Crowd configured to use the correct database. Now start up Crowd and watch the logs for any errors.

1.3.3 - PostgreSQL

This page last changed on Feb 28, 2007 by [justin](#).

1. Configure PostgreSQL

1. Create a database user which Crowd will connect as (e.g. crowduser).
2. Create a database for Crowd to store data in (e.g. crowddb).
3. Ensure that the user has permission to connect to the database, and create and populate tables

2. Copy the PostgreSQL driver to your application server

1. Download the PostgreSQL JDBC driver from <http://jdbc.postgresql.org/download.html>. Get the JDBC 3 driver specific to your Postgres version, eg. + postgresql-8.x-xxx.jdbc3.jar.
2. Add the PostgreSQL JDBC driver jar to the common/lib directory.

3. Configure your application server to connect to PostgreSQL

1. Edit the conf/Catalina/localhost/crowd.xml and customise the username, password, driverClassName and url parameters for the Datasource.

```
<Context path="/crowd" docBase="../../crowd-webapp" debug="0">

    <Resource name="jdbc/CrowdDS" auth="Container" type="javax.sql.DataSource"
        username="[enter db username here]"
        password="[enter db password here]"
        driverClassName="org.postgresql.Driver"
        url="jdbc:postgresql://host:port/database" [ see also
http://jdbc.postgresql.org/doc.html ]"
        [ delete the minEvictableIdleTimeMillis, timeBetweenEvictionRunsMillis and
maxActive params here ]
    />

    <Manager className="org.apache.catalina.session.PersistentManager" saveOnRestart="false"/>

</Context>
```

2. Delete the minEvictableIdleTimeMillis, timeBetweenEvictionRunsMillis and maxActive attributes (which are only needed for HSQL, and degrade performance otherwise).

4. Configure Crowd to use PostgreSQL

1. Edit the build.properties file located in the root of the standalone release and modify the hibernate.dialect to the following

```
hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
```

2. Then run the `./build.sh` or `build.bat`, this will configure crowd to use the PostgreSQL dialect.

If you do not wish to edit this file and run the build script, you can edit the `jdbc.properties` (which the above script modifies) directly. The `jdbc.properties` file is located here:

`crowd-webapp\WEB-INF\classes\jdbc.properties`, modify the file to the following:

```
# - Crowd Configuration Options

hibernate.connection.datasource=java\:comp/env/jdbc/CrowdDS
hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
hibernate.transaction.factory_class=org.hibernate.transaction.JDBCTransactionFactory

...
```

Next steps

You should now have an application server configured to connect to a database, and Crowd configured to use the correct database. Now start up Crowd and watch the logs for any errors.

1.5 - Setup Wizard

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

Welcome to the Setup Wizard




Useful Information

Access the URL <http://localhost:8095/console> or <http://localhost:8095/crowd/console> to bring the setup wizard.

When accessing the Crowd administration console for the first time, the server will require you to configure the application with a set of default values. All of these values can be adjusted after completing the installation.

Licensing

Crowd licensing is limited by the number of applications that are able to communicate with the security server. Evaluation licenses may be obtained from the [Atlassian](#) website.

[Help](#)

Home

License

It appears this is the first time that you are running Crowd, this setup wizard will take you through your initial configuration:

Server ID: AQCX-WX9J-AG0H-R0TZ

License:

An evaluation license key is available from the [Atlassian website](#).

Continue »

Powered by [Atlassian Crowd](#) Version @BUILD@ (@BUILDDATE@)[Report a bug](#) | [Request a feature](#) | [Contact Atlassian](#)

Options

This part of the setup process controls the general options of the Crowd security server.

Options

Deployment Title:	*	<input type="text"/>	The deployment title is the name of this instance.
Domain:	*	<input type="text"/>	The domain for this deployment, i.e. acmecorp.com.
Session Timeout:	*	<input type="text" value="5"/>	How long a session will be valid for before expiring. This is in minutes and must be greater than 0.

Continue »

- **Deployment Title:** - This attribute specifies a unique name for the deployment that is sometimes used when sending email notifications.
- **Domain:** - Used when setting authentication information in a browser. If this attribute is not the correct domain single sign on will not work when switching between applications.
- **Token Seed:** - This attribute is a unique deployment key that is used when generating tokens for an authenticated client.

Mail Server

The security server will send notices to users during special events such as when a password is reset. Enter in the details of your mail server with a username and password if required.


Mail Server

Notification Email:	*	<input type="text"/>	Email address to send server messages to when server notifications occur.
SMTP Host:	*	<input type="text"/>	The host address, i.e. localhost or smtp.acmecorp.com.
From:	*	<input type="text"/>	The sender (or FROM) address to use when sending email notifications.
Subject Prefix:		<input type="text" value="[Atlassian - Atlassian Crowd]"/>	The subject prefix to use when sending email notifications. This is useful for mail client filtering rules, i.e. [ACME CORP - Crowd].
Username:		<input type="text" value="jstepka"/>	The username to use when connecting to the mail server.
Password:		<input type="password" value="*****"/>	The password to use when connecting to the mail server.

Continue »

Directory Server

To access the administration console, a default directory needs to be configured. A default administrator or administration group will be assigned to access the web-console. Specific information about configuring directory servers is discussed in the [2.3 - Directory Manager](#) section.

 [Help](#)

Home

Internal Directory

Name:	<input type="text" value="Atlassian"/> <small>The name of the directory is to categorize the directory instance. This is useful when there are multiple directories configured, i.e. Chicago Employees or Web Customers.</small>
Description:	<input type="text"/> <small>Details about this specific directory.</small>
Password Regex:	<input type="text"/> <small>Regex pattern which new passwords will be validated against. Leave blank to disable this feature.</small>
Maximum Invalid Password Attempts:	<input type="text" value="0"/> <small>The maximum number of invalid password attempts before the authenticating account will be disabled. Enter 0 to disable this feature.</small>
Maximum Unchanged Password Days:	<input type="text" value="0"/> <small>The number of days until the password must be changed. This value is in days, enter 0 to disable this feature.</small>
Password History Count:	<input type="text" value="0"/> <small>The number of previous passwords to prevent the principal from using. Enter 0 to disable this feature.</small>

Continue »

Powered by [Atlassian Crowd](#) Version @BUILD@ (@BUILDDATE@) [Report a bug](#) | [Request a feature](#) | [Contact Atlassian](#)

Internal directories use the Crowd database to store user, group and role information. The internal directory information is stored locally to the configured database server.

Default Administrator

Default Administrator

To configure the security server, a default administrator needs to be created. Additional administrators may be added later.

Email:	*	<input type="text" value="justen.stepka@atlassian.com"/>	<small>Email addresses must follow the RFC2822 format.</small>
Username:	*	<input type="text" value="stepka"/>	<small>Enter administrator user name.</small>
Password:	*	<input type="password" value="*****"/>	
Confirm Password:	*	<input type="password" value="*****"/>	
First Name:	*	<input type="text" value="Justen"/>	
Last Name:	*	<input type="text" value="Stepka"/>	

Continue »

Default administrator and group information will be created, granting the installer access to the administration console. An unlimited number of internal directories may be configured from the administration console.

Demo Application

An option to auto-configure the demo application server is available during the setup to assist you with quickly setting up and configuring the identity server. The demo application highlights best practices when using the Crowd framework. The Crowd download archive contains the entire source for the demo application, which can be used as an example when integrating your web applications.

Configure Demo Application

Would you like to have the demo application auto configured for you?

The demo application highlights best practices when using the Crowd framework. The Crowd download archive contains the entire source to the demo application, which can be used as an example when integrating your web-applications.

Yes »

No

Setup Complete

You are now ready to use the Crowd security server.

1.6 - Deployment FAQ

This page last changed on Feb 21, 2007 by justen.stepka@atlassian.com.

[Self Signed Certificate](#)

Self Signed Certificate

This page last changed on Nov 30, 2006 by justen.stepka@atlassian.com.

I have a self Signed Certificate

You will need to add the self-signed certificate to your JDK truststore using the JDK keytool:
<http://java.sun.com/j2se/1.3/docs/tooldocs/win32/keytool.html>

1.7 - Upgrading Crowd

This page last changed on Mar 06, 2007 by justen.stepka@atlassian.com.

Upgrading Crowd

Database Configuration

You will need to check of the `crowd.xml` in the `conf/catalina/localhost/` directory is hooked up to a database. If this is you will need to modify the `crowd.xml` from the new archive to correspond to the previous one.

Do not forget to copy over any libs necessary for your JDBC connection. If you are using the in memory database (HSQL), you will need to copy over the contents of the `database` folder to the new archive download folder.

The Crowd Administration Console

Copy the old Crowd `crowd.properties` from the `WEB-INF/classes` folder to the new `WEB-INF` classes folder. The `application.password` in this file is critical. If you do not do this the Crowd console will not work.

2. - Administration Console

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

- [2.1 - About the Administration Console](#)
- [2.2 - Options and Settings](#)
 - [2.2.1 - Caching](#)
 - [2.2.2 - General Options](#)
 - [2.2.3 - Licensing](#)
 - [2.2.4 - Mail Server](#)
 - [2.2.5 - Mail Template](#)
 - [2.2.6 - Session](#)
- [2.3 - Directory Manager](#)
 - [2.3.1 - Directory Browser](#)
 - [2.3.2 - Adding a Directory Connector](#)
 - [2.3.3 - Internal Directory Connectors](#)
 - [2.3.4 - LDAP Directory Connectors](#)
 - [2.3.5 - Custom Directory Connectors](#)
- [2.4 - Application Manager](#)
 - [2.4.1 - Application Browser](#)
 - [2.4.2 - Adding an Application](#)
 - [2.4.3 - Managing an Integrated Application](#)
- [2.5 - Principal Manager](#)
 - [2.5.1 - Principal Browser](#)
 - [2.5.2 - Adding a Principal](#)
 - [2.5.3 - Managing a Principal](#)
- [2.6 - Group and Role Manager](#)
 - [2.6.1 - Group and Role Browser](#)
 - [2.6.2 - Adding a Group or Role](#)
 - [2.6.3 - Managing a Group or Role](#)

2.1 - About the Administration Console

This page last changed on Feb 21, 2007 by justen.stepka@atlassian.com.

After completing the installation process, you are now able to access the Crowd administration console. Through the console you will be able to accomplish the following:

- Configure applications that may access the Crowd framework.
- View active sessions and manually expire sessions.
- Add directory servers to manage users allowed access to integration applications.
- Adjust deployment properties configured during the setup process.
- Manage principals along with adjusting group and role memberships.
- Create new directory entities.

The welcome screen will look similar to the following:



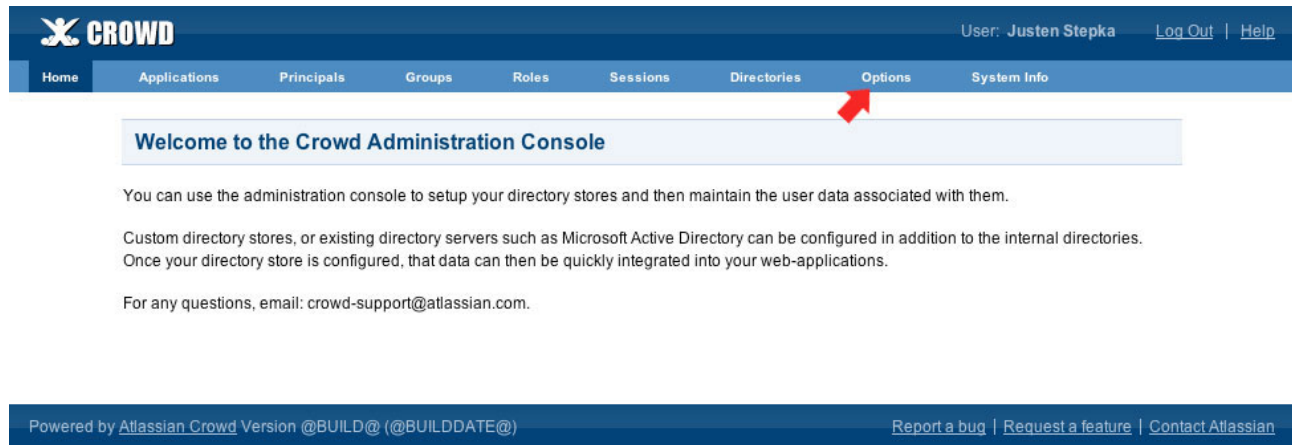
The screenshot shows the Crowd Administration Console interface. At the top is a dark blue header with the 'CROWD' logo on the left and 'User: Justen Stepka' with 'Log Out' and 'Help' links on the right. Below the header is a navigation bar with tabs: Home, Applications, Principals, Groups, Roles, Sessions, Directories, Options, and System Info. The main content area has a light blue header 'Welcome to the Crowd Administration Console'. Below this, it states: 'You can use the administration console to setup your directory stores and then maintain the user data associated with them.' It then explains: 'Custom directory stores, or existing directory servers such as Microsoft Active Directory can be configured in addition to the internal directories. Once your directory store is configured, that data can then be quickly integrated into your web-applications.' It concludes with: 'For any questions, email: crowd-support@atlassian.com.' At the bottom is a dark blue footer with 'Powered by Atlassian Crowd Version @BUILD@ (@BUILDDATE@)' on the left and 'Report a bug | Request a feature | Contact Atlassian' on the right.

