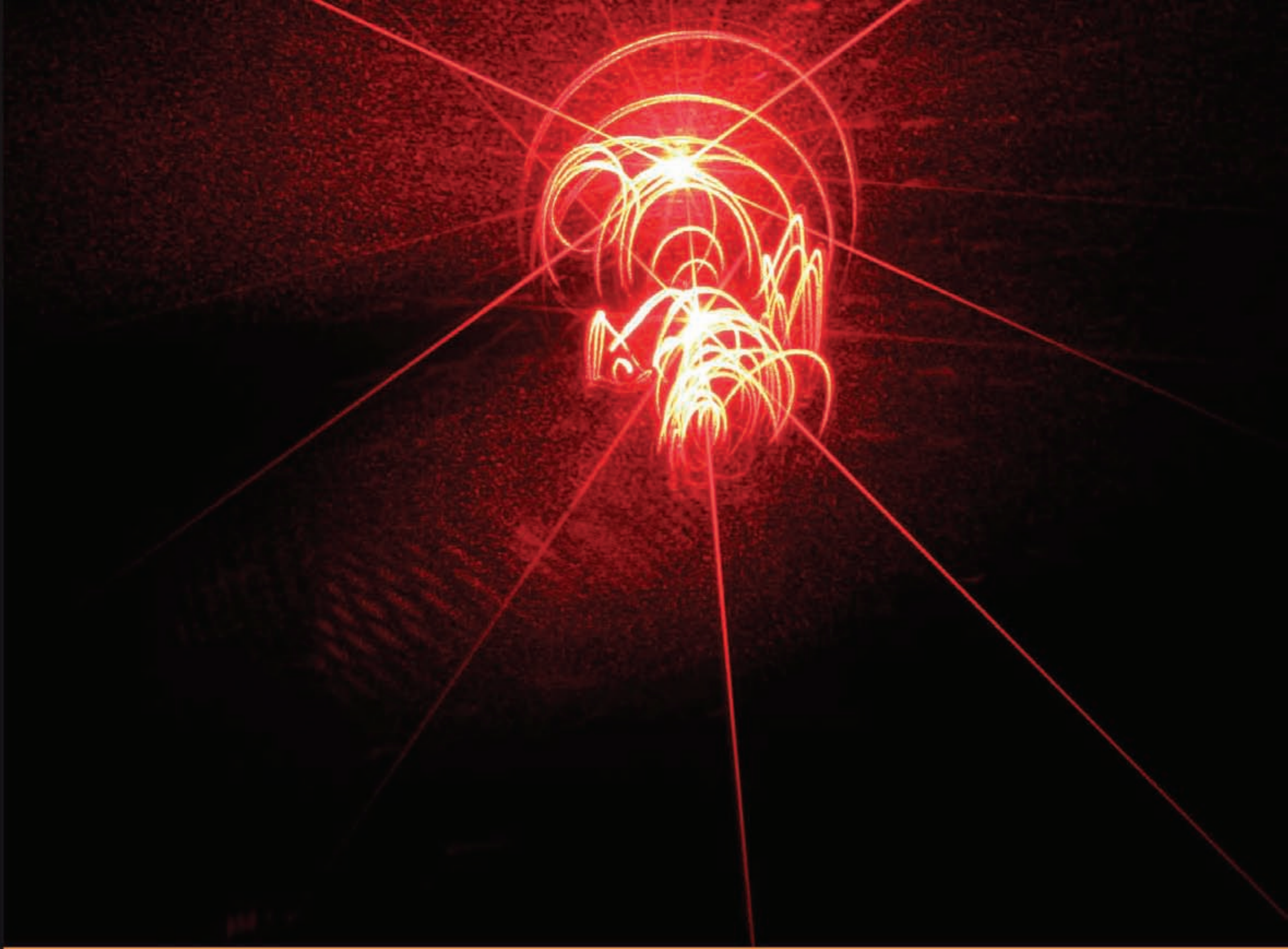


From Technologies to Solutions

Openfire Administration

A practical step-by-step guide to rolling out a secure Instant Messaging service over your network

Mayank Sharma

[PACKT] PUBLISHING

Openfire Administration

A practical step-by-step guide to rolling out a secure Instant Messaging service over your network

Mayank Sharma

PACKT PUBLISHING

BIRMINGHAM - MUMBAI

Openfire Administration

First published: August 2008

Production Reference: 1180808

Published by Packt Publishing Ltd.
32 Lincoln Road
Olton
Birmingham, B27 6PA, UK.

ISBN 978-1-847195-26-5

`www.packtpub.com`

Cover Image by Vinayak Chittar (`vinayak.chittar@gmail.com`)

Credits

Author
Mayank Sharma

Reviewer
Stefan Reuter

Senior Acquisition Editor
David Barnes

Development Editor
Ved Prakash Jha

Technical Editors
Usha Iyer
Gagandeep Singh

Copy Editor
Sumathi Sridhar

Editorial Team Leader
Akshara Aware

Project Manager
Abhijeet Deobhakta

Project Coordinator
Lata Basantani

Indexer
Monica Ajmera

Proofreader
Dirk Manuel

Production Coordinator
Shantanu Zagade

Cover Work
Shantanu Zagade

About the Author

Mayank Sharma is a contributing editor at SourceForge, Inc's Linux.com. He also writes a monthly column for Packt Publishing. Mayank has contributed several technical articles to IBM developerWorks where he hosts a Linux Security blog. When not writing, he occasionally teaches courses on Open Source topics at the Indian Institute of Technology, Delhi, as Industry Expert.

In memory of my grandfather H. C. Sharma and grandmother Shyma Sharma.

Thanks to my mom Shashi and dad Rakesh for laying down tough standards, and younger brother Shashank for his valuable suggestions and technical support.

Gratitude to Ved Prakash Jha for his guidance and advice, to Lata Basantani for managing and scheduling the book, to Usha Iyer and Gagandeep Singh for editing, to Stefan Reuter for his technical insights, and to the wonderful developers and community members of Openfire for developing and supporting such a wonderful project

About the Reviewer

Stefan Reuter is a key contributor to the igniterealtime community, the home of Spark and Openfire. He is also the lead developer of the Asterisk-Java library and the Asterisk-IM plugin for Openfire. After several years in the enterprise architecture group of a major European bank Stefan is now working as an independent consultant on Java and VoIP projects for various international customers.

Table of Contents

Preface 1

Chapter 1: Introduction 7

IM In The Enterprise? 8

But Will IM Work for "Me"? 8

Why Roll Your Own IM Server? 10

What To Look for In An IM Server? 11

Is Openfire The Right IM Server for Me? 12

Summary 15

Chapter 2: Installing and Using Openfire 17

Preparing Your System 17

Linux Users Get Your Cuppa! 18

Meet The Protagonists 18

The Actual Install-Bit 19

Running Openfire on Linux/Unix 20

Installing and Running under Windows 21

Openfire as a Windows Service 22

Fuse Life into Openfire 22

Using The Server 25

Installing The Spark Client 25

Tuning In With Spark 26

Configuring Other IM Clients 28

MirandaIM 28

Kopete 30

Pidgin 31

Summary 32

Chapter 3: Fine-Tuning Openfire 35

Get To Know Your Server 35

Vital Statistics 36

Server Cache and Properties 39

Policing and Tuning The Server **42**
Curb Indiscriminate Registration 42
Resolving Resource Conflicts 44
Handling Offline Users 45
Nurturing The Server 46
Talk to Users 47
Extending with Plugins **48**
Message of The Day 49
Summary **49**

Chapter 4: Organizing Users 51
Adding Users **51**
Editing And Deleting Users 52
Temporarily Suspend Users 53
Organizing Users into Groups **54**
Editing and Deleting Groups 55
Pre-populating Rosters **57**
Scenario #1: See The Members of Your Department 58
Scenario #2: Accounting Reports to The Board 59
Scenario #3: Everyone Wants IT on Their List 60
Scenario #4: Everyone on Everyone's List...Automatically 60
Summary **61**

Chapter 5: Hooking up With a Directory Service 63
Adding Users Via OpenLDAP **64**
Adding Users via Active Directory **69**
Editing the config File **71**
Summary **74**

Chapter 6: Effectively Managing Users 75
Searching for Users with the Search Plugin **76**
Searching for Users from Within The Admin Interface 78
Get Email Alerts via IM **79**
Broadcasting Messages **81**
Setting up the Plugin 82
Using The Plugin 83
Managing User Clients **84**
Private Data Storage **89**
Importing/Exporting Users **90**
Using The Plugin 91
askstatus 94
recvstatus 94
substatus 95
Summary **95**

Chapter 7: Connecting to Other Services 97
Why Connect via Openfire? 98
The Openfire Gateway 98
Controlling Access 100
Scenario #1: Enabling Yahoo! Messenger for All Users 101
Scenario #2: Enabling IRC for IT and Devs 103
Scenario #3: Enabling Experimental Google Talk for Some Users 105
Using A Gateway 106
Keeping An Eye on The Gateway 108
Connecting to VoIP 109
Summary 112
Chapter 8: Playing Big Brother 113
Monitoring and Filtering Content 114
Auditing Messages 116
Setting up a Message Audit Policy 117
Reading Raw Audit Logs 118
Advanced Archiving Settings 120
Searching Archives 122
Light-Weight Third-Party Archiving Apps 124
Using Open Archive 124
Summary 126
Chapter 9: Large-scale Openfire 127
Enabling Database Support in Openfire 127
Setting up MySQL 128
Setting up PostgreSQL 130
Troubleshooting Database Errors 130
Monitoring the Database 131
Scaling Openfire 135
Setting up Clustering 135
Of Caches and Clustering 137
Tips for Optimizing the Cluster 138
Scaling with Connection Managers 139
Deploying Connection Managers 139
Step 1: Configure the Openfire Server 140
Step 2: Configure the Connection Manager 140
Step 3: Run The Connection Manager 141
Summary 142
Chapter 10: Communication Across Multiple Locations 143
Linking Two Branches 144
Why Use DynDNS 144
Setting up Server-to-Server Communications 145

About Remote Networks, Firewalls, and Port Forwarding 146
Checking Connectivity on the Port 147
Adding Users from Remote Network 148
Monitoring Remote Connections 148
Establishing Secure Server-to-Server Communication 149
The Road Warrior 150
File Transfer Across Servers 151
Summary 152

Chapter 11: Running an Online Helpdesk 153
Installing Fastpath and Webchat 153
Creating a Helpdesk 156
Global Settings 157
Workgroup Settings 158
Diversifying Workgroups with Queues 158
Creating Queues 158
Adding Routing Rules 160
Rule Book for Routing Rules 160
Adding Members to Queues 161
Configuring The Helpdesk 162
Canned Responses 163
Offline Settings 164
Chat Transcript 165
Putting the Helpdesk Online 166
Design a Contact Form 167
Review Images and Text 168
Upload the HTML 170
Monitoring Sessions, Reports, and Usage Statistics 172
Snooping Super Users 173
Chat Archives 174
The Agent's Perspective 177
Summary 181

Appendix A: Using Spark 183
Getting Started 183
Chatter Away 186
Spark Plugins 190
Summary 192

Appendix B: Group Chat 193
Group Chat Settings 193
Designing a Room 194
General Room Characteristics 195

User Permissions 196
Occupant Behaviour 197
Putting the Checklist to Test 197
Room 1: A Locked down "boardroom" 198
Room 2: A Free-for-all "Lounge" 199
Room 3: A Restricted "Sales HQ" Room for Sales-Related Discussions 200
Using Spark for Conferencing 200
Impromptu Multi-User Discussions 201
Summary 203

Appendix C: Prepare for Roll Out 205
Identify the Required Objects and their Properties 206
Plan the Deployment Process 207
Instant Messaging – Best Practices 207
User responsibilities 208
Audit Servers and Logs 209
Summary 210

Index 211

Preface

Openfire is a free, open-source and full featured Jabber-based Instant Messaging server.

This book is a functional step-by-step, easy to follow reference guide that explains how to use Openfire can be used to develop a secure instant messenger network. Each chapter will tell you how to add features to your IM.

This book is a guide to setting up Openfire, tweaking it, and customizing it to build a secure and feature-rich alternative to consumer IM networks. The features covered include details about setting up the server, adding and handling users and groups, updating, and extending the service with plug-ins, connecting with users on external IM networks, connecting with external voice over IP solutions and more, with user-friendly instructions and examples so that you can easily set up your IM network.

The book deals with several features of Openfire to streamline communication within an enterprise and beyond. It shows how to configure Openfire to allow only secured connections. It then explains how Openfire complements other existing services running on your network. Managing and fostering IM as a real-time collaboration and communication tool is what this book is about.