2.2 - Options and Settings

This page last changed on Feb 27, 2007 by justen.stepka@atlassian.com.

Options

The Options tab enables you to view and edit the various server deployment parameters.



The screenshot displays the Atlassian Crowd Administration Console interface. At the top, there is a dark blue header bar with the 'CROWD' logo on the left and the user 'User: Justen Stepka' with links for 'Log Out' and 'Help' on the right. Below the header is a navigation menu with tabs: Home, Applications, Principals, Groups, Roles, Sessions, Directories, Options, and System Info. The 'Options' tab is highlighted with a red arrow. Below the navigation menu, a light blue box contains the text 'Welcome to the Crowd Administration Console'. The main content area has a light gray background and contains the following text: 'You can use the administration console to setup your directory stores and then maintain the user data associated with them.', 'Custom directory stores, or existing directory servers such as Microsoft Active Directory can be configured in addition to the internal directories. Once your directory store is configured, that data can then be quickly integrated into your web-applications.', and 'For any questions, email: crowd-support@atlassian.com.' At the bottom, a dark blue footer bar contains the text 'Powered by Atlassian Crowd Version @BUILD@ (@BUILDDATE@)' on the left and links for 'Report a bug', 'Request a feature', and 'Contact Atlassian' on the right.

2.2.1 - Caching

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

Caching

Caching is used to store run-time authentication and authorization rules which can be expensive to calculate. During developing cycles it is recommended caching be turned off and only enabled for production use.

- **Enable:** This option enables or disables the caching on the security server.
- **Cache TTL:** The Cache TTL option controls the amount of time in minutes that an item will remain in the memory cache before having to be reloaded.

The screenshot shows the Atlassian Crowd web interface. The top navigation bar includes the 'CROWD' logo and user information 'User: Justen Stepka' with links for 'Log Out' and 'Help'. Below this is a horizontal menu with tabs: Home, Applications, Principals, Groups, Roles, Sessions, Directories, Options (selected), and System Info. The main content area is titled 'Options' and contains several sub-tabs: Caching (selected), General, Licensing, Mail Server, Mail Template, and Session. Under the 'Caching' tab, there are two settings: 'Enable:' with a checked checkbox and a descriptive text block, and 'Cache TTL:' with a text input field containing the value '5' and a descriptive text block. At the bottom right of the settings area are 'Update »' and 'Cancel' buttons. The footer of the page contains the text 'Powered by Atlassian Crowd Version @BUILD@ (@BUILDDATE@)' and links for 'Report a bug', 'Request a feature', and 'Contact Atlassian'.

Options

Caching General Licensing Mail Server Mail Template Session

Enable: ☒
Enable or disable caching. Caching is used to store run-time authentication and authorization rules which can be expensive to calculate. During developing cycles, it is recommended caching be turned off and only enabled for production use.

Cache TTL:
The amount of time in minutes that an item will be cached before reloading. This value must be greater than 0.

Update » Cancel

Powered by Atlassian Crowd Version @BUILD@ (@BUILDDATE@) [Report a bug](#) | [Request a feature](#) | [Contact Atlassian](#)

2.2.2 - General Options

This page last changed on Mar 05, 2007 by justen.stepka@atlassian.com.

General

The general options are generic in the way the security server interacts with principals.

- **Deployment Title:** Specifies a unique name for this deployment.
- **Domain:** The domain the server is deployed for. This is used when setting security information on HTTP authentication cookies.



Be Careful

When developing on your local machine, the domain should be set to `localhost`. If you wish to have SSO support for `*.mydomain.com`, you will need to set the cookie to `.mydomain.com`. Please take notice of the `.` before the top-level-domain.

- **Token Seed:** This is a unique seed for each site deployment of crowd. This key is used when generating tokens for an authenticated client.

The screenshot shows the Atlassian Crowd web interface. The top navigation bar includes the 'CROWD' logo and a user profile for 'Justen Stepka' with links for 'Log Out' and 'Help'. Below this is a secondary navigation bar with tabs: 'Home', 'Applications', 'Principals', 'Groups', 'Roles', 'Sessions', 'Directories', 'Options' (which is active), and 'System Info'. The main content area is titled 'Options' and contains several sub-tabs: 'Caching', 'General' (selected), 'Licensing', 'Mail Server', 'Mail Template', and 'Session'. The 'General' tab displays three configuration fields: 'Deployment Title' with the value 'Atlassian' and a description 'The deployment title is the name of this instance.'; 'Domain' with the value 'localhost' and a description 'The domain for this deployment, i.e. acmecorp.com.'; and 'Token Seed' with the value 'jFQxnCYy' and a description 'The token seed is used to generate custom authentication tokens for your deployment.' At the bottom right of the form are three buttons: 'Generate', 'Update »', and 'Cancel'. The footer of the page contains the text 'Powered by Atlassian Crowd Version @BUILD@ (@BUILDDATE@)' and links for 'Report a bug', 'Request a feature', and 'Contact Atlassian'.

2.2.3 - Licensing

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.


Licensing

Crowd licensing is limited by the number of applications that are able to communicate with the security server. Evaluations licenses may be obtained from the [Atlassian](http://atlassian.com) website.



Be Careful

If a license is expired, application clients will not be able to use the integration API of the server.

 **CROWD**

User: Justen Stepka [Log Out](#) | [Help](#)

[Home](#) [Applications](#) [Principals](#) [Groups](#) [Roles](#) [Sessions](#) [Directories](#) [Options](#) [System Info](#)

Options

[Caching](#) [General](#) [Licensing](#) [Mail Server](#) [Mail Template](#) [Session](#)

Licensee:	Atlassian Software Systems
Type:	Crowd: Commercial
Purchased:	Monday, 27 Nov 2006
Support Period:	Your commercial Crowd support and updates are available until Wednesday, 28 Nov 2007
User Limit:	500
Current Users:	1
License Server ID:	A5KB-0LHH-A163-BMHD
License:	<div></div> <div>An evaluation license key is available from the Atlassian website.</div>

[Update »](#) [Cancel](#)


Powered by [Atlassian Crowd](#) Version @BUILD@ (@BUILDDATE@) [Report a bug](#) | [Request a feature](#) | [Contact Atlassian](#)

2.2.4 - Mail Server

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

Mail Server

The security server will send notices to users during special events such as when a password is reset. Enter the details of your mail server with a username and password if required.

 User: Justen Stepka [Log Out](#) | [Help](#)

[Home](#) [Applications](#) [Principals](#) [Groups](#) [Roles](#) [Sessions](#) [Directories](#) **Options** [System Info](#)

Options

[Caching](#) [General](#) [Licensing](#) **[Mail Server](#)** [Mail Template](#) [Session](#)

Notification Email:
Email address to send server messages to when server notifications occur.

SMTP Host:
The host address, i.e. localhost or smtp.acmecorp.com.

From:
The sender (or FROM) address to use when sending email notifications.

Subject Prefix:
The subject prefix to use when sending email notifications. This is useful for mail client filtering rules, i.e. [ACME CORP - Crowd].

Username:
The username to use when connecting to the mail server.

Password:
The password to use when connecting to the mail server.

[Update »](#) [Cancel](#)

Powered by [Atlassian Crowd](#) Version @BUILD@ (@BUILDDATE@) [Report a bug](#) | [Request a feature](#) | [Contact Atlassian](#)


2.2.5 - Mail Template

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

Mail Template

The email template used when resetting a principal's password. The supported template macros are available:

- `$firstname`: will be replaced by the principal's first name.
- `$lastname`: will be replaced by the principal's first name.
- `$deploymenttitle`: will be replaced by the principal's first name.
- `$date`: will be replaced by time of the message event.
- `$password`: will be replaced by principal's password.

 User: [Justen Stepka](#) | [Log Out](#) | [Help](#)

[Home](#) | [Applications](#) | [Principals](#) | [Groups](#) | [Roles](#) | [Sessions](#) | [Directories](#) | **[Options](#)** | [System Info](#)

Options

[Caching](#) | [General](#) | [Licensing](#) | [Mail Server](#) | **[Mail Template](#)** | [Session](#)

Template:

Hello `$firstname` `$lastname`,

Your password has been reset by a `$deploymenttitle` administrator at `$date`.

Your new password is: `$password`

`$deploymenttitle` Administrator

The email template is used when resetting a principals password. The supported macros are `$firstname` (firstname), `$lastname` (lastname), `$deploymenttitle` (Crowd deployment title), `$date` (message date), and `$password` (new password).

[Update »](#) | [Cancel](#)

Powered by [Atlassian Crowd](#) Version @BUILD@ (@BUILDDATE@) | [Report a bug](#) | [Request a feature](#) | [Contact Atlassian](#)

2.2.6 - Session

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

Session Options

When a success authentication occurs, a unique token is assigned. Tokens are valid for the period of time specified as the `Session Timeout` attribute. Expired tokens are removed from the databaes by the interval `Reaper Time in minutes`. When a Token is validated, the `Session Timeout` last validation is reset to 0.

- **Session Timeout:** Controls how long a session will be considered valid during any period of inactivity. This is in minutes and must be greater than 0.

The screenshot shows the Atlassian Crowd web interface. At the top is a dark blue header with the 'CROWD' logo on the left and 'User: Justen Stepka' with 'Log Out' and 'Help' links on the right. Below the header is a navigation bar with tabs: Home, Applications, Principals, Groups, Roles, Sessions, Directories, Options (selected), and System Info. The main content area is titled 'Options' and contains several sub-tabs: Caching, General, Licensing, Mail Server, Mail Template, and Session (selected). Under the 'Session' tab, there is a 'Session Timeout' label followed by a text input field containing the number '5'. Below the input field is a small explanatory text: 'How long a session will be valid for before expiring. This is in minutes and must be greater than 0.' At the bottom right of the form are two buttons: 'Update »' and 'Cancel'. The footer of the page is a dark blue bar containing 'Powered by Atlassian Crowd Version @BUILD@ (@BUILDDATE@)' on the left and 'Report a bug | Request a feature | Contact Atlassian' on the right.

2.3 - Directory Manager

This page last changed on Feb 27, 2007 by justen.stepka@atlassian.com.


The Crowd security server supports an unlimited number of configured directory servers. These configured directories can be internal user stores, an Crowd connector (Active Directory, Open Directory, etc) or a custom directory connector such as a legacy database. Mapped directories are responsible for authentication, authorization and modification of directory entities.

2.3.1 - Directory Browser

This page last changed on Feb 27, 2007 by justen.stepka@atlassian.com.

Directory Browser

The Directory Browser allows administrators to view and search for configured directory servers.

 **CROWD**

User: Justen Stepka [Log Out](#) | [Help](#)

Home Applications Principals Groups Roles Sessions **Directories** Options System Info

Directories Browser [Add Directory](#) | [Import Users](#)

Name : Active : Results Per Page :

Name	Active	Type	Action
Atlassian	true	Crowd Internal Directory	View
maltshovel	true	Microsoft Active Directory	View

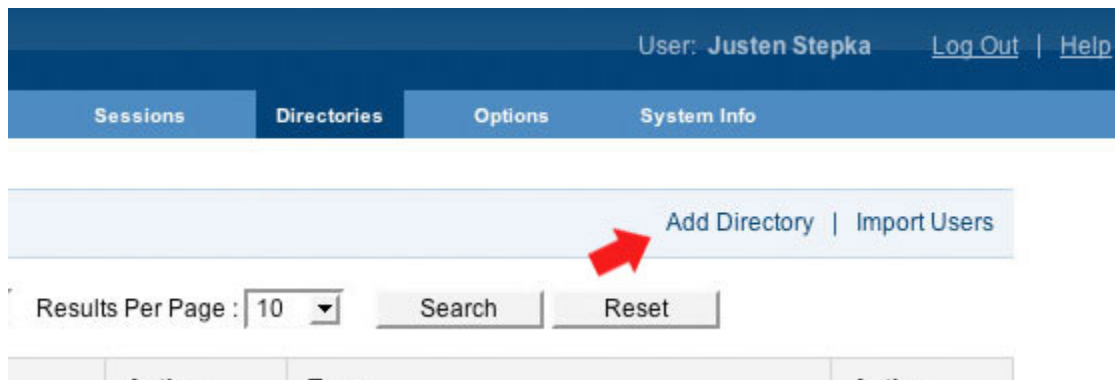
Powered by [Atlassian Crowd](#) Version 0.4.5 (Build #999: Feb 26, 2007) [Report a bug](#) | [Request a feature](#) | [Contact Atlassian](#)