What This Book Covers

Chapter 1 introduces you to the importance and benefits of instant messaging over traditional communication methods. It also deals with benefits of hosting one's own EIM server. It then talks about Openfire real-time collaboration server.

In *Chapter 2* you will learn how to install and configure the fully functional Openfire server environment for both Windows and Linux.

After reading *Chapter 3* , you should be well equipped to run and administer an Openfire server in a single-office organization. You'll learn how to manage the administration console and some common network admin tasks.

Chapter 4 deals with the user organization in groups and populating the user's contact list in a variety of ways.

Chapter 5 shows how to sync your directory service with your Openfire server. Based on the directory service information, Openfire will let you migrate the users' contact information and other details, and use it to populate their profiles, as well as create groups.

Chapter 6 explores several ways of effectively administrating users irrespective of the size of the organization. You will learn how to import user data into their roster lists from any XMPP server to Openfire, and use tools that'll help members of the organization broadcast messages to their peers based on preset company rules.

Chapter 7 covers the Gateway plugin that allows access to a variety of IM services including Yahoo! Messenger, IRC, ICQ, MSN Messenger, AOL Instant Messenger, and so on. Apart from just enabling access to these gateways, the plugin lets you configure several aspects of the service including restricting gateway access to all users or a particular group of users. It also covers how you can hook up Openfire to your Asterisk VoIP server.

Chapter 8 talks about uses and ways to archive conversations using iBall plugin. The chapter also deals with the Monitoring Service plugin's archived conversation searching interface. Some freely available archiving options are also discussed including Open Archieve.

Chapter 9 covers in detail the two ways in which a server can be configured to distribute loads—by forming a cluster of Openfire nodes, and by delegating connection managers to make connections to clients. This chapter also covers installing and deploying connection managers in both Windows and Linux environments.

Chapter 10 looks at how users can connect and interact with users on remote Openfire servers. Server-to-server communication is a very powerful feature of Openfire and it enables users who move between multiple Openfire servers to stay connected with each other easily.

In *Chapter 11* we have a look at installing and configuring the plugins and using them to create an online helpdesk with several queues to manage a couple of products. It also talks about online chat facility using the Fastpath plugin. The last section of the chapter illustrates what a Fastpath session feels like to a user and the various tools (transferring calls, user history, personal canned responses, and so on) at his or her disposal to enable him or her to service the users better.

Appendix A offers a quick look at Spark (its features, tools, and plugins), the official Openfire client. This appendix takes a look at the pre-installed and available plugins that give Spark the ability to spell check messages, control indiscriminate file transfers, and translate messages into a variety of languages among other features while bouncing it between users.

We have explored comprehensive group chat mechanisms of Openfire in *Appendix B*. We have broken down Openfire's group chat options into three broad subheads—general room characteristics, user permissions, and occupant behavior. We have looked at each of these in detail, and tried to understand their individual functions and then put them to test. We have then explored Openfire's flexibility with impromptu multiuser discussions.

In *Appendix C*, we've broadly looked at what goes on behind the scenes in selecting an IM server, planning the deployment, and making sure the server maintains a 100% uptime. We've also looked at the various decisions you'll have to make as the admin, and brainstorm with the management and other departments to maximize productivity and add flexibility to the system

Who is This Book For

This book is for System Administrators who want to set up an in-house enterprise IM system using Openfire.

The reader will need experience in managing servers on any operating system.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text are shown as follows: "This file will be under the `webchat` folder under your application server."

Any command-line input and output is written as follows:

```
type openfire_mysql.sql | mysql openfire;
```

New terms and **important words** are introduced in a bold-type font. Words that you see on the screen, in menus or dialog boxes for example, appear in our text like this: "To set some gateway-specific settings, click the **Options** slider button".

Important notes appear in a box like this.

Tips and tricks appear like this.

Reader Feedback

Feedback from our readers is always welcome. Let us know what you think about this book, what you liked or may have disliked. Reader feedback is important for us to develop titles that you can get the most out of.

To send us general feedback, simply drop an email to `feedback@packtpub.com`, making sure to mention the book title in the subject of your message.

If there is a book that you need and would like to see us publish, please send us a note in the **SUGGEST A TITLE** form on `www.packtpub.com` or email `suggest@packtpub.com`.

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, see our author guide on `www.packtpub.com/authors`.

Customer Support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Errata

Although we have taken every care to ensure the accuracy of our contents, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in text or code—we would be grateful if you would report this to us. By doing this you can save other readers from frustration, and help to improve subsequent versions of this book. If you find any errata, report them by visiting `http://www.packtpub.com/support`, selecting your book, clicking on the **let us know** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata added to the list of existing errata. The existing errata can be viewed by selecting your title from `http://www.packtpub.com/support`.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works in any form on the Internet, please provide the location address or website name immediately so we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at `questions@packtpub.com` if you are having a problem with some aspect of the book, and we will do our best to address it.

1
Introduction

Like it or not, instant messaging in the enterprise is here to stay. But rather than being on the back foot, it's time for all corporations, both big and small, to come forth and embrace this technology. Similar to how you look at half a glass of water—half-empty or half-full—Instant Messaging, or IM for short, can be perceived to have a positive or negative impact on productivity depending on the way you look at it. When you wake up to the importance of IM, you'll notice that, if properly managed, IM can increase connectivity within the realm of your business and have a positive impact on productivity.

This book is about managing and fostering IM as a real-time collaboration and communication tool. It's not about the 'why', although why IM is important, but rather it is about the 'what' and 'how'—what IM offers and how you can use it to your advantage.

Most people who use the Internet have been exposed to IM. Like email, IM is a user-centric technology. It offers something you want to use by addressing a basic human need—the need to communicate. There are dozens of public IM services: companies ranging from the leading software developer, Microsoft, to the leading web application developers, Yahoo and Google, offer free IM services. To make sure you use them, they also develop IM clients that work across platforms—from Microsoft Windows, Apple MacOSX, and Linux, to handheld devices like your mobile phone and PDA.

While you may be using IM to discuss weekend plans to pick up your aunt from across town, or kill time discussing the latest movie with a bunch of friends, IM also holds the potential to eliminate conversation blues in your workplace. Ever thought your boss was inaccessible? Wondered how to communicate your ideas to all of the members of your team without wasting time organizing a group meeting? IM is the key to all of these problems. In this chapter, we will discuss:

- The problems with using IM in an enterprise
- The advantages of IM over email and telephone

- The benefits of hosting your own Enterprise Instant Messaging (EIM) server instead of using one of the free public ones
- Some of the features to look for in a EIM solution
- Openfire's features

IM In The Enterprise?

Now that companies are beginning to acknowledge the issue of employees using IM during company hours, they face a difficult choice. Blocking IM also stops employees from using it as a means to communicate with clients or other employees. Thankfully, several organizations aren't opting for this "easy" way out. In my personal experience, more and more companies are trying not to curb the proliferation of IM, but are rather taking steps to manage its use—looking for ways to oversee and control IM.

Let's take a short history lesson so that you will know how it all began and will be able to get things in perspective. When enterprises woke up to the benefits of IM, they also felt the need to control its use, and they ran into a void. IM was a public service. There wasn't any business-grade IM software that would provide the security and legal compliance expected from enterprise software. This void was filled in 1998, when IBM launched the first "Enterprise Instant Messaging" (EIM) software, called IBM Lotus Sametime. Microsoft quickly followed suit, first in haste with Microsoft Exchange Instant Messaging, and more properly later with Microsoft Office Live Communications Server.

Today, EIM is a multi-billion dollar business.

But Will IM Work for "Me"?

Like any Internet user I use IM daily. Like most, I use it to stay in touch with my friends, but every day for the past four years I've also been using IM to communicate with my editors over at Linux.com. Even this book was vetted out with Packt editors over IM.

While you might be impressed by the IDC figures and how IM works for "me", neither means anything if IM doesn't work for "you". IM is about communication—instant, real-time, communication. But of course, IM isn't the only real-time collaboration tool available. In addition to the telephone, the other digital communication tool you have access to is email.

I will not start a flame war here on which tool is the best, but rather will tell you how it ends. When the dust settles on the 'IM versus email versus telephone' battle and the purists retreat, there is only one option—to use all three tools together. Despite the fact that each tool has advantages over the other, because this book is about IM, let me just tell you what IM can do for you that email and telephone can't; at least not with the same efficiency as IM:

- IM is the coyote of communication: There's no faster means of communication than IM. It's called "instant messaging" for a reason. Your messages are delivered instantaneously. Also bear in mind that an IM can carry pictures, documents, or anything else that you could have attached to an email, with the same ease as a plain text message.
- True interactivity with typing notifications: With IM, you don't have to wait for your email to be read and replied to. As a journalist for Linux.com, speed is very important to me. When working on a breaking news story, neither my editors nor myself can afford to bounce emails to each other, sorting clarifications or edits. Also, IM guarantees that your message has been read and will be replied to. Emails do not come with such a guarantee. No other form of digital communication offers the chance to communicate in real-time. You can mark emails as important, and flag them for a response, but if you need prompt action, there's no faster or more reliable way of communicating than with an IM.
- Less chance of misunderstanding: Because IM conversations are instantaneous, the chance of being misunderstood is pretty low. With emails, misunderstandings can linger for a while—at least until someone gets a chance to explain. Because IM follows the natural flow of a conversation, misunderstandings are quickly cleared up.
- Better for brainstorming ideas: Because physical team meetings don't happen at the drop of a hat, managers often resort to email for their brainstorming sessions. But such emails suffer from a severe infection which is the biggest known reason for reduced productivity—CC'itis. Also, long threads of email, with multiple recipients, are difficult to follow and manage.
- A true extension to your phone book: Email has an address book that lets you manage your contacts, but an IM contact list goes a step further by binding your list of contacts with the power of presence. IM uses a technology called "presence awareness" to detect who is online, so that can see at a glance whether the people with whom you want to communicate are online. Now, can your phone do that?

- Knock-knock notification: IM's biggest benefit that'll affect you more than any other feature is its unobtrusiveness. What helps you stay on top of things is a little feature called notifications. These are alerts that let you know when a contact wants to communicate with you, and notify you when someone you'd like to chat with comes online.
- Cheap: Several of the features mentioned above can be accomplished, to some degree, with a telephone. The telephone offers true interactivity, and is ideal for group meetings. But you surely run the risk of catching someone at a "bad time" or of using an outdated contact list. The biggest concern for most companies, however, is cost. If I were using traditional methods of communication, I'd be bankrupt—there are several thousand miles and a couple of time zones between me and my editors in the UK and US. IM helps in keeping your communication—costs down, irrespective of whether you work in a one-office home-office setup, or a multinational corporation.

Why Roll Your Own IM Server?

As I mentioned previously, the only way enterprises could safely allow IM proliferation was if they could apply necessary control. Some companies decided to apply "soft" control over public IM, by restricting, limiting, and pre-screening access. Others opted for more concrete steps, looking for a greater degree of control and, more importantly, privacy. These were the first users of EIMs.

By bringing the EIM infrastructure in-house, with an EIM server, a business can truly manage IM sessions, completely eradicating concerns about security and privacy. As with any good server application, an EIM system is also designed to function like any other enterprise application, offering centralized management, and directory and user integration with corporate directory systems such as Lightweight Directory Access Protocol (LDAP).

Because an in-house EIM server is tailor-tweaked to fit an organization, it does offer some level of customization not found with public IM. This allows a business to integrate IM and its features with other enterprise applications, like corporate email, intranet portals, ERP, and Supply Chain Management solutions and services such as over-the-network telephony.

There's no denying the advantage of an in-house EIM solution, if you want to effectively deploy, regulate, and be in charge of this new and useful means of communication.

What To Look for In An IM Server?

There's no dearth of IM systems available. As we'd like to keep things under control and have decided to roll our own IM service, the next obvious question is: what features should we look for in an EIM solution? Not all products are the same, but like email, there are some features common to all and without which you wouldn't call them an IM system.