2.3.2 - Adding a Directory Connector

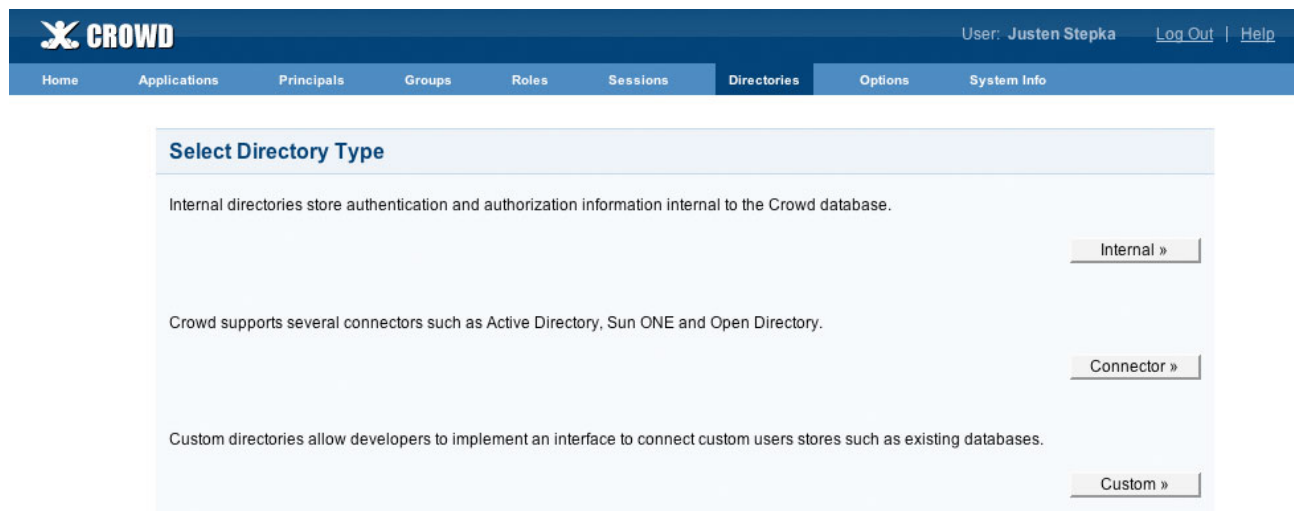
This page last changed on Feb 27, 2007 by justen.stepka@atlassian.com.

Adding a Directory Server

Adding a directory server is a very straight forward process. You will need to click the 'Add Directory' option when browsing or editing an integrated directory server.



The first step involved is selecting the type of directory server to be added. Crowd provides support for three types of directory servers:



Directory Type	Description
Internal	Internal directories store authentication and authorization information internal to the Crowd database.
Connector	Crowd support several LDAP connectors such as Active Directory, SunONE and Open Direcotry.
Custom	Custom directories allow developers to implement an interface t connect customer users stores such as an existing database or legacy system.

Regardless of the directory server type, all directories share a set of generic values that can be configured per directory server. The following are the generic directory configuration parameters:

Attribute	Description
Name	The name of the directory is used to categorize the directory instance. This is useful when there are multiple directories configured, i.e. Chicago Employees or Web Customers.
Description	Details about this specific directory.
Active	If the directory server mapping is active or not.
Add Group	Allow groups to be added to the directory.
Add Principal	Allow principals to be added in the directory.
Add Role	Allow roles to be added in the directory.
Modify Group	Allow groups to be modified in the directory.
Modify Principal	Allow principal to be modified to the directory.
Modify Role	Allow roles to be modified to the directory.
Remove Group	Allow groups to be removed from the directory.
Remove Principal	Allow principals to be removed from the directory.
Remove Role	Allow roles to be removed from the directory.

Directory permissions serve a special purpose by restricting the way directories can be used. Often, administrators may want to limit application clients to only being able to read directory server entity data. Through the permissions setting, these enforcements are possible.

Create Internal Directory

Details **Permissions**

Add Group:	<input checked="" type="checkbox"/>	Allow groups to be added to the directory.
Add Principal:	<input checked="" type="checkbox"/>	Allow principals to be added to the directory.
Add Role:	<input checked="" type="checkbox"/>	Allow roles to be added to the directory.
Modify Group:	<input checked="" type="checkbox"/>	Allow groups to be modified to the directory.
Modify Principal:	<input checked="" type="checkbox"/>	Allow principals to be modified to the directory.
Modify Role:	<input checked="" type="checkbox"/>	Allow roles to be modified to the directory.
Remove Group:	<input checked="" type="checkbox"/>	Allow groups to be removed from the directory.
Remove Principal:	<input checked="" type="checkbox"/>	Allow principals to be removed from the directory.
Remove Role:	<input checked="" type="checkbox"/>	Allow roles to be removed from the directory.

[Continue »](#) [Cancel](#)

Directory Specific Configuration Options

For more information about configuring a specific directory type, see one of the sections listed below:

- [2.3.4 - LDAP Directory Connectors](#)
- [2.3.5 - Custom Directory Connectors](#)
- [2.3.3 - Internal Directory Connectors](#)

2.3.3 - Internal Directory Connectors

This page last changed on Feb 27, 2007 by justen.stepka@atlassian.com.

Internal Directories

Internal directory stores are provisioned repositories where authentication and authorization information can be stored. Crowd supports an unlimited number of groups, principals and roles in an unlimited number of provisions. This allows administrators to create silos of users such as: customers and employees all the while separating out security information.

Create Internal Directory

Details	Permissions
Name:	<input type="text"/> <small>The name of the directory is to categorize the directory instance. This is useful when there are multiple directories configured, i.e. Chicago Employees or Web Customers.</small>
Description:	<input type="text"/> <small>Details about this specific directory.</small>
Active:	<input checked="" type="checkbox"/>
Password Regex:	<input type="text"/> <small>Regex pattern which new passwords will be validated against. Leave blank to disable this feature.</small>
Maximum Invalid Password Attempts:	<input type="text" value="0"/> <small>The maximum number of invalid password attempts before the authenticating account will be disabled. Enter 0 to disable this feature.</small>
Maximum Unchanged Password Days:	<input type="text" value="0"/> <small>The number of days until the password must be changed. This value is in days, enter 0 to disable this feature.</small>
Password History Count:	<input type="text" value="0"/> <small>The number of previous passwords to prevent the principal from using. Enter 0 to disable this feature.</small>
<div>Continue » Cancel</div>	

Internal Directory Attributes	Description
Password Regex	Regex pattern which new passwords will be validated against. Leave blank to disable this feature.
Maximum Invalid Password Attempts	The maximum number of invalid password attempts before the authenticating account will be disabled. Enter 0 to disable this feature.
Maimum Unchanged Password Days	The number of days until the password must be changed. This value is in days, enter 0 to disable this feature.
Password History Count	The number of previous passwords to prevent the principal from using. Enter 0 to disable this feature.

2.3.4 - LDAP Directory Connectors

This page last changed on Mar 05, 2007 by justen.stepka@atlassian.com.

LDAP Connectors

Crowd offers pre-built connectors for the most popular directory servers such as: Microsoft Active Directory, Apple OS X, and SunONE. These LDAP connectors enable administrators and developers to quickly integrate desktop logins to existing web-applications.

The first step when setting up an connector is to select the connector type and fill in the basic connection information for the directory server:

Connector: * Microsoft Active Directory

The directory connector to use when communicating with the directory server. Custom directory connectors can be configured if an out-of-box connector is not supplied, code examples are included with the Crowd download.

Once selecting a Connector, various LDAP object and attribute settings may be adjusted by selecting the Configuration tab. Here the node and attributes of the specific LDAP server may be modified. Generic settings have been provided by based on the Connector selected.

Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, for example ldap://localhost:389 or port 639 for SSL.
Secure SSL	Specifies if the connection to the directory server is a SSL connection.
Base DN	Enter the root distinguished name to use when running queries versus the directory server, for example, o=acmecorp,c=com.
User DN	Connect to the directory server using the supplied username.
Password	Connect to the directory server using the supplied password.

Active Directory

Active Directory Attribute Example	Value
Base DN	cn=users,dc=ad,dc=acmecorp,dc=com
User DN	administrator@ad.acmecorp.com

For Microsoft AD the base is of the format `dc=domain1,dc=local`. You will want to replace the `domain1`

and `local` to your specific configuration. Microsoft Server provides a tool called `ldp.exe` which is useful for finding out and configuring the the LDAP structure of your server.

The URL for MS AD should will be in the format of `ldap://domainname`.

Apple OSX Open Directory

Apple OS X Open Directory Example	Value
Base DN	<code>dc=acmecorp,dc=com</code>
User DN	<code>cn=Manager,dc=acmecorp,dc=com</code>

SunONE

SunONE Directory Example	Value
Base DN	<code>dc=acmecorp,dc=com</code>
User DN	<code>cn=Directory Manager</code>

OpenLDAP

OpenLDAP Directory Example	Value
Base DN	<code>dc=exampel,dc=com</code>
User DN	<code>cn=Manager,dc=example,dc=com</code>

The OpenLDAP connector only work with version 2.3.X and higher. Previous versions do not support the paging attribute and will result in the following error:

LDAP_UNAVAILABLE_CRITICAL_EXTENSION: Indicates that the LDAP server was unable to satisfy a request because one or more critical extensions were not available. Either the server does not support the control or the control is not appropriate for the operation type.

Configuration Details

When configuring your LDAP server, if you are using non-standard object types, you will need to adjust the default filter and object type configurations. Default values are configured for the integrated LDAP servers. If your connector is added successfully, but you unable to see an data when browsing your LDAP server it is likely your object and filters are configured incorrectly.

User Configuration

User DN:

This value is used in addition to the base DN when searching and loading users, an example is ou=Users. If no value is supplied, the subtree search will start from the base DN.

User Object Class:

*

The LDAP user object class type to use when loading principals.

User Object Filter:

*

The filter to use when searching user objects.

User Name Attribute:

*

The attribute field to use when loading the principal username.

Group Configuration

Attribute	Description
Group DN	This value is used in addition to the base DN when searching and loading groups, an example is ou=Groups. If no value is supplied, the subtree search will start from the base DN.
Group Object Class	This value is used in addition to the base DN when searching and loading groups, an example is ou=Groups. If no value is supplied, the subtree search.
Group Object Filter	The filter to use when searching group objects.
Group Name Attribute	The attribute field to use when loading the group name.
Group Description Attribute	The attribute field to use when loading the group description.
Group Members Attribute	The attribute field to use when loading the group members.

Role Configuration

Attribute	Description
Role DN	This value is used in addition to the base DN when searching and loading roles, an example is ou=Roles. If no value is supplied, the subtree search will start from the base DN.
Role Object Class	This value is used in addition to the base DN when searching and loading roles, an example is ou=Roles. If no value is supplied, the subtree search.
Role Object Filter	The filter to use when searching role objects.
Role Name Attribute	The attribute field to use when loading the role name.

Role Description Attribute	The attribute field to use when loading the role description.
Role Members Attribute	The attribute field to use when loading the role members.

Principal Configuration

Attribute	Description
User DN	This value is used in addition to the base DN when searching and loading users, an example is ou=Users. If no value is supplied, the subtree search will start from the base DN.
User Object Class	The LDAP user object class type to use when loading principals.
User Object Filter	The filter to use when searching user objects.
User Name Attribute	The attribute field to use when loading the principal username.
User First Name Attribute	The attribute field to use when loading the principal first name.
User Last Name Attribute	The attribute field to use when loading the principal last name.
User Email Attribute	The attribute field to use when loading the principal email.
User Group Attribute	The attribute field to use when loading the principal's groups.
User Password Attribute	The attribute field to use when manipulating a principal password.

LDAP Object Structures

Active Directory

The Active Directory LDAP connector assumes that all LDAP object types are of the default structure. Any changes to the default object structure of the `User` and `Group` objects will require a custom connector to be coded.

LDAP Connector Object Structures

The Crowd LDAP connectors assume that all container objects (groups and roles) have the full DN to the associated member. As of now the membership attributes on a Principal object are not used, however in the future these associations may be used to assist with performance when looking up memberships.

Supported Object Types:

- groupOfUniqueNames
- inetorgperson

Non-supported Object types:

The following object types are not supported because of the required `guiNumber` attribute. Crowd does not currently support the adding of unique

- posixGroup
- posixUser



Zimbra Mail Server LDAP Types

Principal objects have been tested and are known to work with the `zimbraAccount` LDAP object types.

2.3.5 - Custom Directory Connectors

This page last changed on Feb 27, 2007 by justen.stepka@atlassian.com.

Custom Directories

Custom directories allow developers to implement an interface to connect custom users stores such as existing databases or legacy system. The simplest solution to accomplish this is to add a JAR file with the necessary classes to the Crowd `WEB-INF/lib` application deployment folder.

Create Custom Connector

Details

Permissions

Name:

*

The name of the directory is to categorize the directory instance. This is useful when there are multiple directories configured, i.e. Chicago Employees or Web Customers.