In this section, we're not discussing those features that are common and obvious; we're identifying the features that separate a good IM solution from an average one. Understanding these features will help us select an ideal, cost-effective solution that not only delivers now, but also grows as the company around it grows:

- Authentication: Checking the credentials of the users is the foremost task of a server of any kind. A good IM server should make the task of managing users fairly simple. Rather than insisting on managing users themselves, an IM server should be capable of interfacing with third-party authentication systems, such as Directory Server. This also keeps things simple for your employees who won't have to maintain more than one username and password to access multiple services.
- Security: Like all systems in the enterprise, ensuring security is a prime concern. In the case of IM, security becomes all the more important because of the nature of communication. You need an IM system that takes security for messages pretty seriously as they fly across the network,. Some commonly-used security features include secure sign-on, digital signatures, and good ol' encryption.
- Protection against infection: This feature of an IM system flows from another feature—interoperability. Because a good IM system wouldn't discriminate against users of a particular operating system, it has to make sure it doesn't transmit virus-infected files between users. Having your IM system use a third-party anti-virus product for scanning files before transmission is a good idea.
- Logging: Regulations in some sectors mandate keeping logs of all communication, including IM. Even if it doesn't, monitoring conversations or keeping logs isn't a bad idea. Not only does logging prevent users from misusing the system but some IM systems also have features such as on-the-fly keyword flagging that will alert the appropriate person in case of misuse. If you are required to keep logs, then make sure that the system keeps them in a format that's easily accessible.

- Extensibility: The features I've listed above are more or less what you *need* an IM system to have. Some products are more feature-rich than others. You need a system that offers you the basic set of features needed to get started and then offers exotic ones as extensions or plugins. If you use other services, you should also look for a system that plugs into those services if applicable.
- Administration: Adding an IM server to the mix of existing network services increases the administration load. You have to make sure that the IM system doesn't get in your way too much, is easy to manage, and can run on its own, once it's configured and operational.
- Not too demanding system requirements: Finally, you have to weigh in what the IM system brings to the table versus what it requires from you. Like most server software, an IM server in itself doesn't require much. But as the number of users using the service increase, the service exerts more pressure on the physical hardware supporting it. Additionally, IM is an always-on service; therefore, you are looking at a machine that can handle the load of possibly hundreds of simultaneous users generating megabytes of logs.

Is Openfire The Right IM Server for Me?

It's been a long time since the days when IBM and Microsoft dominated the EIM market. Now, there are a range of EIM platforms in addition to IBM Lotus Sametime and Microsoft Office Live Communications Server. Some of the most popular ones are ejabberd, jabberd2, Tigase, and Jabber XCP. According to Wikipedia, there are about 90 million users using Jabber-based servers. Openfire is one such server.

Note: The protocol on which Openfire is based on is now formally called XMPP or Extensible Messaging and Presence Protocol. It was formally known as Jabber, but some people confuse this with the company of the same name (Jabber Inc.). In this book, both terms are used interchangeably to refer to the protocol.

But what's so special about Openfire? There are many Jabber-based EIM servers available on the market. Openfire, which is written in Java, implements most features of XMP, according to Jabber's own website. As a product, Openfire is cross-platform, and is also very easy to setup and administer.

Openfire has lots of features designed to streamline communication within an enterprise. Some of Openfire's features, such as its secure design, are due to its well respected Jabber protocol. Jabber uses Transport Layer Security (TLS) by default, and will establish a secure connection if one is available. Openfire can be configured to allow only secured connections.

While the basic version of the server is available free of cost, an enterprise version, which can be bought for a fee, has features suitable for a large multi-office corporation. Openfire will suit a wide range of enterprises, from home office set-ups to large multi-site enterprises, as it is dual-licensed under GPL along with a commercial extension.

As we'll see in the course of this book, the free GPL'd version is no dumb cousin and has a variety of features that you'd need in an IM server, such as centralized administration of user lists, the ability to broadcast messages to entire groups, and customizable presence states, and tops it all off with a secure feature-rich client. Openfire is designed to complement other existing services running on your network. So, for example, it can plug into a directory server for authenticating users, or into an Asterisk setup for telephony.

Here's a list of some of Openfire's features:

1. **Standards compliant.**
2. **Easy to install.**
3. **Works with multiple external databases in addition to a built-in one:** Openfire can work with several databases including MySQL, PostgreSQL, Oracle, MS SQLServer, and IBM DB2. Its own embedded database is powered by HSQLDB.
4. **Can interface with OpenLDAP or Active Directory.**
5. **Cross-linked, easy to navigate web-based front-end:** Irrespective of what setting you are trying to tweak, the interface provides you with lots of information and offers you various options to choose from. For example, if you want to change how Openfire handles offline messages, the server offers you various permutations to store, bounce, or drop messages. The interface is also cross-linked, which allows you quicker access to relevant portions of the configuration. For example, in the "Server Information" section, along with a list of ports in use, Openfire has a link to the "Security Settings" section from where you can edit the security settings of the ports.
6. **Easy to create and manage user groups:** These user groups can be shared to easily pre-populate new users' contact lists with the right people.
7. **Custom audit policy:** Openfire can audit IM traffic on the server and save the data to XML files. Audit policy settings allow control over how auditing occurs.
8. **Group chat and room administration:** You can easily create and manage chat rooms. Options allow control over room moderation, maximum occupancy, presence information, and more. The group chat room summary page allows you to view and edit current chat rooms and create new ones.

9. **Act as a gateway to other public networks:** Gateway settings allow you to authorize individual client applications so that only clients that have been audited for proper security are allowed on your network.
10. **Lots of security options:** Security settings allow you to control who your users can and can't talk to. Client control allows you to determine which features are enabled in users' IM clients, such as enabling file transfer, message broadcasting, or group chat.
11. **Has a secure client:** The developers of Openfire also make available a free and open-source client called Spark. Spark is written in Java and is designed to make full use of Openfire's security features.
12. **Extend with plugins:** A host of plugins are available for functionality such as importing and exporting data and exposing presence data as a web service. Plugins can be fully administered from inside the Openfire administration console. Even the Spark client can learn new tricks with plugins.
13. **Advanced Reporting tools:** Openfire has advanced reporting tools, which include statistics on active users, conversations, group chat rooms, packet counts and more. With the enterprise edition, you can generate reports for preset time frames or enter specific dates to narrow results. Openfire reports can also be exported as a PDF file.
14. **Client control:** Openfire Enterprise lets you control the features that are enabled in users' IM clients (for Spark and other clients), such as enabling file transfer, message broadcasting, or group chat. You can also control the version of Spark deployed by users from inside the administration console.
15. **Distribute Loads:** When deployed in a large enterprise, Openfire has a couple of tricks up its sleeve to distribute and manage loads with other Openfire servers.
16. **Run an online helpdesk:** With Openfire's Fastpath service you can establish a communication link with users outside your network. This can be used for something as simple as communicating with visitors to your website or as comprehensive as an online helpdesk.

This is just a brief round-up of Openfire's features to get you excited about the book. Again the book is not written to show off Openfire. Instead it's designed to help you setup a usable instant messaging workhorse of a server to kick out any communication blues and enhance the productivity of anyone using the system.

Summary

There's no denying the importance of instant messaging in the modern office. The question in front of the management is not whether to use IM or to block it, but rather how to control it. In this chapter, we've looked at some of the benefits of using IM over "traditional" communication methods such as the telephone and email. We've also discussed the benefits of hosting our own EIM server instead of using one of the free public ones.

After analyzing some of the features to look for in a EIM solution, we've decided to use the Openfire real-time collaboration server. The server is dual-licensed under the open source GPL and a commercial license. It supports instant messaging, group chat and VoIP and uses the only widely adopted open protocol for RTC, XMPP (also called Jabber). Openfire is incredibly easy to setup and administer, but offers rock-solid security and performance.

In subsequent chapters of this book, we'll setup Openfire, tweak it, and customize it until we have a more secure and feature-rich alternative to other consumer IM networks.

2
Installing and Using Openfire

So, I've teased you with the advantages and usefulness of Openfire. You've popped in the red pill. Now let's begin our climb down the rabbit hole.

The Openfire instant messaging server is very easy to install. In fact, it's totally newbie-proof. So much so, that unlike other complex server software, even if you've never setup up Openfire before, you'll be able to get it up and running on your first try. If you're sceptical, by the time we are done with this short chapter, we'll have ourselves a fully-functional Openfire server that'll register users and connect with clients.

In this chapter, we will cover:

- Pre-requisites for Openfire installation
- Installing and running Openfire on Linux/Unix
- Installing and running Openfire on Windows
- Installing Instant Messaging clients

Preparing Your System

Openfire is a cross-platform server and can be installed under Linux, Solaris, Mac, or Windows operating system environments. Openfire reserves its enormity for its users. When it comes to system requirements, Openfire is very suave and a perfect gentleman who has very moderate demands.

You don't need to spend much time preparing your system for installing Openfire. Just pick out the environment you're comfortable with—Windows or one of the popular Linux distributions such as Fedora, Debian, or Ubuntu, and you're good to go. You don't have to run around getting obscure libraries or worry about mismatched versions.

But like any hard-working gentleman, Openfire has a thing for caffeine, so make sure you have Java on your system. No need to run to the kitchen—this isn't the Java in the cupboard. Openfire is written in the Java programming language, so it'll need a Java Runtime Environment (JRE) installed on your system. A JRE creates a simple (breathable, so to say) environment for Java applications to live and function in. It's available as a free download and is very easy to install.

[If you're installing under Windows, just skip to the "Installing Under Windows" section later in the chapter.]

Linux Users Get Your Cuppa!

Sun's Java Runtime Environment is available as a free download from Sun's website (`http://www.java.com/en/download/linux_manual.jsp`) or it can also be installed from your distribution's software management repositories. Users of RPM-based systems can safely skip this section because the Openfire installer for their distribution already includes a JRE.

On the other hand, users of Debian-based systems such as Ubuntu will have to install the JRE before installing Openfire. Thanks to the popular `apt-get` package management system, there isn't much to installing the JRE.

[Because Sun's JRE isn't free and is also not an open source software, most Linux distributions make the JRE package available in their non-free tree. If the following command doesn't work, check out the detailed installation instructions for your specific distribution, at `https://jdk-distros.dev.java.net`.]

Open a console and issue the following command:

```
$ sudo apt-get install sun-java6-jre
```

Now the `apt-get` system will automatically fetch, install, and activate the JRE for you!

Meet The Protagonists

This chapter is about making sure that you have no trouble installing one file. This one file is the Openfire installer and it is available in multiple flavors. The four flavors we're concerned with aren't as exotic as Baskin Robbins' 31 flavors but that doesn't make the decision any easier.

The Openfire project releases several installers. The four flavors we're concerned with are:

- Openfire-3.5.2-1.i386.rpm: RPM package for Fedora Linux and other RPM-based variants
- Openfire_3.5.2_all.deb: DEB package for Debian, Ubuntu Linux and their derivates
- Openfire_3_5_2.tar.gz: Compressed "tarball" archive that'll work on any Linux distribution
- Openfire_3_5_2.exe: Openfire installer for Windows

Openfire 3.5.2 is the latest version available at the time of writing this chapter. There will quite likely be a newer version available when you visit the website after/while reading this chapter.

We'll cover installing Openfire from all of these files, so that you may use Openfire from your favorite Linux distribution or from within Windows. Just to reiterate here, the Windows installer and the RPM Linux installer both bundle the JRE, while the other other versions do not.

The Actual Install-Bit

Alright, so you have the Java JRE setup and you've downloaded the Openfire installer. In this section, we'll install Openfire server from the various versions we discussed in the last section.

Let's first install from the source tarball.

The first step when dealing with `.tar.gz` source archive is to extract the files. Let's extract ours under `/tmp` and then move the extracted directory under `/opt`.

```
# tar zxvf openfire_3_5_2.tar.gz
# mv openfire /opt
```

Now we'll create a non-priviledged user and group for running Openfire.

```
# groupadd openfire
# useradd -d /opt/openfire -g openfire openfire
```

Next, we'll change ownership of the `openfire/`directory to the newly-created user and group.

```
# chown -R openfire:openfire /opt/openfire
```

Believe it or not, that's it! You've just installed Openfire server. Surprised? Get ready for more. It gets even simpler if you install using the precompiled RPM or DEB binaries. In the case of RPM, Openfire is installed under `/opt/openfire` and in case of the DEB file, Openfire resides under `/etc/openfire`.

On RPM-based systems such as Fedora and its derivates (as root), use:

```
# rpm -ivh openfire-3.5.2-1.i386.rpm
```

On DEB-based systems such as Debian, Ubuntu, and so on, use:

```
$ sudo dpkg -i openfire_3.5.2_all.deb
```

Voila! You're done. Now, who thought my "installing Openfire is totally newbie-proof" comment was an exaggeration?

Running Openfire on Linux/Unix

So, we now have Openfire on our favourite Linux distribution, whichever distribution this may be. Now it's time to fire it up and get going. Depending on how you installed Openfire, the procedure to start it varies a little.

If you've installed Openfire from the RPM or DEB, you'll be pleased to know that the Openfire developers have already done most of the hard work for you. These binaries contain some custom handling for the RedHat/Debian-like environments.