Description:

Details about this specific directory.

Active:

☒

Implementation Class:

*

Implementation of `com.atlassian.crowd.integration.directory.RemoteDirectory` Java interface. Must be in the Crowd CLASSPATH.

Continue »

Cancel

Custom Directory Store Attributes	Description
Implementation Class	Implementation of <code>com.atlassian.crowd.integration.directory.RemoteDirectory</code> Java interface. Must be in the Crowd CLASSPATH.

Full Javadoc for the `RemoteDirectory` interface can be found here:

<http://docs.atlassian.com/crowd/current/com/atlassian/crowd/integration/directory/RemoteDirectory.html>

2.4 - Application Manager

This page last changed on Feb 27, 2007 by justen.stepka@atlassian.com.

The Crowd security server integrates and provisions application as the concept of Applications. A Configured application is mapped to an integrated directory server, where groups are then configured to access the application.


Communication with the Crowd security server by an application will only occur when client credentials are validated and a known host address is used by the connecting application.

2.4.1 - Application Browser

This page last changed on Feb 27, 2007 by justen.stepka@atlassian.com.

Application Browser

The Application Browser allows administrators to view and search for configured directory servers.

 User: Justen Stepka [Log Out](#) | [Help](#)

[Home](#) | [Applications](#) | [Principals](#) | [Groups](#) | [Roles](#) | [Sessions](#) | [Directories](#) | [Options](#) | [System Info](#)

Applications Browser

[Add Application](#)

Name : Active : Results Per Page :

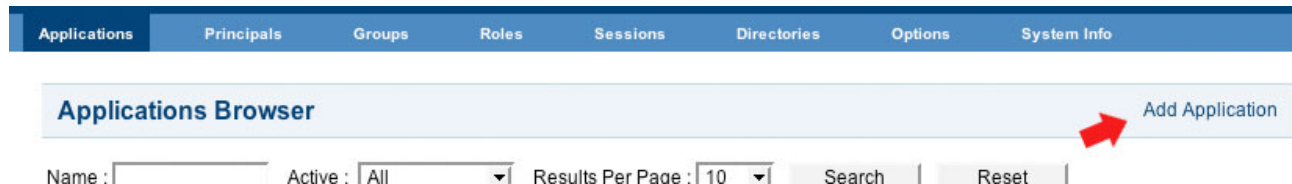
Name	Active	Description	Action
confluence	true		View
crowd	true	Atlassian Crowd - Java SSO & Identity Management	View
demo	true	Crowd demo feature highlight application.	View
jira	true		View

Powered by [Atlassian Crowd](#) Version 0.4.5 (Build #999: Feb 28, 2007) [Report a bug](#) | [Request a feature](#) | [Contact Atlassian](#)

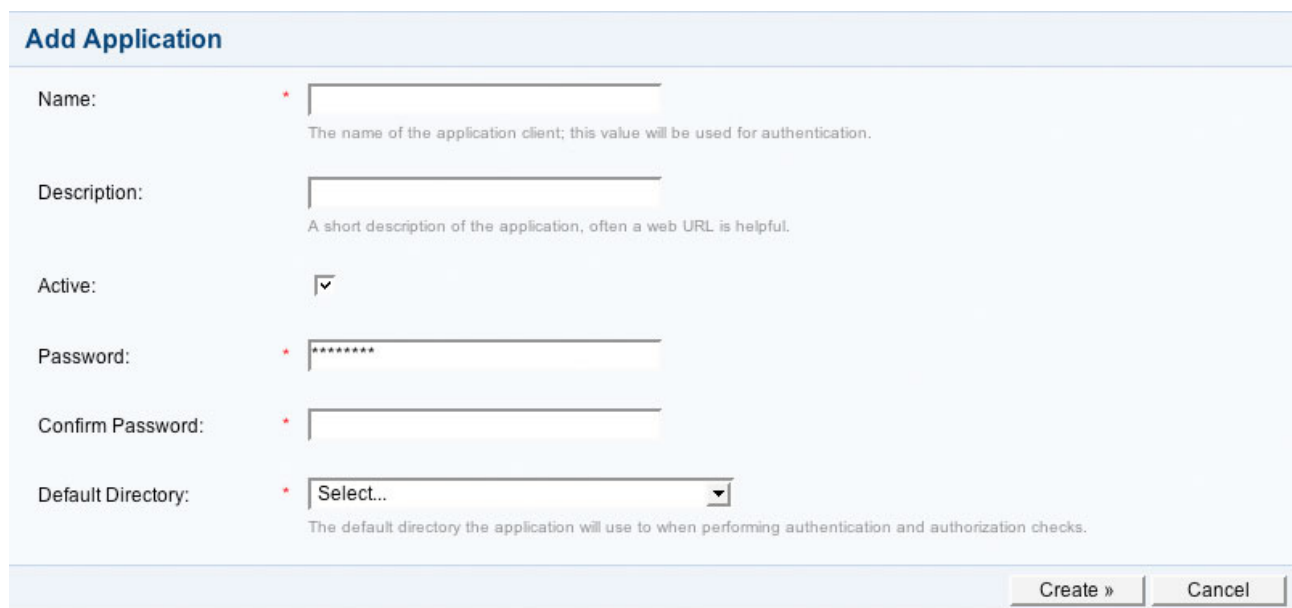
2.4.2 - Adding an Application

This page last changed on Feb 27, 2007 by justen.stepka@atlassian.com.

Adding a Directory Server



Adding a directory server is a very straight forward process. You will need to click the 'Add Directory' option when browsing or editing an integrated directory server.



Attribute	Description
Name	The name of the application client; this value will be used for authentication. This value must be unique and can used by more than one application client.
Description	A short description of the application, often a web URL is helpful.
Active	If the application client is active or not.
Password	The password for the authenticating application client.
Default Directory	The password for the authenticating application client.

Additional options will be available upon the creation of the new Application.

- [2.4.3 - Managing an Integrated Application](#)

2.4.3 - Managing an Integrated Application

This page last changed on Feb 27, 2007 by justen.stepka@atlassian.com.

Managing an Application Client

The management of an application client through the administration console allows administrators to configure the authentication, linked directories, access groups and valid application host that are allowed to make API calls versus the Crowd security server.

View Application – crowd[Add Application](#) | [Remove Application](#)

DetailsDirectoriesGroupsRemote AddressesConfig Test

Name:
The name of the application client; this value will be used for authentication.

Description:
A short description of the application, often a web URL is helpful.

Active:☒

Conception:26 Feb 2007, 16:08:00

Last Modified:26 Feb 2007, 16:08:00

Password:
To set a new password, enter the password and confirm. Leave blank to make no changes when updating.

Confirm Password:

Update »

Cancel

Application Directory Mappings

Directory mappings control which user stores will be used when authenticating and authorizing a principal's access request. For a principal to be considered a valid application user, their account must belong to a group that is assigned to the application. The order of the assigned directories can be configured as necessary.

The Directory listed first will be called when authentication or authorization calls are necessary. If the security call can be processed by the associated directory the operation will then return the result. If the call can not be processed, the next directory in the list will then be used when running the security call until all directory servers have been exhausted.

If the security call can not be processed, an `Exception` based on the method will be thrown.

To allow all users from a directory to authenticate, change the `Allow all to Authenticate` option to `true`. This will allow anyone in the configured directory to authenticate with application.

Directory	Allow all to Authenticate	Action
Atlassian	False	Remove

Application Group Mappings

Group mappings control which principals are allowed to authenticate versus the application. If the principal is a member of an assigned group their authentication will be valid.

Application Addresses

Address mappings are used to validate the remote address of a requesting application client.

View Application – crowd
[Add Application](#) | [Remove Application](#)

Details Directories Groups **Remote Addresses** Config Test

Address mappings are used to validate the remote address of a requesting application client. For an application to be able to use the remote API of this security server, the address must be valid and active.

Address	Status	Action
192.168.0.110	True	Remove
127.0.0.1	True	Remove
localhost	True	Remove

Address:



Common Misconfiguration

For an application to be able to use the remote API of the security server, the client address must be valid and active.

Configuration Test.

The Configuration Test tool allows you to validate that user is valid for an application. Authentications are valid only when a group the user is a member of is enabled to login to an application. If the username and password provided are valid but the authentication check fails, the security settings for the integrated Directories or Groups will need to be adjusted.

2.5 - Principal Manager

This page last changed on Feb 28, 2007 by justen.stepka@atlassian.com.


The Principal Manager allows administrators to view, add and edit principal entity objects within a configured directory server.

2.5.1 - Principal Browser

This page last changed on Feb 28, 2007 by justen.stepka@atlassian.com.

Principal Browser

The Principal Browser allows administrators to view and search for principals within a specified directory server. Additional search options such as `username` and `email` may be specified.

 User: Justen Stepka [Log Out](#) | [Help](#)

[Home](#) [Applications](#) **[Principals](#)** [Groups](#) [Roles](#) [Sessions](#) [Directories](#) [Options](#) [System Info](#)

Principals Browser [Add Principal](#) | [Import Users](#)

Directory : Username : Email : Active :

Results Per Page :

Username	Email	Action
jstepka	justen.stepka@atlassian.com	View

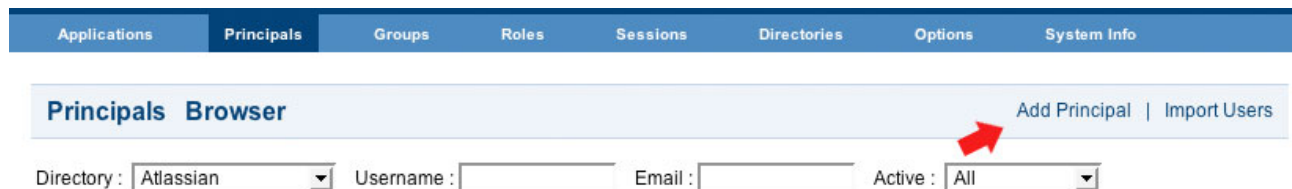
Powered by [Atlassian Crowd](#) Version 0.4.5 (Build #999: Feb 28, 2007) [Report a bug](#) | [Request a feature](#) | [Contact Atlassian](#)

2.5.2 - Adding a Principal

This page last changed on Feb 28, 2007 by justen.stepka@atlassian.com.

Adding a Principal

Adding a principal is a very straight forward process. You will need to click the 'Add Principal' option when browsing or editing an integrated directory server.



Adding a new principal requires entering general identity information. Within a given directory server, the `username` must be unique. If the permission settings for adding a principal are not enabled for the selected directory server, the addition of the entity will result in a permission exception.



Attribute	Description
Email	Email address of the principal.
Active	If the group or role is active or not.
Username	Unique name of the principal.
Active	If the group or role is active or not.
Password	The password for the authenticating principal.
First Name	The first name of the principal.

Last Name	The last name of the principal.
Directory	The directory the principal will be added to.

2.5.3 - Managing a Principal

This page last changed on Feb 28, 2007 by justen.stepka@atlassian.com.

The Principal Manager allows administrators to view, add and edit principal entity objects within a configured directory server.

Managing Principals

The management of principals through the administration console allows administrators to adjust the entity attributes of a principal along with group and role memberships.

View Principal – jstepka[Add Principal](#) | [Reset Password](#) | [Remove Principal](#)

Details | Attributes | Groups | Roles

Username:	jstepka
Directory:	Atlassian — Crowd Internal Directory
Email:	<input type="text" value="justen.stepka@atlassian.com"/> <small>Email addresses must follow the RFC2822 format.</small>
Active:	<input checked="" type="checkbox"/>
First Name:	<input type="text" value="Justen"/>
Last Name:	<input type="text" value="Stepka"/>
Password:	<input type="password"/>
Confirm Password:	<input type="password"/>

Attributes

Entity attributes such as addresses, date of birth and other principal entity information are maintained through the attributes editor.

View Principal – jstepka

[Add Principal](#) | [Reset Password](#) | [Remove Principal](#)

Details

Attributes

Groups

Roles

Attribute	Values	Action
givenName	<input type="text" value="Justen"/>	Remove
invalidPasswordAttempts	<input type="text" value="0"/>	Remove
lastAuthenticated	<input type="text" value="1172643042268"/>	Remove
mail	<input type="text" value="justen.stepka@atlassian.com"/>	Remove
passwordLastChanged	<input type="text" value="1172466480524"/>	Remove
requiresPasswordChange	<input type="text" value="false"/>	Remove
sn	<input type="text" value="Stepka"/>	Remove

Attribute : Value : [Add »](#) [Update »](#) [Cancel](#)

Group and role Memberships

The memberships a principal belongs to be may be adjusted by adding or removing the various group and role objects that are maintained for the directory server the principal belongs to.

View Principal – jstepka

[Add Principal](#) | [Reset Password](#) | [Remove Principal](#)

Details

Attributes

Groups

Roles

These are the groups the principal is a member of.