You can start and stop Openfire just like any other service on your system:

```
# /etc/init.d/openfire start
Starting Openfire:
```

You can also view the other options available:

```
# /etc/init.d/openfire
Usage /etc/init.d/Openfire {start|stop|restart|status|condrestart|reload}
```

On the other hand, if you've installed Openfire using the `.tar.gz` archive, you can start and stop Openfire using the `bin/openfire` script in your Openfire installation directory. First, change to the user that owns the `/opt/openfire` directory:

```
# su - openfire
# cd /opt/openfire/bin/
# ./openfire start
Starting Openfire
```

And now you have Openfire up and running!

> If you are using a firewall, which you most probably are, make sure to forward traffic on ports 5222 and 5223 (for SSL) which clients use for connecting with the Openfire server. Also forward traffic on port 7777 for file transfer.
>
> Linux users can skip the next section on installing Openfire under Windows and move directly to the section that discusses the preliminary Openfire setup.

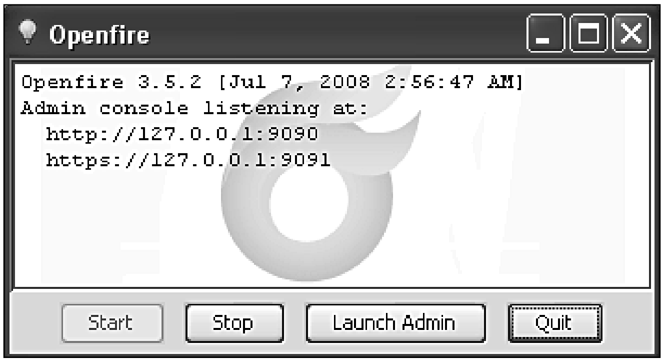
Installing and Running under Windows

Installing Openfire under Windows isn't all that different from installing under Linux. One big difference is that the Openfire installer for Windows includes a JRE. This saves you the trouble of obtaining and installing one yourself, but adds to the size of the Openfire installer.

The process is pretty simple. Download the Windows Openfire installer to any convenient location and double-click it. This launches the Openfire installer wizard which first displays a license agreement and, upon agreeing to their terms, asks standard Windows installation questions, such as the directory you want to install Openfire under, and its start menu entry.

The installer creates the directory, copies the files, and you're done! You have just successfully installed Openfire server.

Once you have installed Openfire, a shortcut to a graphical launcher will be added under the Start Menu, as for any other Windows application. By default, it's accessible via **Start** | **Programs** | **Openfire** | **Openfire Server**.



```
Openfire 3.5.2 [Jul 7, 2008 2:56:47 AM]
Admin console listening at:
  http://127.0.0.1:9090
  https://127.0.0.1:9091
```

When you start the launcher for the first time and click on the **Start** button (refer to the previous screenshot), Openfire will display the message **Admin console listening at http://127.0.0.1:9090**. Use the **Launch Admin** button to start the web interface. It will automatically open your web browser to the correct URL to finish setting up the server.

Openfire as a Windows Service

If you're not a big fan of graphical interfaces, you can also control Openfire from the Windows command-line as a standard Windows service.

You'll find an `Openfire-service.exe` file in the '`bin`' directory of the installation. You can use this executable file to install and control the Openfire service.

From a console window, you can run the following commands:

* To install the service:

```
Openfire-service /install
```

* To uninstall the service:

```
Openfire-service /uninstall
```

* To start the service after installation:

```
Openfire-service /start
```

* To stop the service:

```
Openfire-service /stop
```

You can also use the Services tool in the Windows Control Panel to start and stop the service.

> If you install the Openfire service, you should use that to start and stop Openfire instead of using the graphical launcher under your start menu.

Fuse Life into Openfire

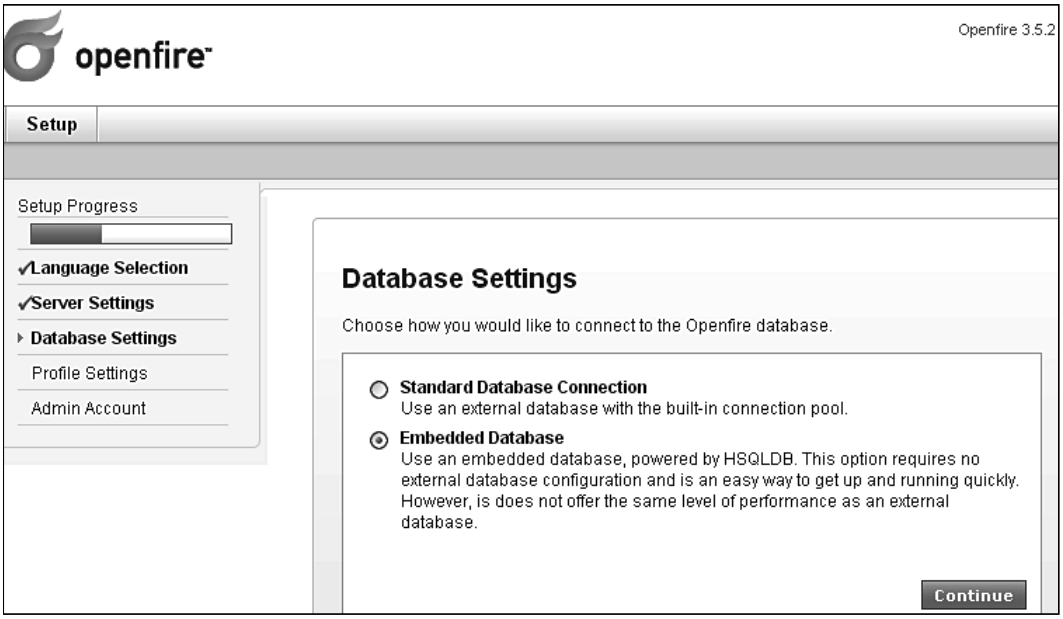
Now we're on the final leg of our journey. This last step is common to both Windows and Linux environments. We've installed Openfire and it's up and running. Now we need to fuse life into it, which is a colorful way of saying that we need to tweak certain settings and point it to our network.

We must tweak these settings from Openfire's Web interface, which runs on port 9090. To access it, launch your browser, and in the address bar enter your IP address followed by a colon and 9090. For example, if the Openfire server is running on a machine with the IP address 192.168.2.5, then the server interface is at `http://192.168.2.5:9090`. If you are on the same machine on which you've installed Openfire, you can also use `http://localhost:9090` or `http://127.0.0.1:9090`. Windows users can also launch the web browser using the Launch Admin option from the graphical launcher in their Start Menu.

Linux users can check their IP address by running the `ifconfig` command

The first time you launch the interface, it'll take you through a brief five-step setup process asking questions about the default language and configuring some server settings. It'll also ask you about the database connection to use. Openfire can work with several databases, including MySQL, PostgreSQL, Oracle, Microsoft SQL Server, and IBM DB2. You can also use Openfire's embedded database, which is powered by HSQLDB. For this last option choose the **Embedded Database** (refer to the following screenshot) option when prompted.

We'll cover Openfire using an external database later in this book.



Database Settings

Choose how you would like to connect to the Openfire database.

○ **Standard Database Connection**
Use an external database with the built-in connection pool.

◉ **Embedded Database**
Use an embedded database, powered by HSQLDB. This option requires no external database configuration and is an easy way to get up and running quickly. However, is does not offer the same level of performance as an external database.

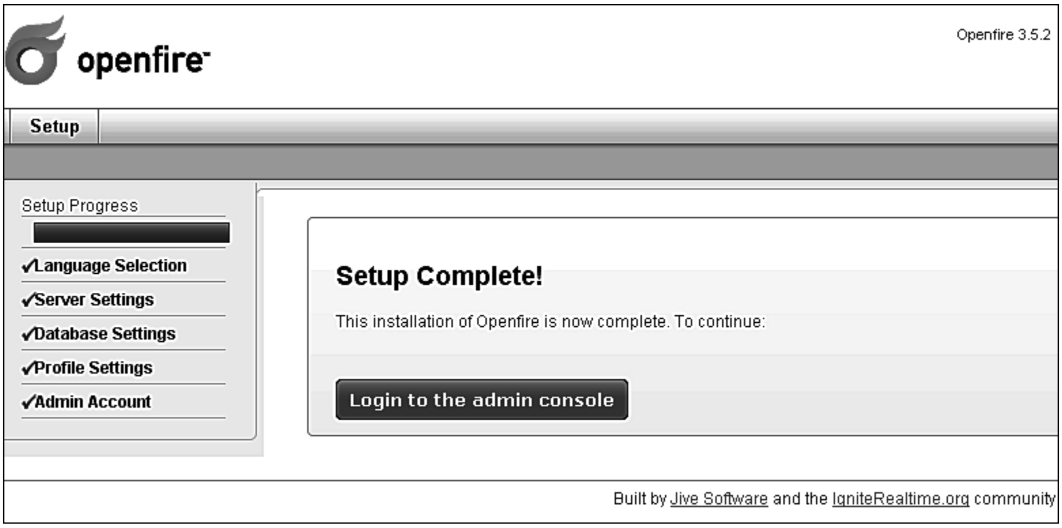
One of the things for which you need the database is storing user information for authentication. But if you have a directory server running on your network, Openfire can obtain authentication information from either OpenLDAP or Active Directory. As we haven't setup a directory server yet, for the moment we'll entrust user management to Openfire. When we're asked to select a user and group system to use, we'll select the Default Option to store users and groups in the embedded server database.

Hooking up Openfire with a directory service will be handled later in the book.

The network settings of the Openfire server involve picking up a domain name for the server and altering the admin console secure and non-secure ports. The interface already includes default values for all of these fields based on the network settings of the machine. If you're not sure what they mean, it is a good idea to leave them to their default values.

If you plan to connect users from two Openfire servers in multiple locations with server-to-server communication, as explained in Chapter 10, please choose a DNS-resolvable name.

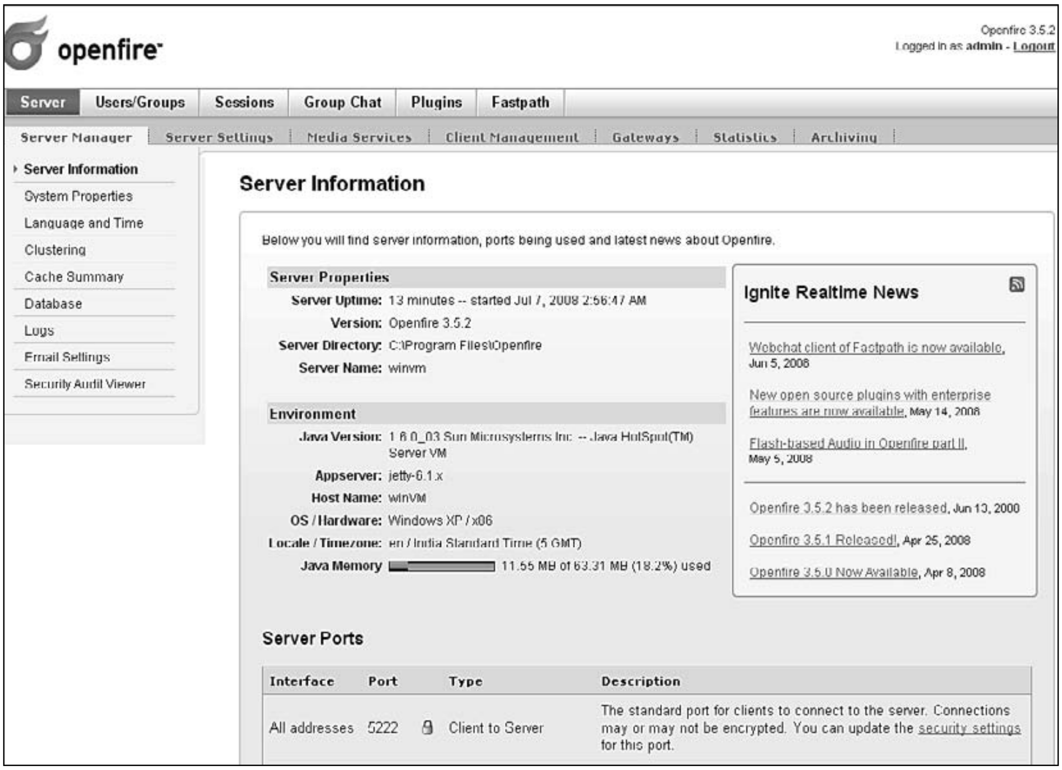
Finally, we'll seal the configuration with a password and provide an email address for the default admin user. On future visits to the server interface, you'll have to use the admin username and the password you've specified to log in to the Openfire server. When it's all done, you'll see a **Setup Complete!** message (see the screenshot below).