Group	Action
crowd-administrators	Remove

[Add »](#) [Update »](#) [Cancel](#)

2.6 - Group and Role Manager

This page last changed on Mar 04, 2007 by justen.stepka@atlassian.com.


The Group and Role Managers allow administrators to view, add and edit the various permission container objects stored within a configured directory server.

2.6.1 - Group and Role Browser

This page last changed on Mar 04, 2007 by justen.stepka@atlassian.com.

Principal Browser

The Principal Browser allows administrators to view and search for groups or roles within a specified directory server. Additional search options such as `name` and `active` may be specified.

 **CROWD**

User: Justen Stepka [Log Out](#) | [Help](#)

Home Applications Principals **Groups** Roles Sessions Directories Options System Info Backup & Restore

Group Browser

[Add Group](#)

Directory : Name : Active : Results Per Page :

Name	Active	Action
Administrators	true	View
Users	true	View
Guests	true	View
Print Operators	true	View
Backup Operators	true	View
Replicator	true	View
Remote Desktop Users	true	View
Network Configuration Operators	true	View
Performance Monitor Users	true	View
Performance Log Users	true	View

[« Previous](#)[Next »](#)

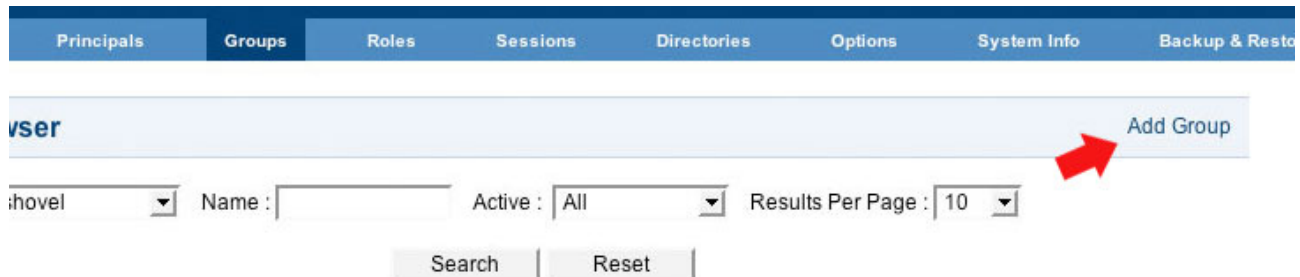
Powered by [Atlassian Crowd](#) Version 1.0.0 (Build #100: Mar 05, 2007) [Report a bug](#) | [Request a feature](#) | [Contact Atlassian](#)

2.6.2 - Adding a Group or Role

This page last changed on Mar 04, 2007 by justen.stepka@atlassian.com.

Adding a Group or Role

Adding a group is a very straight forward process. You will need to click the 'Add Group' option when browsing or editing an integrated directory server.



The screenshot shows the 'Groups' tab selected in a navigation bar. Below the navigation bar, there is a search bar with a dropdown menu showing 'shovel'. To the right of the search bar, there are fields for 'Name', 'Active' (set to 'All'), and 'Results Per Page' (set to '10'). Below these fields are 'Search' and 'Reset' buttons. On the far right, there is a link labeled 'Add Group' which is highlighted with a red arrow.

The name, directory and active status of the container object are required. If the permission settings for adding a group or role are not enabled for the selected directory server, the addition of the entity will result in a permission exception.

Attribute	Description
Name	The unique name of the group and role.
Description	A short description of the group or role.
Directory	The directory the group or role will be added to.
Active	If the group or role is active or not.



The 'Add Group' form contains the following fields:

- Name:** A text input field with a red asterisk indicating it is required. Below the field is the text 'The unique name of the group.'
- Description:** A text input field. Below the field is the text 'Description of the group.'
- Directory:** A dropdown menu with a red asterisk indicating it is required. The selected value is 'Select...'. Below the dropdown is the text 'The directory the group belongs to.'
- Active:** A checkbox that is currently checked.

At the bottom right of the form are two buttons: 'Create »' and 'Cancel'.

2.6.3 - Managing a Group or Role

This page last changed on Mar 05, 2007 by justen.stepka@atlassian.com.

Managing a Group or Role

The management of groups and roles through the administration console is somewhat limited because of the way container objects work. Principals are assigned to groups and roles, additionally groups are allowed to log into applications.

View Group – crowd-administrators[Add Group](#) | [Remove Group](#)

Details

Name:	crowd-administrators
Directory:	Atlassian — Crowd Internal Directory
Description:	<input type="text"/>
Active:	<input checked="" type="checkbox"/>

[Update »](#)[Cancel](#)

3. - Integration Guide

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

- [3.1 - Integration Overview](#)
- [3.2 - Client Configuration](#)
- [3.3 - Integration Libraries](#)
- [3.4 - SOAP API](#)
- [3.5 - .NET Client](#)

3.1 - Integration Overview

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

Integration is very straight forward with Crowd. When performing a security request to the server the following happens:

- The application client authenticates with the security server, this token may be reused by the application client by followup calls. During this step the security server will validate the client's credentials and the remote address verses known client addresses.
- Using the authenticated token from the previous step, the application client then is able to perform the security request.

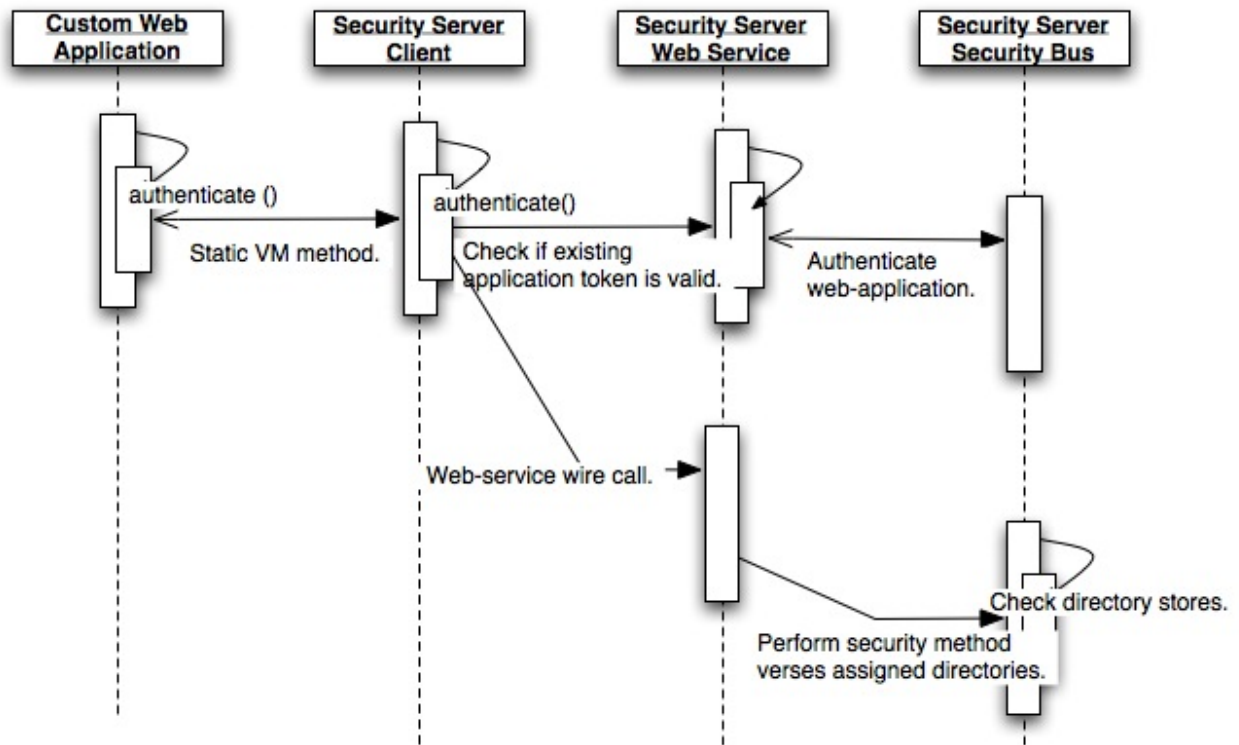
Crowd ships with pre-built integration classes that handle the authentication and token management for application clients. Should the requesting token become invalid, the client library will attempt to re-authenticate and perform the security request. If the second authentication request fails, an exception is thrown specifying the application client credentials are invalid.

The Crowd framework allows the application client to perform the following:

- Authenticate a principal.
- Validate and invalidate an existing principal authentication.
- Find a principal by their authentication token.
- Search principals, groups and roles by name or attributes
- Add principals, groups and roles.
- Validate principal group and role membership.
- Add and remove principals from groups and roles.
- Update principal attribute data.
- Update or reset and principal's authentication credentials.

As a reference, bundled with the download archive, is the source to the demo application. The demo application highlights best practices when using the Crowd framework. The Crowd download archive contains the entire source to the demo application, which can be used as an example when integrating your web applications. The source to the demo application is located in the src folder of the download archive.

Communication Sequence Example



3.2 - Client Configuration

This page last changed on Feb 26, 2007 by rosie@atlassian.com.

Configuring your application clients to communicate with the Crowd framework merely takes a moment. The integration libraries and configuration files are part of the downloaded distribution, this folder is client. You will find the Crowd integration library, and the client libraries the framework depends on, in the `lib` folder. An example client properties file `crowd.properties` is located in the `conf` folder.

To configure your application client, perform the following:

- Copy the crowd client and supporting libraries to your application classpath, typically `WEB-INF/lib`.
 - These files will be in the `client` folder similar to `crowd-core-0.4.1.jar` and all supporting jars in the `client/lib` folder.
- Copy the client properties file `crowd.properties` to your applications deployment directory, typically `WEB-INF/classes`.
- Edit the `crowd.properties` file to reflect the values of your deployment parameters.

A description of the `crowd.properties` attributes as follows:

Parameter	Description
<code>application.name</code>	The application name to use when authenticating with the Crowd server.
<code>application.password</code>	The application password to use when authenticating with the Crowd server.
<code>application.login.url</code>	The path to redirect the principal should their token expire or invalid due to security restrictions.
<code>crowd.server.url</code>	The URL to use when connecting with the integration libraries to communicate with the Crowd server.
<code>session.isauthenticated</code>	The session key to use when storing a <code>Boolean</code> value if the principal is authenticated or not.
<code>session.tokenkey</code>	The session key to use when storing a <code>String</code> value of the principal's authentication token.
<code>session.validationinterval</code>	The number of minutes between authentication validation. If this value is set to 0, each HTTP request will be authenticated.
<code>session.lastvalidation</code>	The session key to use when storing a <code>Date</code> value of the principal's last authentication.



Handy Hint

When configuring your client-application through Crowd, make sure to do the following:

- Assign a group to an application that is allowed to login.
- Add the IP of the client-application. Crowd restricts communication to specific IPs for client-applications making API calls.

Developing your own SOAP Client

Developing your own integration API for the Crowd server requires writing an application client that is capable of handling SOAP requests. To obtain the WSDL of the security server, access the URL:
<http://myserver/crowd/services/SecurityServer?wsdl>

The Crowd API has been tested with: Axis 1/2, Microsoft .NET and XFire.

3.3 - Integration Libraries

This page last changed on Mar 05, 2007 by justen.stepka@atlassian.com.

HttpAuthenticator

The `HttpAuthenticator` simplifies the authentication of HTTP based clients. When an authentication or invalidation is performed, the `HttpAuthenticator` manages the setting and resetting of integration variables for the principals HTTP session. If the application client has little need beyond authentication and validation, the `HttpAuthenticator` is a simple and very straight forward integration piece, shown below is a code example of authenticating and logging off of a principal.

Example 1:

```
HttpAuthenticator.authenticate(request, response, username, password);
```

Example 2:

```
HttpAuthenticator.authenticate(request, response);
```

If there were any issues with the authentication or logoff calls, an `Exception` will be thrown to the application client.

The `HttpAuthenticator` manages the following:

- Authenticating an HTTP request, and setting the session with the correct attributes for other integration points of the IDX framework.
- Invalidating an HTTP request includes removing session related attributes.
- Obtaining a principal's authenticated token from a session or browser cookie.
- Validating an existing HTTP authentication for single sign-on. If another application in the same domain has already authenticated the principal, the `HttpAuthenticator` will attempt to validate the existing authentication.
- Building a standard `AuthenticationContext` for a principal. This can be used to assure the authentication is consistent across all clients when setting validation factors of the client.

VerifyTokenFilter

The `VerifyTokenFilter` is a http servlet filter that protects secured resources by verifying the session or cookie token is active and the principal has access to the requesting application. The token filter works in conjunction with the `HttpAuthenticator` validating and setting various session and cookie attributes. Should the principal's token become expired or invalid due to security restrictions, the principal will be redirected to the URL provided by the `crowd.properties`.

Using the token filter is very straight forward, simply edit your `web.xml` deployment descriptor to reflect the filter and desired resource mapping:

```
<filter>
  <filter-name>VerifyTokenFilter</filter-name>
  <filter-class>com.atlassian.crowd.integration.http.VerifyTokenFilter</filter-class>
</filter>

<filter-mapping>
  <filter-name>VerifyTokenFilter</filter-name>
  <url-pattern>/secure/*</url-pattern>
</filter-mapping>
```

In this example, the verify token filter will prevent any pages on the /secure/ path from being accessed unless a valid token is found.

Should the token expire or be found invalid, the original url will be stored in the principal's session at a String with the key of `VerifyTokenFilter.ORIGINAL_URL`. This is useful because, when the principal later authenticates, the original URL and parameters can then be used as a redirect bringing the principal back to their original POST. An example of how this can be accomplished at login is shown below:

```
HttpAuthenticator.authenticate(request, response, username, password);

// Check if principal was requesting a page that was prevented, if so, redirect.
String requestingPage = (String) getSession().getAttribute(VerifyTokenFilter.ORIGINAL_URL);

if (requestingPage != null) {
    // redirect the principal to the requesting page
    response().sendRedirect(requestingPage);
} else {
    // return the to the login page
    return SUCCESS;
}
```

3.4 - SOAP API

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

The SOAP WSDL is available on the following URL with the standalone version:

- <http://localhost:8080/crowd/services/SecurityServer?wsdl>

The Java Remote Interface that is used to generate the SOAP service is available here:

- <http://docs.atlassian.com/crowd/current/com/atlassian/crowd/integration/service/soap/server/SecurityServer.html>

This JavaDoc file details inputs and outputs for the available Crowd security server SOAP server. You will see that all methods require an `AuthenticatedToken`. A valid token can be obtained by calling the `authenticateApplication` service method.

Like a user token, the the client token is valid only for the same period of time a user token would be. If you receive a SOAP fault for an invalid application client you will need to re-authenticate your application client and recall the SOAP service.

authenticateApplication – Authenticating an Application Client

Here is the server request which passes in the server name and a password credential.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soap:Body>
    <authenticateApplication xmlns="urn:SecurityServer">
      <in0>
        <credential
xmlns="http://authentication.integration.crowd.atlassian.com">
          <credential>password</credential>
        </credential>

        <name
xmlns="http://authentication.integration.crowd.atlassian.com">jira</name>

        <validationFactors
xmlns="http://authentication.integration.crowd.atlassian.com" xsi:nil="true" />
      </in0>
    </authenticateApplication>
  </soap:Body>
</soap:Envelope>
```

The server will respond with an application token:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soap:Body>
    <authenticateApplicationResponse xmlns="urn:SecurityServer">
      <out>
        <name
xmlns="http://authentication.integration.crowd.atlassian.com">jira</name>
```

```

                <token
xmlns="http://authentication.integration.crowd.atlassian.com">9vN5haaWY+xGBs3XitgAIg==</token>
            </out>
        </authenticateApplicationResponse>
    </soap:Body>
</soap:Envelope>

```

authenticatePrincipal – Authenticating an Principal

In this message the principal is authenticated using the previously obtained application token.

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soap:Body>
        <authenticatePrincipal xmlns="urn:SecurityServer">
            <in0>
                <name
xmlns="http://authentication.integration.crowd.atlassian.com">jive</name>
                <token
xmlns="http://authentication.integration.crowd.atlassian.com">9vN5haaWY+xGBs3XitgAIg==</token>
            </in0>
            <in1>
                <application
xmlns="http://authentication.integration.crowd.atlassian.com">jive</application>
                <credential
xmlns="http://authentication.integration.crowd.atlassian.com">
                    <credential>password</credential>
                </credential>
                <name
xmlns="http://authentication.integration.crowd.atlassian.com">jstepka</name>
                <validationFactors
xmlns="http://authentication.integration.crowd.atlassian.com" />
            </in1>
        </authenticatePrincipal>
    </soap:Body>
</soap:Envelope>

```

The server then responds back with the token for the now authenticated user:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soap:Body>
        <authenticatePrincipalResponse xmlns="urn:SecurityServer">
            <out>o7MSozJJbKQtOLvC4hN2w==</out>
        </authenticatePrincipalResponse>
    </soap:Body>
</soap:Envelope>

```

In invalid authentication attempt will look like the following:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soap:Body>
        <soap:Fault>
            <faultcode>soap:Server</faultcode>
            <faultstring>Fault:
com.atlassian.crowd.integration.exception.InvalidAuthenticationException</faultstring>
            <detail>
                <InvalidAuthenticationException xmlns="urn:SecurityServer" />
            </detail>
        </soap:Fault>
    </soap:Body>
</soap:Envelope>

```

```

        </soap:Fault>
    </soap:Body>
</soap:Envelope>

```

findPrincipalByToken – Finding a Principal by their Authenticated Token

Now that the principal is authenticated, we may want to find additional details about the principal. With the authenticated principal token, the application can now lookup a user by a token or their name. The example below shows looking up a principal by their authenticated token:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soap:Body>
        <findPrincipalByName xmlns="urn:SecurityServer">
            <in0>
                <name
xmlns="http://authentication.integration.crowd.atlassian.com">jive</name>
                <token
xmlns="http://authentication.integration.crowd.atlassian.com">9vN5haaWY+xGBs3XitgAIg==</token>
            </in0>
            <in1>jstepka</in1>
        </findPrincipalByName>
    </soap:Body>
</soap:Envelope>

```

The server lookup response for the principal token:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soap:Body>
        <findPrincipalByNameResponse xmlns="urn:SecurityServer">
            <out>
                <ID xmlns="http://soap.integration.crowd.atlassian.com">-1</ID>
                <active xmlns="http://soap.integration.crowd.atlassian.com">true</active>
                <attributes xmlns="http://soap.integration.crowd.atlassian.com">
                    <SOAPAttribute>
                        <name>sn</name>
                        <values>
                            <ns1:string xmlns:ns1="urn:SecurityServer">Stepka</ns1:string>
                        </values>
                    </SOAPAttribute>
                    <SOAPAttribute>
                        <name>invalidPasswordAttempts</name>
                        <values>
                            <ns1:string xmlns:ns1="urn:SecurityServer">0</ns1:string>
                        </values>
                    </SOAPAttribute>
                    <SOAPAttribute>
                        <name>requiresPasswordChange</name>
                        <values>
                            <ns1:string xmlns:ns1="urn:SecurityServer">>false</ns1:string>
                        </values>
                    </SOAPAttribute>
                    <SOAPAttribute>
                        <name>mail</name>
                        <values>
                            <ns1:string
xmlns:ns1="urn:SecurityServer">justen.stepka@atlassian.com</ns1:string>
                        </values>
                    </SOAPAttribute>
                    <SOAPAttribute>
                        <name>lastAuthenticated</name>

```

```

        <values>
          <ns1:string
xmlns:ns1="urn:SecurityServer">1169440408520</ns1:string>
        </values>
      </SOAPAttribute>
    <SOAPAttribute>
      <name>givenName</name>
      <values>
        <ns1:string xmlns:ns1="urn:SecurityServer">Justen</ns1:string>
      </values>
    </SOAPAttribute>
    <SOAPAttribute>
      <name>passwordLastChanged</name>
      <values>
        <ns1:string
xmlns:ns1="urn:SecurityServer">1168995491407</ns1:string>
      </values>
    </SOAPAttribute>
  </attributes>
  <conception
xmlns="http://soap.integration.crowd.atlassian.com">2007-01-17T11:58:11+11:00</conception>
  <description xmlns="http://soap.integration.crowd.atlassian.com"
xsi:nil="true"/>
  <directoryID
xmlns="http://soap.integration.crowd.atlassian.com">1</directoryID>
  <lastModified
xmlns="http://soap.integration.crowd.atlassian.com">2007-01-17T18:38:51+11:00
  </lastModified>
  <name xmlns="http://soap.integration.crowd.atlassian.com">jstepka</name>
</out>
</findPrincipalByNameResponse>
</soap:Body>
</soap:Envelope>

```

3.5 - .NET Client

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

You will need to create a .NET proxy to the SOAP API, as follows:

1. Open a Microsoft Visual Studio .NET Command Prompt.

1. Run the following command to generate a proxy class (change the location of the WSDL according to your installation):

```
wsdl /l:CS /protocol:SOAP http://localhost:8080/crowd/services/SecurityServer?wsdl
```

(Note: Ignore any schema validation warnings returned here)

2. Compile the generated class with the following references:

```
csc /t:library /r:System.Web.Services.dll /r:System.Xml.dll SecurityServer.cs
```

This should generate a .NET assembly called `SecurityServer.DLL`.

When creating your .NET client application, remember to add a reference to this proxy. You will also need to add a reference to `System.Web.Services.DLL`.

The [sample code](#) calls methods from the proxy to perform authentication in a sample Crowd Application. Change the constants at the top of the code relevant to any Application you have previously set-up in Crowd.

4. - Connectors

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

- [4.1 - Supported Applications and Directories](#)
- [4.2 - Integrating Crowd with Apache](#)
- [4.3 - Integrating Crowd with Bamboo](#)
- [4.4 - Integrating Crowd with Fisheye](#)
- [4.5 - Integrating Crowd with Confluence](#)
- [4.6 - Integrating Crowd with JIRA](#)
- [4.7 - Integrating Crowd with Jive Forums](#)

4.1 - Supported Applications and Directories

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

Application Connectors

- [Atlassian JIRA](#)
- [Atlassian Confluence](#)
- [Atlassian Bamboo](#)
- [Cenqua Fisheye](#)
- [Jive Forums](#)
- Jive Wildfire

Directory Connectors

- Microsoft Active Directory
- Open LDAP
- Apple Open Directory
- Sun Java System (Sun ONE) Directory Server
- Internal Crowd Directory

4.2 - Integrating Crowd with Apache

This page last changed on Mar 06, 2007 by justen.stepka@atlassian.com.



IN DEVELOPMENT

The attached Perl module is in beta, use the module with caution.

- ([Apache-CrowdAuth-0.03.zip](#)) - Works with Perl 1.99 and 2.

If you find any problems with the module please comment on the following ticket:

- <http://jira.atlassian.com/browse/CWD-97>

Introduction

This documentation describes how to configure Crowd to authenticate HTTP Authentication requests made to an [Apache](#) webserver.

- These instructions assume some Unix system and Apache configuration knowledge.

Prerequisites

- Apache web server version 2.0 or above with the `mod_perl` module installed and configured.
- `SOAP::Lite` perl module (v0.69 or greater recommended).

Installation and Configuration

The following instructions are for Unix systems. If you're running Apache on Windows, see the [notes](#) below.

Installing the `SOAP::Lite` Perl Module

[SOAP::Lite](#) is a Perl library for managing SOAP calls. It is used by the `CrowdAuth` module to talk to the Crowd server.

The easiest way to install `SOAP::Lite` is via [CPAN](#), by running the following command.

```
perl -MCPAN -e 'install SOAP::Lite'
```

Alternatively, you can [download and install the package manually](#).

Installing the Apache::CrowdAuth Perl Module

Download the Apache-CrowdAuth-0.03.tar.gz file and extract and install it as follows:

```
tar xvzf Apache-CrowdAuth-0.03.tar.gz
cd Apache-CrowdAuth-0.03
perl Makefile.PL
make
make install
```

Configuring Apache

Ensure that `mod_perl` is enabled.

Your Apache config file should contain a line like the following:

```
LoadModule perl_module modules/mod_perl.so
```

Many common distributions of Apache come with `mod_perl` preconfigured.

Configure Authentication

To tell Apache to use Crowd to authenticate requests for a particular location, edit the Apache config file to add the following commands to a `<Location>` or `<Directory>` section.