Setup Complete!

This installation of Openfire is now complete. To continue:

You can now log in to your server using the **Login to the admin console** button on the page, to see the following screenshot.



Server Information

Below you will find server information, ports being used and latest news about Openfire.

Server Properties

Server Uptime: 13 minutes -- started Jul 7, 2008 2:56:47 AM
Version: Openfire 3.5.2
Server Directory: C:\Program Files\Openfire
Server Name: winvm

Environment

Java Version: 1.6.0_03 Sun Microsystems Inc. -- Java HotSpot(TM) Server VM
Appserver: jetty-6.1.x
Host Name: winVM
OS / Hardware: Windows XP / x86
Locale / Timezone: en / India Standard Time (5 GMT)
Java Memory: 11.55 MB of 63.31 MB (18.2%) used

Ignite Realtime News

Webchat client of Fastpath is now available, Jun 5, 2008
New open source plugins with enterprise features are now available, May 14, 2008
Flash-based Audio in Openfire part II, May 5, 2008
Openfire 3.5.2 has been released, Jun 13, 2008
Openfire 3.5.1 Released!, Apr 25, 2008
Openfire 3.5.0 Now Available, Apr 8, 2008

Server Ports

Interface	Port	Type	Description
All addresses	5222	Client to Server	The standard port for clients to connect to the server. Connections may or may not be encrypted. You can update the security settings for this port.

Using The Server

We now have the server up and running. But how do users connect to the server? Or for that matter, where are the users? Well, let me introduce you to some instant messaging clients.

Installing The Spark Client

Installing the client in both Linux and Windows is a walk in the park. Remember that, like the server, Spark is also a Java application. But you don't have to worry about installing Java for Spark, because versions for all platforms of the client include the JRE.

On your chosen Linux distribution, download the Spark tarball and extract its contents under a standard location, such as `/opt`.

```
$ cd /tmp
$ tar zxvf spark_2_5_8.tar.gz
$ mv Spark/ /opt/
```

Now simply navigate to the newly-extracted directory and run the client by issuing the following commands:

```
$ cd /opt/Spark/
$ ./Spark
```

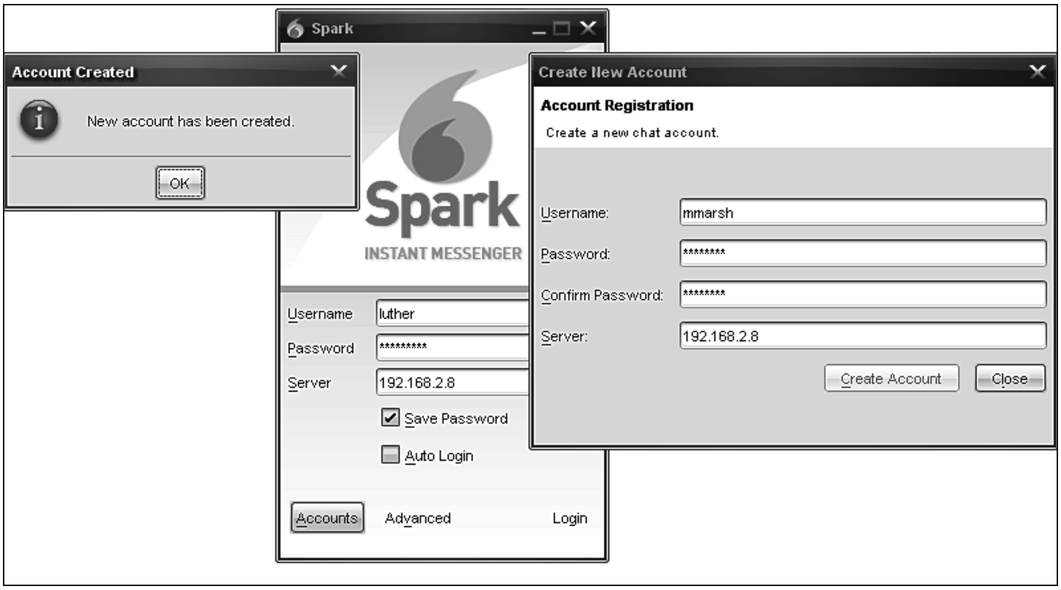
Look ma, no installation!

Let's move on to Windows now. Download the Spark installer for Windows and double-click on it to launch the installer. Like any other Windows application, you need to select the destination directory, the Start Menu entry, and decide whether to create desktop and quick launch icons. When it has completed the installation, the installer gives you the option to launch the client. Use it!

Tuning In With Spark

Now that we've installed Spark, it's time to use it to connect to our server. Before you do so, please remember to keep two things handy:

1. The name of the server: We specified this in the web interface when we setup Openfire. It's listed under Server Information in the web-based administration console.
2. The IP address of the server: You should already know this. The key to running a successful network application is to remember the IP addresses of the machines running those services. You can find the IP address of the machine running Openfire using the `ifconfig` command under Linux and `ipconfig` under Windows.



To create a new user with Spark, launch the client and click on **Accounts**. In the pop-up window enter a **Username**, and lock it with a **Password** (as shown in the previous screenshot). In the **Server** field, enter the IP address of the Openfire server and click on the **Create Account** button. When your account is created, Spark will display a pop-up box saying so.

By default, Openfire allows users to create accounts from IM clients. This isn't always a good policy and you can easily take away this power from the users. We'll cover this in a later chapter.

Now that we've created an account, let's log on to the server. Like most things with Openfire, this is easily done. To log in, launch Spark and, in the space provided, enter your username, password, and the IP address of the Openfire server. Optionally you may also choose to let Spark remember your password by selecting the **Save Password** checkbox. Now click the **Login** button and you're logged in.

Configuring Other IM Clients

There's no dearth of multi-protocol instant messaging clients (see `http://en.wikipedia.org/wiki/Comparison_of_instant_messaging_clients`) nor is there any shortage of Jabber clients (see `http://www.jabber.org/software/clients.shtml`). Not that I am complaining. It's just that I can't hand-hold you through the process of using your favorite IM client to connect to your Openfire server. Honestly though, and by now I am sure you'll trust me when I say this, there's hardly anything to it.

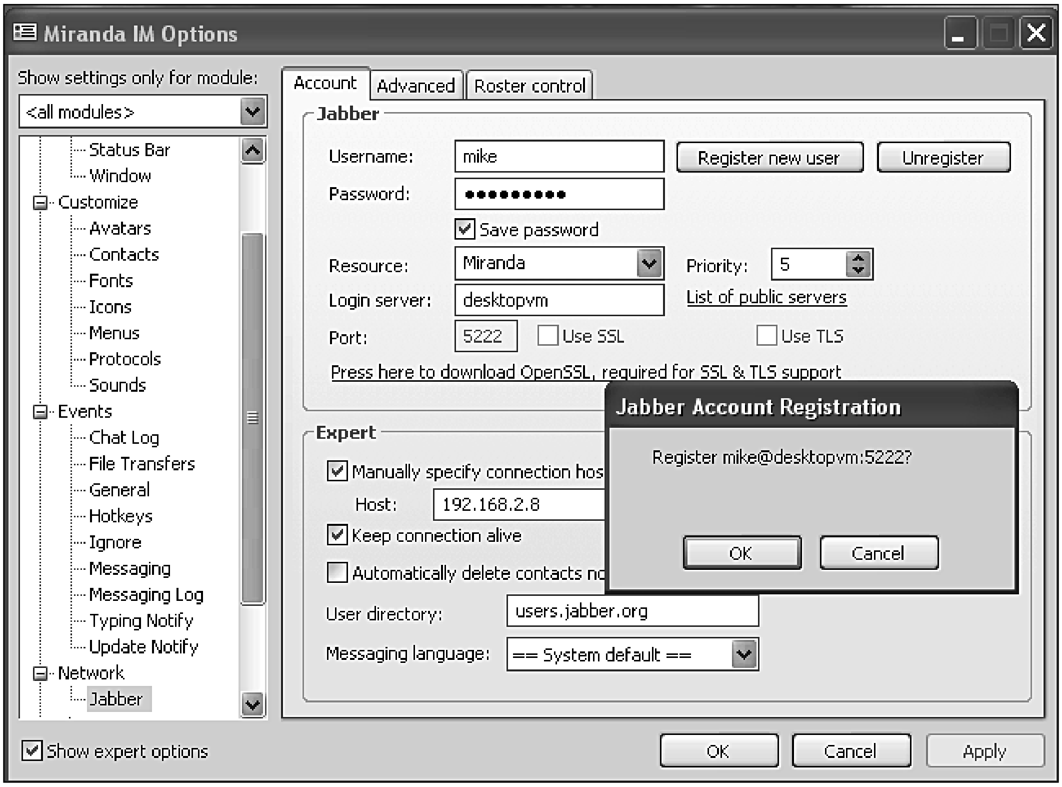
Just keep these three bits of information handy and you can configure just about any client:

1. The username and password of the user.
2. The name of the server as specified in the administration console.
3. The IP address of the server running Openfire.

To demonstrate, I'll configure three of my favorite multi-protocol clients under Windows and Linux—MirandaIM (Windows), Kopete (Linux:KDE), and Pidgin (Linux:GNOME).

MirandaIM

MirandaIM implements protocols as plugins, so while installing it please make sure you select the Jabber protocol for installation. When it's running, MirandaIM sits in the system tray close to the clock in the bottom right-hand side of your Windows system tray. Right-click on its icon in this system tray and navigate to **Main Menu | Options**. Scroll down to the **Network** tab and select the **Jabber** option. This divides the right-side pane into two sections. In the **Jabber** section, enter your **Username**, **Password** (see the following screenshot), and the name of the Openfire server. In the **Expert** section, select the **Manually specify connection host** checkbox and enter the IP address of the server in the host entry.



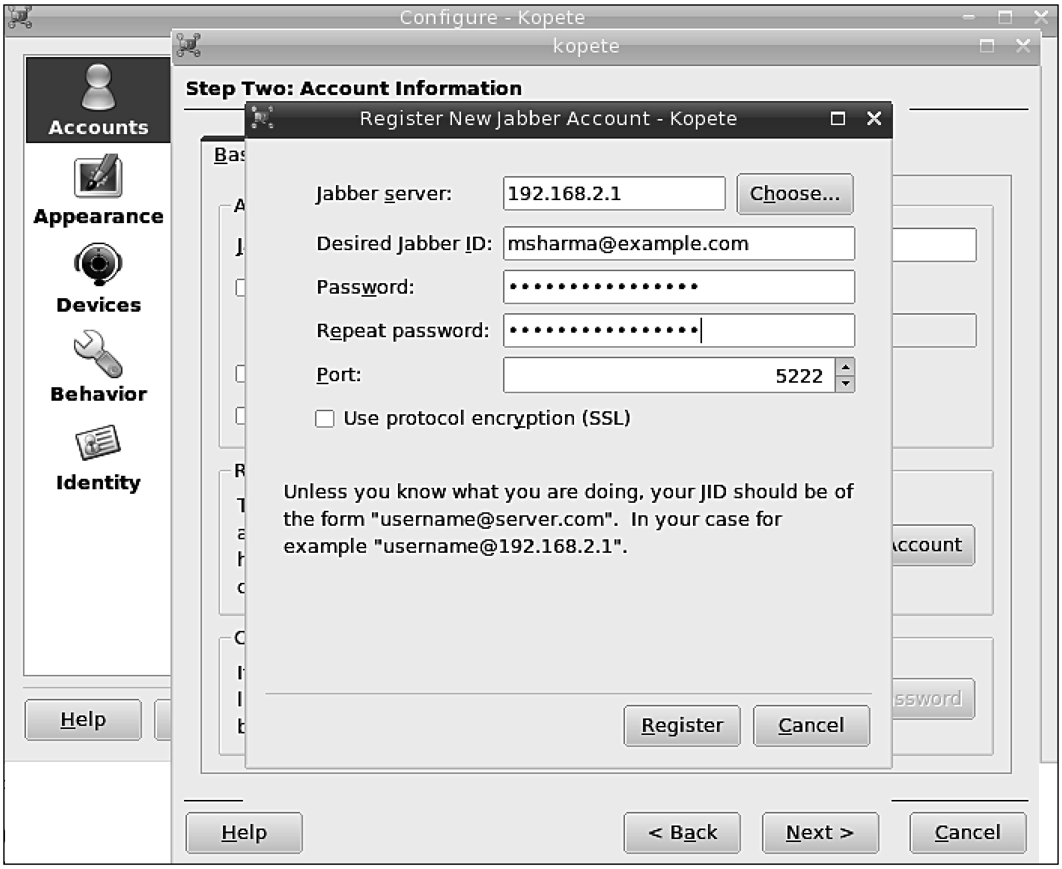
Username: mike
Password: ●●●●●●●●●
☑ Save password
Resource: Miranda
Priority: 5
Login server: desktopvm
Port: 5222 ☐ Use SSL ☐ Use TLS
☑ Manually specify connection hos
Host: 192.168.2.8
☑ Keep connection alive
☐ Automatically delete contacts nc
User directory: users.jabber.org
Messaging language: == System default ==
Register mike@desktopvm:5222?
☑ Show expert options

If you're registering a new user at this stage, click on the **Register new user** button. If you've already registered, just click on **OK** to save the settings. Finally, double-click on the icon in the system tray and from the MirandaIM interface, navigate to the **Status** drop-down menu and select **Online**.