```
Alias /crowd/ "/var/crowd/"
<Directory "/var/crowd/">
.
.
.
    AuthName crowd
    AuthType Basic

    PerlAuthenHandler Apache::CrowdAuth
    PerlSetVar CrowdAppName appname
    PerlSetVar CrowdAppPassword apppassword
    PerlSetVar CrowdSOAPURL http://localhost:8080/crowd/services/SecurityServer

    require valid-user
.
.
.
</Directory>
```

Command	Explanation
AuthName crowd	Defines the realm of the authentication. This information is typically provided to the user in the dialog box popped up by their browser
AuthType Basic	Tells apache to use basic authentication
PerlAuthenHandler Apache::CrowdAuth	Tells Apache to delegate authentication to the CrowdAuth module

PerlSetVar CrowdAppName	Set the Application Apache should authenticate as
PerlSetVar CrowdAppPassword	Set the password for the Application
PerlSetVar CrowdSOAPURL	The URL of the Crowd SOAP service
require valid-user	Tells Apache that clients must provide a valid username/password to access the location

Subversion Integration

If you are using Apache to manage access to a subversion repository ([instructions](#)), you can use the same configuration method to delegate user authentication to Crowd.

Example:

```
<Location /svn>

# Uncomment this to enable the repository,
DAV svn

# Set this to the path to your repository
SVNPath /var/lib/svn

AuthName crowd
AuthType Basic

PerlAuthenHandler Apache::CrowdAuth
PerlSetVar CrowdAppName subversion
PerlSetVar CrowdAppPassword svn
PerlSetVar CrowdSOAPURL http://localhost:8080/crowd/services/SecurityServer

require valid-user

# The following three lines allow anonymous read, but make
# committers authenticate themselves.
<LimitExcept GET PROPFIND OPTIONS REPORT>
Require valid-user
</LimitExcept>

</Location>
```

Note that Apache will have to be restarted before any changes to its config files will take effect.

Troubleshooting

- The CrowdAuth module logs detailed output if the Apache [LogLevel](#) parameter is set to `info` or `debug`. This can be useful in diagnosing problems (WARNING: passwords are logged in plaintext to the Apache log file when `LogLevel` is set to `debug`).

Apache Log Error Messages

CrowdAppName or CrowdAppPassword is not defined	One or both of the CrowdAppName or CrowdAppPassword parameters is missing from the Apache config file
Failed to authenticate application	The attempt to authenticate the application with

	crowd failed. Check the values of the <code>CrowdAppName</code> or <code>CrowdAppPassword</code> parameters
Failed to authenticate principal	Failed to authenticate a username/password pair provided by the client. This may just mean that the username or password supplied is incorrect. Note that <code>CrowdAuth</code> won't log successful authentications unless the <code>LogLevel</code> is <code>info</code> or above.
User token not found in SOAP response for user <code><user></code>	Internal SOAP protocol error
error 500...at <code>CrowdAuth.pm..</code>	Indicates that Apache can't connect to the Crowd SOAP service
error 404...at <code>CrowdAuth.pm...</code>	Indicates that the URL used to connect to the Crowd SOAP service is incorrect. Check the value of the <code>CrowdSOAPURL</code> parameter
failed to resolve handler `Apache::CrowdAuth': Can't locate <code>Apache/CrowdAuth.pm ...</code>	The <code>CrowdAuth.pm</code> file isn't located on the Perl include path (or it is permissioned incorrectly)
failed to resolve handler `Apache::CrowdAuth': Can't locate <code>SOAP/Lite.pm...</code>	The <code>SOAP::Lite</code> module hasn't been installed

Installing Perl, `mod_perl` and Perl Modules on Windows

Setting up `CrowdAuth` on an Apache instance running on Windows requires that some things be done differently.

(The following instructions assume you are using [ActivePerl](#) as your Perl environment).

- If you don't already have a Perl interpreter installed, you'll need one. The following instructions assume an install of [ActiveState's ActivePerl](#).
- Windows installations of Apache are less likely to come with `mod_perl` pre-installed. A Win32 version of `mod_perl` in [PPM](#) format is available [here](#).
- The `.tar.gz` format used to distribute `CrowdAuth` (and other modules) is supported by most modern Windows archiving utilities ([WinZip](#), for example).
- The `make` utility used to build the Perl modules is not part of a Windows. `nmake`, Microsoft's equivalent, is available (as a self-extracting archive) [here](#).

Installing `SOAP::Lite` on Windows

Use the `cpan` shell

```
C:\> cpan
cpan> install SOAP::Lite
```

Installing `Apache::CrowdAuth` on Windows

```
Extract Apache-CrowdAuth-0.03.tar.gz using Winzip or equivalent...  
cd Apache-CrowdAuth-0.03  
perl Makefile.PL  
nmake  
nmake install
```

4.3 - Integrating Crowd with Bamboo

This page last changed on Feb 26, 2007 by rosie@atlassian.com.

Atlassian's [Bamboo integration server](#) can quickly be configured to use the atlassian-user libraries to link in single or multiple directory servers through Crowd.

To configure the atlassian-user framework, perform the following:

1. Copy the Crowd integration libraries and configuration files as described in the [3.2 - Client Configuration](#) documentation.
2. Edit the `\bamboo\webapp\WEB-INF\classes\atlassian-user.xml` file to add the following repository:

```
<repository class="com.atlassian.crowd.integration.atlassianuser.CrowdRepository">
  <classes>
    <processor>com.atlassian.crowd.integration.atlassianuser.CrowdRepositoryProcessor</processor>
    <userManager>com.atlassian.crowd.integration.atlassianuser.CrowdUserManager</userManager>
    <groupManager>com.atlassian.crowd.integration.atlassianuser.CrowdGroupManager</groupManager>
    <authenticator>com.atlassian.crowd.integration.atlassianuser.CrowdAuthenticator</authenticator>
    <propertySetFactory>com.atlassian.crowd.integration.atlassianuser.CrowdPropertySetFactory</propertySetFactory>
    <entityQueryParser>com.atlassian.crowd.integration.atlassianuser.CrowdEntityQueryParser</entityQueryParser>
  </classes>
</repository>
```

You will need to comment out the Hibernate repository key

```
<!-- <hibernate name="Hibernate Repository" key="hibernateRepository" description="Hibernate Repository"/> -->
```

3. This step is only necessary if you wish to enable single sign-on:



Enabling Single Sign-On

Single sign-on (SSO) is optional when integrating Bamboo and other Atlassian products. To use centralised authentication do not configure Seraph based authentication.

Edit the `\bamboo\webapp\WEB-INF\classes\seraph-config.xml`, changing the authenticator node to read:

```
<authenticator class="com.atlassian.crowd.integration.seraph.BambooAuthenticator"/>
```

Bamboo's authentication and access request calls will now be performed using the atlassian-user Crowd plugin.

When utilising the atlassian-user and Crowd framework together with Bamboo, it is highly recommended that caching be enabled. Multiple redundant calls to the atlassian-user framework are made on any given request. These results can be stored locally between calls by enabling caching in the Crowd 'Options' menu. In doing so, Bamboo will obtain all necessary information for the period specified by the cache in minutes. If a security change or addition occurs in Crowd, these changes will not be visible in Confluence until the item cache expires.

Additional configuration steps:

- Create the 'bamboo' application in the Crowd administration console. Make sure that you use the same password as configured in the `crowd.properties` file. More information on adding an application is available here
 - You will need to make sure you add the IP address of the client address, in this case Bamboo's IP address to the list of
- Create a group `bamboo-admin`, through the Crowd console or directly in your directory server.
 - You will need to assign the `bamboo-admin` group to the newly configure 'bamboo' application through the Crowd administration console or authentication attempts will fail.



For more information please refer to the [Bamboo documentation](#).

4.4 - Integrating Crowd with Fisheye

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

Fisheye offers you the ability to specify an implementation to provide authentication and authorisation external to the application.

Copy the Crowd integration libraries and configuration files as described in the [3.2 - Client Configuration](#) documentation.

To configure Fisheye you will need to specify the following Custom Authenticator in the `Users/Security` section of the administration console.

```
com.atlassian.crowd.integration.fisheye.FisheyeAuthenticator
```

You will need to make sure that a group is designed that is allowed to authenticate with Crowd and that all users who will authenticate with Fisheye through Crowd will be members of this group.

It is also important to note that Fisheye requires you to pass in the configuration attributes for Crowd, by specifying your configuration data through the properties editor:

<code>application.name</code>	<code>fisheye</code>
<code>application.password</code>	<code>password</code>
<code>application.login.url</code>	<code>http://localhost:8081/</code>
<code>crowd.server.url</code>	<code>http://localhost:8080/crowd/services/</code>
<code>session.isauthenticated</code>	<code>session.isauthenticated</code>
<code>session.tokenkey</code>	<code>session.tokenkey</code>
<code>session.validationinterval</code>	<code>0</code>
<code>session.lastvalidation</code>	<code>session.lastvalidation</code>



Fisheye Documentation

<http://www.cenqua.com/fisheye/doc/latest/admin/customauth.html>



Migration Caveat

If a username is already configured through the Fisheye administration console, the account type will need to be changed from 'built-in' to 'custom' for the new permissioning through Crowd to work properly.

4.5 - Integrating Crowd with Confluence

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

Atlassian's popular [Confluence wiki](#) can quickly be configured to use the atlassian-user libraries to link in single or multiple directory servers through [Crowd].



Supported Versions

Crowd supports Confluence version 2.2 and later.

To configure the atlassian-user framework, perform the following:

1. Copy the Crowd integration libraries and configuration files as described in the [3.2 - Client Configuration](#) documentation.



Use Confluence Spring JAR

Use the Spring JAR shipped with Confluence. Using the Spring JAR shipped with Crowd will not work with Confluence and prevent the starting of Confluence.

2. Edit the `confluence\WEB-INF\classes\atlassian-user.xml` file to add the following repository:

```
<repository key="crowd" class="com.atlassian.crowd.integration.atlassianuser.CrowdRepository">
  <classes>
    <processor>com.atlassian.crowd.integration.atlassianuser.CrowdRepositoryProcessor</processor>
    <userManager>com.atlassian.crowd.integration.atlassianuser.CrowdUserManager</userManager>
    <groupManager>com.atlassian.crowd.integration.atlassianuser.CrowdGroupManager</groupManager>
    <authenticator>com.atlassian.crowd.integration.atlassianuser.CrowdAuthenticator</authenticator>
    <propertySetFactory>com.atlassian.crowd.integration.atlassianuser.CrowdPropertySetFactory</propertySetFactory>
    <entityQueryParser>com.atlassian.crowd.integration.atlassianuser.CrowdEntityQueryParser</entityQueryParser>
  </classes>
</repository>
```

You will need to comment out the OSUser repository key

```
<!-- <osuser key="osuserRepository" name="OSUser Repository"/> -->
```

You will also need to comment out the Hibernate repository key

```
<!-- <hibernate name="Hibernate Repository" key="hibernateRepository" description="Hibernate Repository" /> -->
```

3. The Confluence file `confluence\WEB-INF\classes\oscache.properties` has a default cache limit of 100 objects. This should be commented out or adjusted to something beyond the total number of users and/or groups in your configuration.

```
oscache.memory=true

# CACHE SIZE
# Default cache size in number of item. If a size is specified but not
# an algorithm, the cache algorithm used will be LRUCache.
```

```
#cache.size=100
#
# CACHE UNLIMITED DISK
# Use unlimited disk cache or not
#cache.unlimited_disk=false
```

4. This step is only necessary if you wish to enable single sign-on:



Enabling Single Sign-On

Single sign-on (SSO) is optional when integrating Confluence and other Atlassian products. To use centralised authentication do not configure Seraph based authentication.

Edit the `\confluence\webapp\WEB-INF\classes\seraph-config.xml`, changing the authenticator node to read:

```
<authenticator class="com.atlassian.crowd.integration.seraph.ConfluenceAuthenticator"/>
```

Confluence's authentication and access request calls will now be performed using the `atlassian-user\Seraph Crowd` plugin.

When utilising the `atlassian-user` and Crowd framework together with Confluence, it is highly recommended that caching be enabled. Multiple redundant calls to the `atlassian-user` framework are made on any given request. These results can be stored locally between calls by enabling caching in the Crowd 'Options' menu. In doing so, Confluence will obtain all necessary information for the period specified by the cache in minutes. If a security change or addition occurs in Crowd, these changes will not be visible in Confluence until the item cache expires.

Additional configuration steps:

- Create the 'confluence' application in the Crowd administration console. Make sure that you use the same password as configured in the `crowd.properties` file. More information on adding an application is available [here](#)
- Create two groups, `confluence-users`, and `confluence-administrators`, through the Crowd console or directly in your directory server.
 - You will need to assign the `confluence-users`, and `confluence-administrators` group to the newly configured 'confluence' application through the Crowd administration console or authentication attempts will fail.
- Confluence's security requires that principals be members of a Confluence group that has the 'Global Permission' Confluence Users.
- In the General Configuration administration section, turn on External user management.



For more information please refer to the [Confluence documentation](#).

4.6 - Integrating Crowd with JIRA

This page last changed on Mar 04, 2007 by rosie@atlassian.com.



Before you begin:

Please make sure you have already setup JIRA before performing the below Crowd integration. Crowd cannot be installed on an unconfigured JIRA instance.

Atlassian's popular [JIRA issue management system](#) takes advantage of the OSUser framework and can quickly be configured to use OSUser to link in single or multiple directory servers through Crowd. Crowd provides integration libraries for the OpenSymphony OSUser module, which has a simple- to-use API for user-management that allows pluggable implementations. More about the OSUser API can be reviewed at <http://www.opensymphony.com/osuser/>.

JIRA and Crowd Compatibility

JIRA Version	Integration Points
< 3.7.3	<ul style="list-style-type: none">• External user management must be enabled.
>3.7.4	<ul style="list-style-type: none">• External user management must be enabled.• User preferences are supported.

It is recommended that all integrations with Crowd upgrade their JIRA instances to 3.7.4+.

Enabling "External user management" in JIRA means that the following functions can no longer be performed from within the JIRA administration interface:

- adding users.
- adding groups.
- editing users.
- editing groups.

Configuring JIRA to work with Crowd

To configure the OSUser framework:

1. Copy the Crowd integration libraries and configuration files as described in the [3.2 - Client Configuration](#) documentation.
2. Edit the JIRA file `atlassian-jira\WEB-INF\classes\osuser.xml` to add the following providers:



You will need to comment out the existing providers and now use the Crowd providers.

```
<opensymphony-user>
  <authenticator class="com.opensymphony.user.authenticator.SmartAuthenticator"/>
  <provider class="com.atlassian.crowd.integration.osuser.CrowdCredentialsProvider"/>
```

```

<provider class="com.atlassian.crowd.integration.osuser.CrowdAccessProvider"/>
<provider class="com.atlassian.crowd.integration.osuser.CrowdProfileProvider"/>
<!--
<provider class="com.atlassian.core.ofbiz.osuser.CoreOFBizCredentialsProvider">
  <property name="exclusive-access">true</property>
</provider>

<provider class="com.opensymphony.user.provider.ofbiz.OFBizProfileProvider">
  <property name="exclusive-access">true</property>
</provider>

<provider class="com.opensymphony.user.provider.ofbiz.OFBizAccessProvider">
  <property name="exclusive-access">true</property>
</provider>
-->
</opensymphony-user>

```

1. Edit the `propertyset.xml` to add the following propertyset at the end of the file as the last propertyset:

```

<propertyset name="crowd" class="com.atlassian.crowd.integration.osuser.CrowdPropertySet"/>

```

2. This step is only necessary if you wish to enable single sign-on:



Enabling Single Sign-On

Single sign-on (SSO) is optional when integrating JIRA and other Atlassian products. To use centralised authentication, do not configure Seraph-based authentication.

Edit the `\atlassian-jira\WEB-INF\classes\seraph-config.xml`, changing the authenticator node to read:

```

<authenticator class="com.atlassian.crowd.integration.seraph.JIRAAuthenticator"/>

```

JIRA's authentication and access request calls will now be performed using the `atlassian-user/osuser/Seraph Crowd` plugin.

Now when authentication or access request calls are performed versus the OSUser framework, the JIRA stack will call the Crowd providers and propertyset implementations.

When utilising the OSUser and Crowd framework together with JIRA, it is highly recommended that caching be enabled. Multiple redundant calls to the OSUser framework are made on any given request. These results can be stored locally between calls by enabling caching in the Crowd 'Options' menu. In doing so, JIRA will obtain all necessary information for the period specified by the cache in minutes. If a security change or addition occurs in Crowd, these changes will not be visible in JIRA until the item cache expires.

Additional configuration steps:

- Create the 'jira' application in the Crowd administration console. Make sure that you use the same password as configured in the `crowd.properties` file. More information on adding an application is available [here](#)
- Create three groups, `jira-users`, `jira-developers` and `jira-administrators`, through the Crowd console or manually in your directory server for each associated directory server.
 - You will need to assign the `jira-users` group to the newly configured 'jira' application through

- the Crowd administration console or authentication attempts will fail.
- When integrating with JIRA, only principals who are members of the `jira-users` group will be able to authenticate.
 - Only principals who are members of the `jira-administrators` group will be able to administer the JIRA console.
 - JIRA's security requires that principals be members of a JIRA group that has the 'Global Permission' JIRA Users.
 - In the JIRA General Configuration administration section, turn on External user management and External password management.



For more information please refer to the [JIRA documentation](#).

4.7 - Integrating Crowd with Jive Forums

This page last changed on Mar 07, 2007 by [shamid](#).

Jive Forums offers you the ability to specify an implementation to provide authentication and authorisation external to the application. This document outlines how to integrate Crowd's authenticator with Jive Forums.

Currently Crowd provides centralised authentication for Jive users. Full SSO with Jive is a soon to be implemented feature ([CWD-195](#)).

Prerequisites

1. Download and configure Crowd. Refer to the [installation guide](#) for detailed information on how to do this. We will refer to the Crowd root folder as `CROWD`.
2. Configure Jive Forums. Refer to the relevant Jive Forums documentation for information regarding this installation process. The documentation is usually supplied with the software distribution. Do not attempt to use Crowd as the authentication system during the installation process (use the default authentication system for the installation process).

Configure Jive Forums WebApp

Jive Forums may be deployed on an application server as a single WAR file or as an exploded WAR folder. For the rest of the installation process, we will assume that Jive Forums has been set up as an exploded war file. If you need Jive Forums to be installed as a single WAR file, simply expand the WAR to a directory, make the changes as described below, and zip up the directory to form the WAR file. We will refer to the root folder of the Jive Forums web-app as `JIVEFORUMS`.

1. Copy the Crowd integration libraries and configuration files (this is described in the [Client Configuration](#) documentation). This is summarised below:

Copy From	Copy To
<code>CROWD/client/*.jar</code>	<code>JIVEFORUMS/WEB-INF/lib</code>
<code>CROWD/client/lib/*.jar</code>	<code>JIVEFORUMS/WEB-INF/lib</code>
<code>CROWD/client/conf/crowd.properties</code>	<code>JIVEFORUMS/WEB-INF/classes</code>

2. Examine the `JIVEFORUMS/WEB-INF/lib` folder and delete any duplicate JARs. Duplicate JARs represent common libraries used by both the Crowd client and Jive Forums. For Jive Forums 5.0.5, the duplicate JARs to delete are:

<code>commons-codec-1.3.jar</code>
<code>commons-collections-3.1.jar</code>
<code>commons-httpclient-3.0.jar</code>
<code>commons-logging-1.0.4.jar</code>

3. Edit `JIVEFORUMS/WEB-INF/classes/crowd.properties`. Change the following properties:

Key	Value
<code>application.name</code>	<code>jiveforums</code>

application.password	set a password
----------------------	----------------

Configure Jive Forums Directory/Users

The Jive Forums application will need to locate users from a directory configured in Crowd. You will need to set up a directory in Crowd for Jive. For more information on how to do this, view the documentation on [Adding a Directory Connector](#). We will assume that the directory is called Jive Forum Directory for the rest of this document. It is possible to assign more than one directory for an application, but for the purposes of this example, we will use Jive Forum Directory to house Jive Forum users.

If you have an existing Jive Forums deployment and would like to import existing users (principals) into Crowd, use the Jive Importer tool by navigating Principals > Import Users > JIVE. Select the Jive Forum Directory as the directory into which Jive Forum users will be imported. The database drivers for the Jive Forums database will need to be on the Crowd's classpath. To do this, simply copy the database driver JAR for your particular Jive database across to `CROWD/apache-tomcat-5.5.20/common/lib` and restart Crowd. Note: the passwords for users in Jive will not be copied across to Crowd as they are stored as hashes in Jive's internal database.

Configure Jive Forums Application

Jive Forums needs to be configured in the Crowd console as an authenticated application. For reference, refer to the generic documentation about [Adding an Application](#).

1. Login to the Crowd Console (by default, this is <http://localhost:8095/crowd/console>) and click navigate to Applications > Add Application.
2. Fill out the form to add the Jive Forums application:

CROWD User: Shihab Hamid Log Out | Help

Home Applications Principals Groups Roles Sessions Directories Options System Info

Add Application

Name: *
The name of the application client; this value will be used for authentication.

Description:
A short description of the application, often a web URL is helpful.

Active: ☒

Password: *

Confirm Password: *

Default Directory: * The default directory the application will use to when performing authentication and authorization checks.

Create » Cancel

Powered by Atlassian Crowd Version 1.0.1 (Build #102: Mar 06, 2007) Report a bug | Request a feature | Contact Atlassian

The name and password values must match those set earlier in the JIVEFORUMS/WEB-INF/classes/crowd.properties.

Configure Authentication Permissions for the Application

Now that Crowd is aware of the Jive Forums application, Crowd needs to know which directories or users can authenticate (log in) via Crowd. You can either configure entire directories to authenticate or allow particular groups. In our example, we can simply allow the entire group to authenticate:

View Application – jiveforumsAdd Application | Remove Application

Details

Directories

Groups

Remote Addresses

Config Test

Directory mappings control which user stores will be used when authenticating and authorizing a principals access request. For a principal to be considered a valid application user, their account must belong to a group that is assigned to the application.

Directory	Allow all to Authenticate	Action
Jive Forums Directory	<input type="checkbox"/> True	Remove

Crowd Standalone Deployment

Add »

Update »

Cancel

In a similar manner we can also allow the application to authenticate groups of users via the Groups tab. See the documentation regarding [Managing an Integrated Application](#) for more information.

Remote Addresses configuration:

- Jive is on a different host to Crowd
If you are running the Jive on a different host to Crowd, you will need to modify the permissible hosts via the Remote Addresses tab. This lists the hosts/IP addresses that are allowed to authenticate to Crowd. If Jive Forums is remote to Crowd, add the IP address of your Jive Forums server and ensure the "Status" field is set to "true". Remove the entry for localhost.
- Jive is on the same host as Crowd
By default, when you add an application, localhost is a permissible foreign host. However, you will also need to manually add the IP address 127.0.0.1, as incoming requests to Crowd from Jive (both on the same, local, host) may be from the host 127.0.0.1 and not localhost. Crowd does not do a DNS lookup of the hostname, rather, it compares the values as is. Ensure the "Status" field is set to "true".

Configure Jive Forums to use Crowd's Authenticator

Crowd is now set up to provide authentication services to Jive - now Jive needs to be set up to use Crowd's authenticator. There are a few ways of doing this, the most user-friendly method is outlined below:

1. In your jiveHome directory, edit a file named jive_startup.xml. Modify the <setup> node to be

false:

```
<jive>
  <!-- When setup is false, you can access the setup tool. -->
  <setup>>false</setup>
  ...
</jive>
```

As the XML comment states, this let's us re-run Jive's setup.

1. Restart Jive Forums so that it picks up the changes.
2. View the Jive Forums site with a web browser (usually under the `/jiveforums` context-root. Jive will run the "Jive Forums Setup".
3. In the Install Checklist screen, click continue to navigate through the setup process.
4. In the Datasource Settings screen, re-enter your database configuration details and click continue.
5. In the User System screen, select Custom authentication system and click continue:

Jive Forums SetupJive Forums

Setup Progress » ● [Install Checklist](#) ● [Datasource Settings](#) ● [User System](#) ● Email Settings ● Admin Account

User, Group and Authentication Systems

Choose a user, group and authentication system below. Most installations should use the default implementation. The other options can be used when you need to integrate Jive Forums with an existing user database or authentication system.

- ☐ **Default** - Use the Jive Forums default user, group and authentication implementations.
- ☐ **LDAP** - Use LDAP for authentication and storing user data.
- ☒ **Custom** - Specify a custom user, group or authentication implementation.

Continue

6. You should be at the Custom User System screen. Enter the following details which specify Crowd as the custom authenticator:

Setup Progress » ● [Install Checklist](#) ● [Datasource Settings](#) ● [User System](#) ● Email Settings ● Admin Account

Custom User System

Enter the classnames of your custom classes below. A valid classname should be something like `com.mycompany.MyUserManager`. Please see the developer's guide and Javadocs for more information about defining your own user manager, group manager, and authentication factory.

UserManager implementation

GroupManager implementation

AuthFactory implementation

[Continue](#)

UserManager implementation:

```
com.atlassian.crowd.integration.jive.CrowdUserManager
```

GroupManager implementation:

Do not specify an implementation.

AuthFactory implementation:

```
com.atlassian.crowd.integration.jive.CrowdAuthFactory
```

Click continue.

If you have any errors at this stage, it is very likely that there is a classpath issue (eg. the Crowd client libraries aren't being properly loaded by Jive). Please read the documentation regarding [INTEGRATING APPLICATIONS \- CROWD CLIENT LIBRARIES](#) for help identifying the problem.

7. In the Email Settings screen, re-enter your email configuration details and click continue.
8. In the Admin Account Setup screen, do not enter any details, click skip this step.



Warning

The default administrator for Jive Forums is the user `admin`. This user will need to exist in your mapped directory to the application configuration for Jive Forums. Without this user, you will not be able to access the administration console of Jive Forums.

9. Bounce the server and test that Crowd is authenticating users for Jive. You could do this by creating users (principals) via the Crowd Console and verifying that they are able to log in to Jive Forums.



Jive Forums Documentation

For further information regarding Jive Forums Authentication Integration, check out the [Jive Forums Documentation](#) at

<http://www.jivesoftware.com/builds/docs/latest/documentation/developer-guide.html#userintegration>

5. - External Resources

This page last changed on Feb 25, 2007 by justen.stepka@atlassian.com.

Navigation

This page last changed on Feb 27, 2007 by justen.stepka@atlassian.com.

[Crowd Home](#)

- [Documentation](#)
- [Blogs](#)

Development

- [Feedback](#)
- [Request a Feature](#)

Blogs

This page last changed on Sep 29, 2006 by justen.stepka@atlassian.com.

Title	Author	Date Posted
-------	--------	-------------