Although MirandaIM is great, it lacks the level of depth in implementing the Jabber protocol as compared to dedicated Jabber clients for Windows such as Exodus (`http://code.google.com/p/exodus/`), Pandion (`http://www.pandion.be/screenshots/`), and Psi (`http://psi-im.org/`).

Kopete

To add an account in Kopete, right-click on its icon in the taskbar and select **Configure**. In the pop-up configuration window, click on the **New** button to bring up the Add Account wizard. Select the **Jabber** protocol and move on to the next step.



Jabber server: 192.168.2.1

Desired Jabber ID: msharma@example.com

Password: ••••••••••••••••

Repeat password: ••••••••••••••••

Port: 5222

☐ Use protocol encryption (SSL)

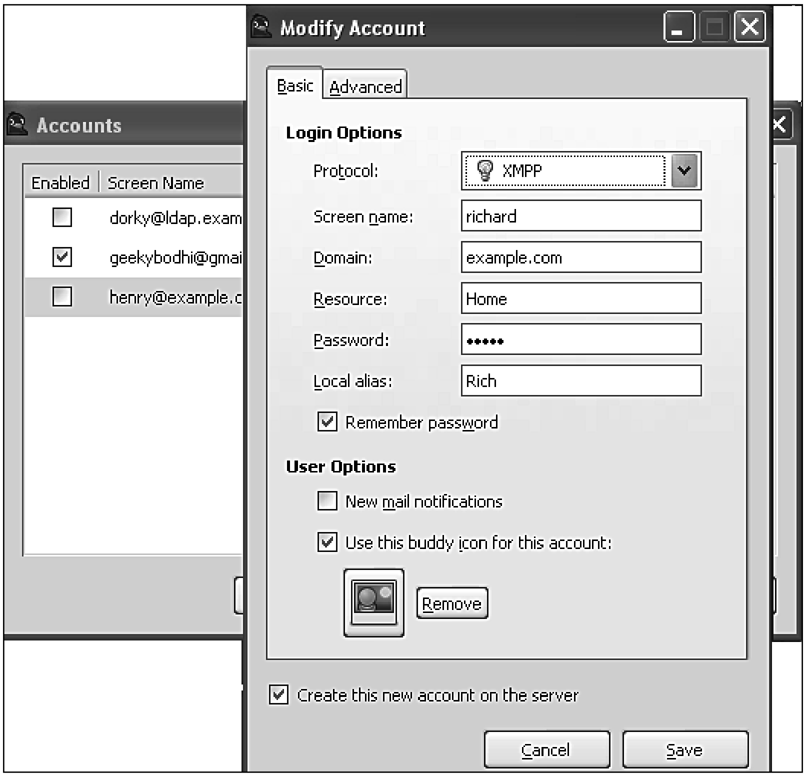
Unless you know what you are doing, your JID should be of the form "username@server.com". In your case for example "username@192.168.2.1".

If you want to add a new account, click on the **Register New Account** button. Here, enter the IP address of your Openfire server in the **Jabber server** field, the desired user name in the form username@server name (for example `msharma@example.com`), and a **Password** for the account, before clicking on the **Register** button (see the previous screen capture). Kopete will let you know if the registration was successful or if an error occured.

You'll be returned to step two of the account registration process. Here, enter the Jabber ID (username@server name) and password of the user. Under the **Connection** tab, select the option **Override default server information** and in the **Server** field, enter the IP address of your Openfire server. Now proceed to the last step and click on **Finish** to exit the setup process. You should now be logged in.

Pidgin

Pidgin not only supports multiple protocols but is multi-platform as well. In addition to Linux, it also runs on Windows. The instructions for setting up Pidgin under Linux and Windows are the same.



Enabled	Screen Name
☐	dorky@ldap.exam
✓	geekybodhi@gmai
☐	henry@example.c

Login Options

Protocol: XMPP

Screen name: richard

Domain: example.com

Resource: Home

Password: •••••

Local alias: Rich

✓ Remember password

User Options

☐ New mail notifications

✓ Use this buddy icon for this account:

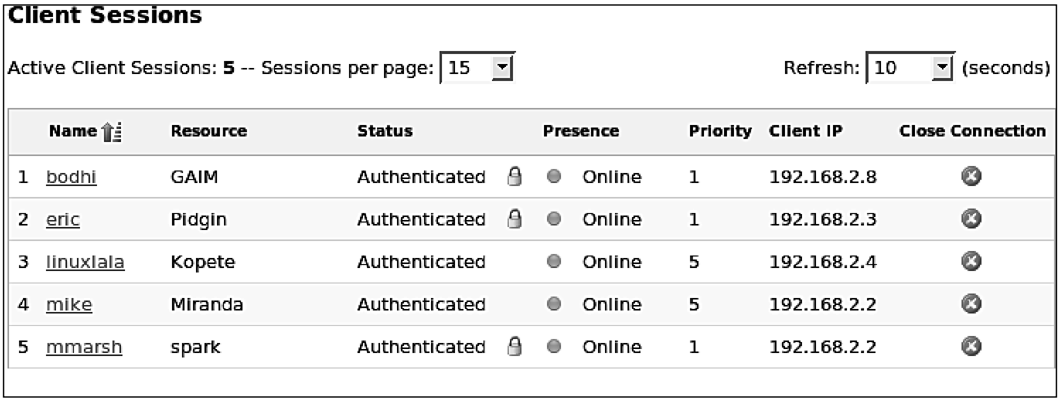
✓ Create this new account on the server

You can add accounts in Pidgin using the **Accounts** window, which is under Accounts | Manage. Click on the **Add** button at the bottom of the window to bring up the Add Account window. Under the **Basic** tab, select the **XMPP** protocol, and enter your screen name, which is your username on the Openfire server. Enter the name of your server in the **Domain** field, and your **Password** (see the previous screenshot). Now switch over to the **Advanced** tab and in the **Connect server** text box, enter the IP address of your Openfire server.

If you want to create this user on the server, make sure you select the **Create this new account on the server** checkbox at the bottom of the window. Click on **Save**.
If you're creating a new user, Pidgin will now ask you to enter a name and email address for the new user. Enter these details and click on the **Register** button. Pidgin will indicate whether the registration was successful.

On the other hand, if the user already exists, after saving your account details, you'll be passed back to the Accounts window. Close the window and head over to **Accounts | Enable accounts** and select the account that you want to log into.

You can also monitor the clients currently logged on to the system from the Openfire admin interface. Head over to **Sessions | Active Sessions | Client Sessions**. Here you will see a list of all the users currently logged on to the system, with their IP and client information, similar to what is shown in the screenshot below.



Client Sessions

Active Client Sessions: **5** -- Sessions per page: 15 Refresh: 10 (seconds)

	Name	Resource	Status		Presence	Priority	Client IP	Close Connection
1	bodhi	GAIM	Authenticated	🔒	● Online	1	192.168.2.8	⊗
2	eric	Pidgin	Authenticated	🔒	● Online	1	192.168.2.3	⊗
3	linuxlala	Kopete	Authenticated		● Online	5	192.168.2.4	⊗
4	mike	Miranda	Authenticated		● Online	5	192.168.2.2	⊗
5	mmarsh	spark	Authenticated	🔒	● Online	1	192.168.2.2	⊗

Touch down! We're at the bottom of our rabbit hole. When I first got here, unlike Dorothy, I wasn't confused; I felt elated. I hope you feel the same.

Summary

In this chapter, we've prepared an environment for Openfire, and installed and configured the server on both Windows and Linux. From installing the Java Runtime Environment to launching and tweaking the server, we've covered all the steps it takes to have to a fully-functional Openfire server.

It sounds like a lot of work, but if you get down to listing the steps, there aren't many. The whole process doesn't take more than half-an-hour. Credit must be given to the Openfire developers, not only for packaging the server for multiple Linux distributions and operating systems, but also for taking the load off the users who are setting it up.

The server we've setup in this chapter will work for most enterprises. Depending on their current system landscape, some large corporations might want to hook up their directory services with Openfire or use an external database to hold Openfire's data. We'll get to these scenarios in due course, and while they are important options, they are by no means necessary.

In the next chapter, we'll roam around the Openfire admin interface, tweaking the server to our liking as we move forward.

3
Fine-Tuning Openfire

In the previous chapter, we setup Openfire, installed and configured some IM clients, used these to add some users, and logged into our IM server. Basically, we've done what 70% of IM users would be happy with. But we've barely scratched the surface of Openfire. Our setup is too liberal, too raw. Although this is a book on running the Openfire server, we haven't even looked around the web administrator interface yet. In this chapter, we'll run around the interface, and get a feel for things as we tweak our server to the liking. In this chapter, we will cover:

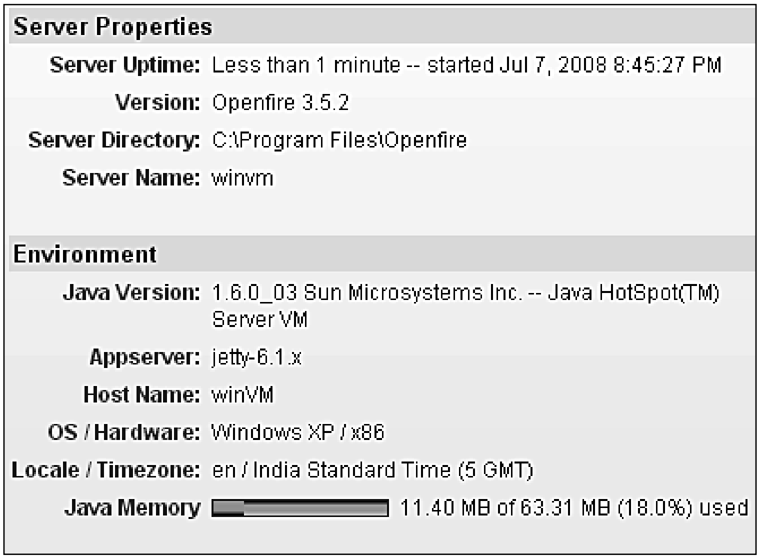
- Details about the server and server ports
- Server cache and its properties
- Policing and tuning the server
- Curbing indiscriminate registration
- Handling offline users
- Resolving resource conflicts
- Updates
- Sending administrative messages
- Extending with plugins

Get To Know Your Server

Openfire has a simple and straightforward administration console. It's divided into several tabs with each tab housing multiple configuration options. From within these you can tweak every aspect of the server and integrate it with existing network services. The other aspect of running a server is being able to monitor its activity. The interface is also designed to provide you with visual feedback at a glance to help you keep tabs on the server and gauge its performance.

Vital Statistics

When you log in into the server, you're presented with basic details about the server—what version of Openfire are you running, where is it installed, what platform are you running it on, how long has it been running, what's the version of Java powering the server, how much memory is it consuming, and so on (see the following screenshot).



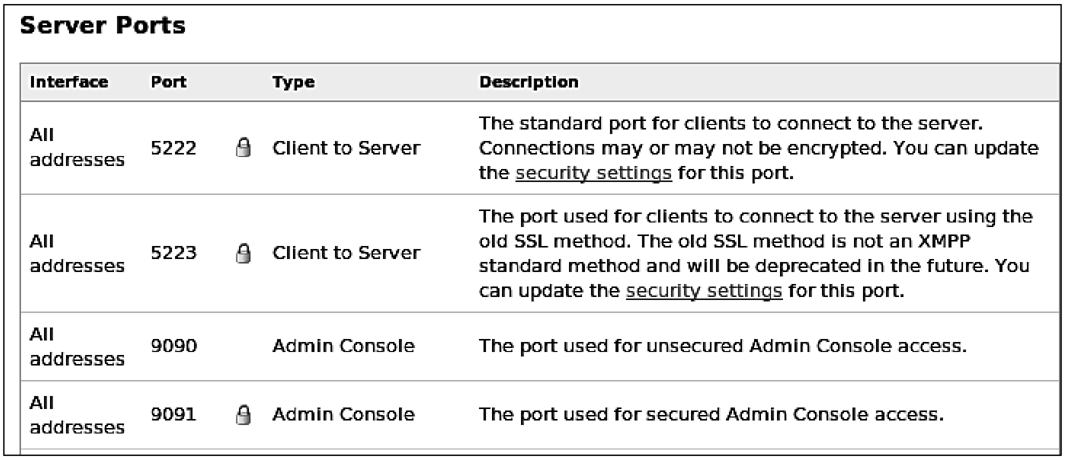
Server Properties

Server Uptime:	Less than 1 minute -- started Jul 7, 2008 8:45:27 PM
Version:	Openfire 3.5.2
Server Directory:	C:\Program Files\Openfire
Server Name:	winvm

Environment

Java Version:	1.6.0_03 Sun Microsystems Inc. -- Java HotSpot(TM) Server VM
Appserver:	jetty-6.1.x
Host Name:	winVM
OS / Hardware:	Windows XP / x86
Locale / Timezone:	en / India Standard Time (5 GMT)
Java Memory	11.40 MB of 63.31 MB (18.0%) used

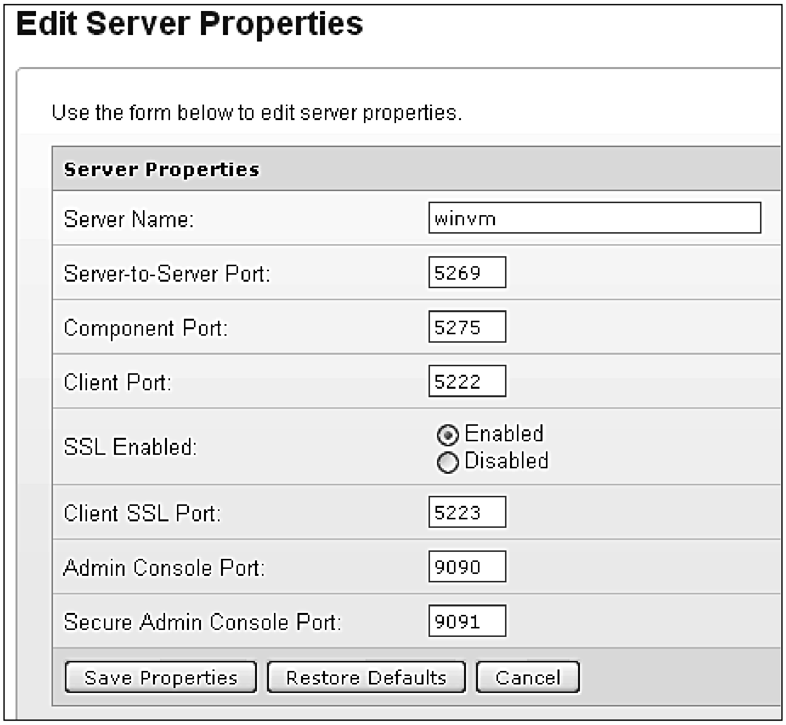
On this page, you can also see the ports Openfire uses, and their purposes (refer to the following screenshot).



Server Ports

Interface	Port		Type	Description
All addresses	5222	🔒	Client to Server	The standard port for clients to connect to the server. Connections may or may not be encrypted. You can update the security settings for this port.
All addresses	5223	🔒	Client to Server	The port used for clients to connect to the server using the old SSL method. The old SSL method is not an XMPP standard method and will be deprecated in the future. You can update the security settings for this port.
All addresses	9090		Admin Console	The port used for unsecured Admin Console access.
All addresses	9091	🔒	Admin Console	The port used for secured Admin Console access.

Like other parts of the server, these ports are also configurable. Remember the default secured (9091) and unsecured (9090) ports for accessing the server administration console? To change them from their default values, click on the **Edit Properties** button beneath the **Server Ports** listing.



Edit Server Properties

Use the form below to edit server properties.

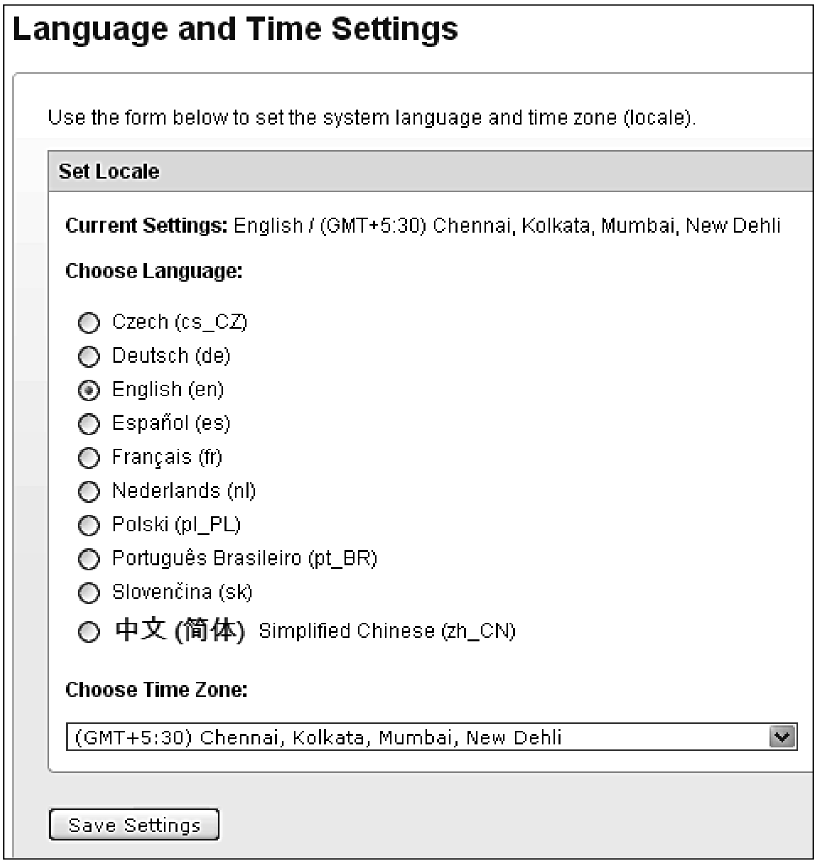
Server Properties	
Server Name:	winvm
Server-to-Server Port:	5269
Component Port:	5275
Client Port:	5222
SSL Enabled:	[✓ Enabled] Disabled
Client SSL Port:	5223
Admin Console Port:	9090
Secure Admin Console Port:	9091

In the page that opens (refer to the previous screenshot), Openfire lets you change the value of any setting. Because we are concerned only with the administration console ports, we'll alter only these and leave the rest as-is.

Once you're done making the changes, click on the **Save Properties** button to save the changes. You'll have to restart the server for the changes to take effect. If you've changed the administration console port, make sure that you use the new port number when logging into the console.

Please make sure that you note down the port numbers somewhere that you can find them easily. This is all the more important when you change the port number of the administration console. If you forget this, you will not be able to log in into the administration console!

Because Openfire is multilingual, you can also choose to run it in several other languages in addition to English. During setup, in the previous chapter, we chose English as our default language. If you're feeling adventurous and want to test your language skills, you can change the language of the interface in the **Language and Time Settings** section (refer to the following screenshot). All joking aside, you really shouldn't fool around with the language setting unless you're absolutely fluent in the other supported languages. You might end up with a fully functional yet useless system.



Language and Time Settings

Use the form below to set the system language and time zone (locale).

Set Locale

Current Settings: English / (GMT+5:30) Chennai, Kolkata, Mumbai, New Dehli

Choose Language:

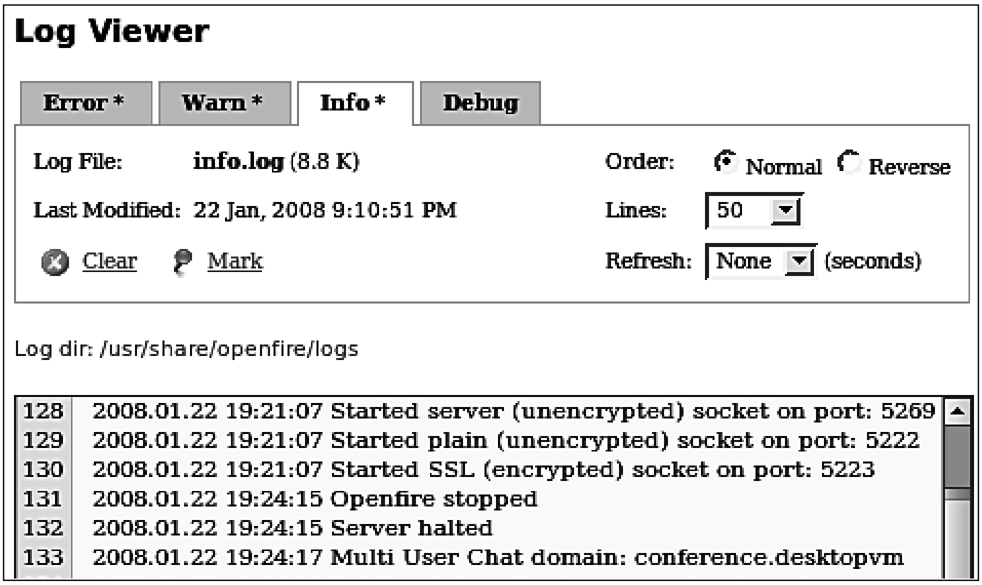
- ○ Czech (cs_CZ)
- ○ Deutsch (de)
- ● English (en)
- ○ Español (es)
- ○ Français (fr)
- ○ Nederlands (nl)
- ○ Polski (pl_PL)
- ○ Português Brasileiro (pt_BR)
- ○ Slovenčina (sk)
- ○ 中文 (简体) Simplified Chinese (zh_CN)

Choose Time Zone:

(GMT+5:30) Chennai, Kolkata, Mumbai, New Dehli

You can also choose to change the time zone you're in from a drop-down menu in this section. Again, don't be too adventurous with this option, as the logs are time stamped based on the time zone you've selected here. If you select a time zone that's different from the one you're in, you'll have trouble understanding the logs when troubleshooting a problem.

Talking of logs, the administration console has a log viewer that you can use to view the server logs. From the Logs section, you can get a snapshot of the **Error**, **Warn**, **Info**, and **Debug** logs (refer to the following screenshot).



Log File: info.log (8.8 K)
Last Modified: 22 Jan, 2008 9:10:51 PM
Order: [✓ Normal] Reverse
Lines: 50
Refresh: None (seconds)

Log dir: /usr/share/openfire/logs

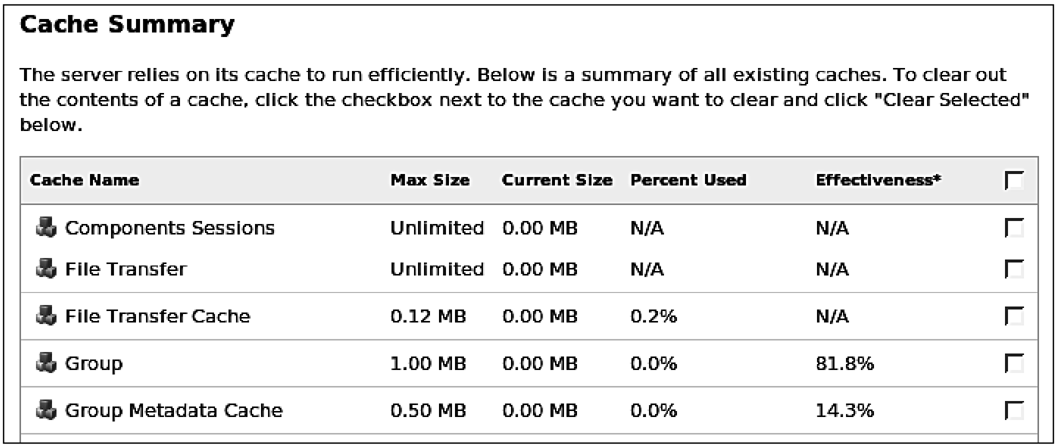
```
128  2008.01.22 19:21:07 Started server (unencrypted) socket on port: 5269
129  2008.01.22 19:21:07 Started plain (unencrypted) socket on port: 5222
130  2008.01.22 19:21:07 Started SSL (encrypted) socket on port: 5223
131  2008.01.22 19:24:15 Openfire stopped
132  2008.01.22 19:24:15 Server halted
133  2008.01.22 19:24:17 Multi User Chat domain: conference.desktopvm
```

All of these logs, except **Debug**, are enabled by default. Enable the **Debug** log if the server is misbehaving—for example, after you alter a system property or add a new component. To do this, head over to the **Debug** tab and select the **Enabled** option and click on **Save Changes**.

Server Cache and Properties

The Openfire server manages several tasks and resources. For all the components it has to maintain, ther server keeps track of their maximum permissible size, their current size, usage as a percentage, and the effectiveness of the resource. The effectiveness, as explained in the interface itself, measures how well your cache is working. If the effectiveness is low, this usually means that the cache is too small. Openfire flags such caches. You can view these caches from the **Cache Summary** section (refer to the following screenshot).

To clear any individual cache, select the checkbox in the rightmost column for the cache you want to empty, then scroll down and click the **Clear Selected** button.



Cache Summary

The server relies on its cache to run efficiently. Below is a summary of all existing caches. To clear out the contents of a cache, click the checkbox next to the cache you want to clear and click "Clear Selected" below.

Cache Name	Max Size	Current Size	Percent Used	Effectiveness*	☐
Components Sessions	Unlimited	0.00 MB	N/A	N/A	☐
File Transfer	Unlimited	0.00 MB	N/A	N/A	☐
File Transfer Cache	0.12 MB	0.00 MB	0.2%	N/A	☐
Group	1.00 MB	0.00 MB	0.0%	81.8%	☐
Group Metadata Cache	0.50 MB	0.00 MB	0.0%	14.3%	☐

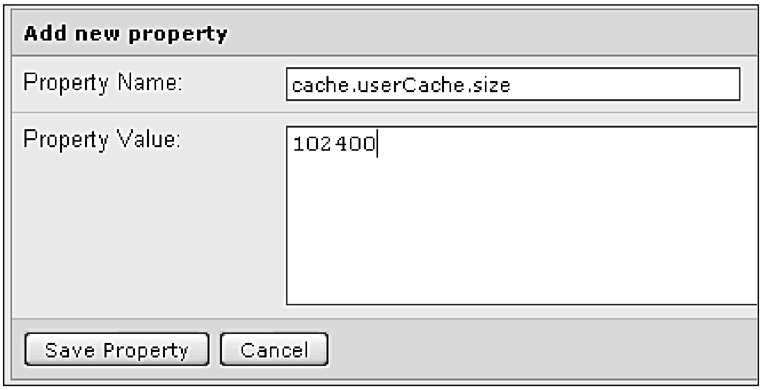
Editing a particular cache size isn't a particularly straightforward task. To change the size, you'll have to rely on another Openfire statistics feature—System Properties. The System Properties section of the Openfire administrator interface displays some of the variables used by Openfire, and their current values.

So let's now change the size of a user's cache. By default, under the **Cache Summary** section, a user has a maximum of 0.50 MB of cache at his or her disposal. Let's say you'll have thousands of users on your network and you wish to reduce their cache to 100 KB. Head over to **System Properties**, scroll down to the **Add new property** section (refer to the following screenshot) and in the space provided, enter these values:

Property Name: **cache.userCache.size**

Property Value: **102400**

Click on the **Save Property** button to add the property to the pool of system properties.

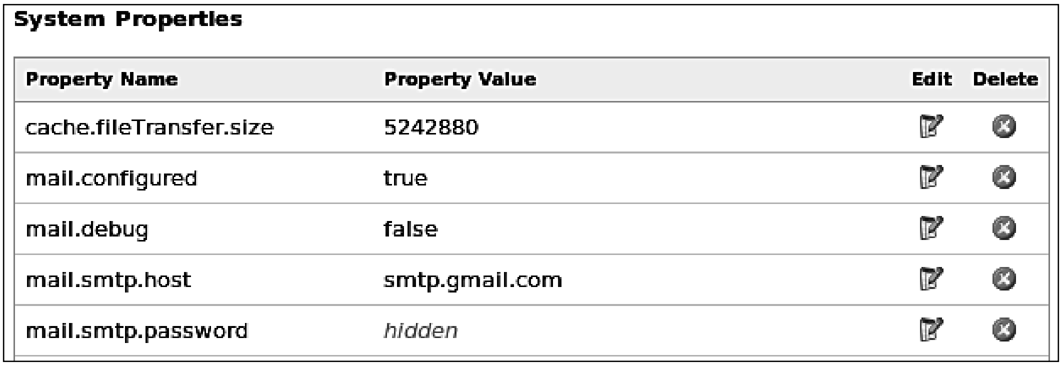


Add new property

Property Name:	cache.userCache.size
Property Value:	102400

The size of system properties should be entered in bytes. Use Google's search bar to convert a size from the more familiar denomination of MBs and GBs into bytes. For example, entering "100 KB in bytes" in Google's search bar should return the result "100 kilobytes = 102400 bytes".

You'll have to restart the Openfire server before the change in value is reflected in the **Cache Summary** section. Once you've added the variable as a system property, you can click on the icon under the **Edit** column adjacent to the property name, to change its value. You can click the icon under the **Delete** column to remove the variable from under the **System Property** section (refer to the screenshot below). If you delete the `cache.userCache.size` variable, its size will revert to the default 0.50 MB. Any change to the cache will take effect only after the Openfire server has been restarted.



System Properties

Property Name	Property Value	Edit	Delete
cache.fileTransfer.size	5242880	✎	⊗
mail.configured	true	✎	⊗
mail.debug	false	✎	⊗
mail.smtp.host	smtp.gmail.com	✎	⊗
mail.smtp.password	*hidden*	✎	⊗

In addition to the cache elements (see `http://wiki.igniterealtime.org/display/WILDFIRE/How+to+configure+Wildfire%27s+caches`), there are dozens of Openfire properties (seet `http://www.igniterealtime.org/community/docs/DOC-1061`) that are documented on the Openfire website.

Policing and Tuning The Server

Remember how I told you that we were too liberal with the basic setup in the previous chapter? It's now time to do something about it. In this section, we'll look at some Openfire functions that will help us police the network. We'll resolve conflicts, control indiscriminate user registration, and keep watch while the users aren't online.

Curb Indiscriminate Registration

Allowing users to register themselves is, like all server decisions, a good and a bad idea. You have to balance convenience with how tight you want to run the ship. If you let users register themselves, there's a strong possibility you'll have more Openfire users than there are people using the network—who's stopping them from registering a new nickname every time they have a new favorite song?

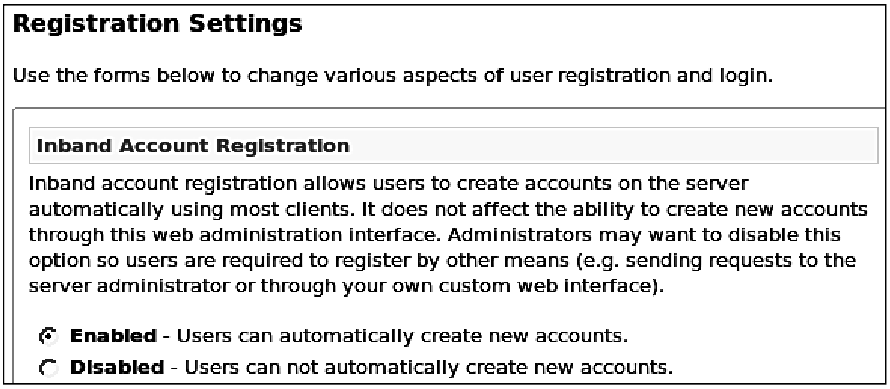
By restricting user registration to the administrator, you'll be reducing the clutter of unused nicknames. Of course, you'll have to take time out to process each request for a new nickname, which, depending on the size of the company and number of requests, could be a sizable distraction.

[Managing users with an external system like a Directory Server is covered in an upcoming chapter.]

I call the following the"three steps to network domination" a.k.a. "revenge of the network admin".

Step 1: Stop users from registering themselves

If you think that keeping a network clean and streamlined is paramount then, despite the extra work it entails, you'll have to tell Openfire to not pay attention to user's requests for registering nicknames. Head over to the **Registration & Login** section under **Server Settings** and disable **Inband Account Registration**. Because this option is enabled by default, users can connect to the Openfire server and create accounts for themselves from their clients. Once inband account registration is disabled, users will not be able to create accounts from their clients (refer to the following screenshot).



Registration Settings

Use the forms below to change various aspects of user registration and login.

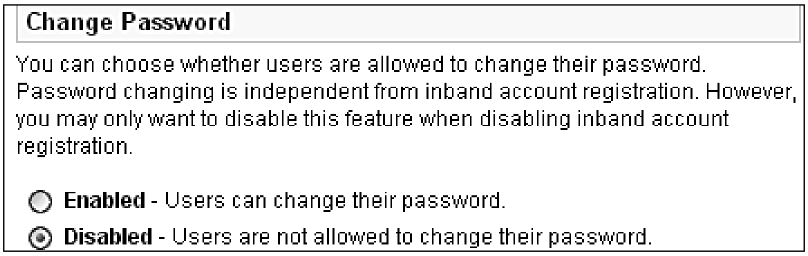
Inband Account Registration

Inband account registration allows users to create accounts on the server automatically using most clients. It does not affect the ability to create new accounts through this web administration interface. Administrators may want to disable this option so users are required to register by other means (e.g. sending requests to the server administrator or through your own custom web interface).

[✓ **Enabled** - Users can automatically create new accounts.]
Disabled - Users can not automatically create new accounts.

Step 2: Don't let them change passwords

Once you've taken away your users' power to register themselves, you might as well withdraw their ability to change their passwords. This can be easily done by setting the **Change Password** option to **Disabled** (refer to the following screenshot) under the **Registration & Login** section. The force is strong in this one!



Change Password

You can choose whether users are allowed to change their password. Password changing is independent from inband account registration. However, you may only want to disable this feature when disabling inband account registration.

Enabled - Users can change their password.
[✓ **Disabled** - Users are not allowed to change their password.]

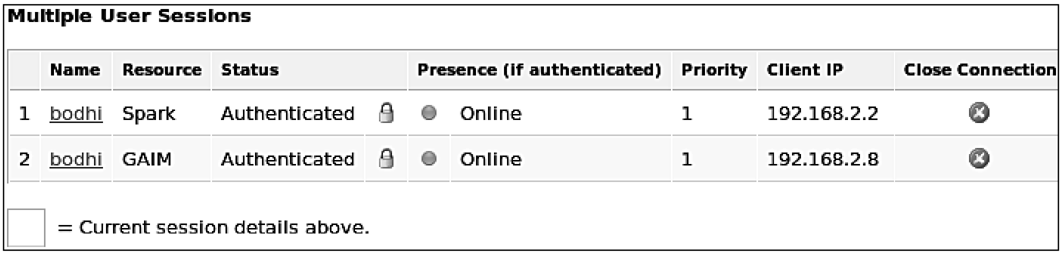
Before you put on your Darth Vader helmet, you might want to ponder over this option. In all seriousness, changing passwords often is a good policy. Think twice before setting this option.

Step 3: Restrict Login

Do you have a network user who is ignoring IM usage rules? Make them pay by denying them access to the network completely. This is done using Openfire's white list. If Openfire reads a list of comma-separated IP addresses (such as 192.168.2.131, 192.168.2.170) or range (for example, 192.168.56.*) in the box provided under the Restrict Login box, it'll only allow users to log in to the network from these IP addresses. If the list is empty, users are allowed to connect to Openfire from anywhere. So the trick is to add all the users' IP addresses to this list, and when someone misbehaves, simply delete their IP address. Ah! True power!

Resolving Resource Conflicts

While you were configuring clients, did you notice the **Resource** field? In most clients, the **Resource** field has the name of the IM client you're using (refer to the following screenshot). The XMPP protocol is designed to use this resource name along with the login name of a user to uniquely identify each connection. For example, a user with the user name **bodhi** can connect from two different clients—Pidgin and Spark—since Openfire identifies him as bodhi@ldap.example.com/Pidgin and bodhi@ldap.example.com/Spark.


Multiple User Sessions

	Name	Resource	Status		Presence (if authenticated)		Priority	Client IP	Close Connection
1	bodhi	Spark	Authenticated		●	Online	1	192.168.2.2	⊗
2	bodhi	GAIM	Authenticated		●	Online	1	192.168.2.8	⊗

☐ = Current session details above.

Now what if a new connection from a user already logged into the system requests the same resource name as the one already in use? For example, what if the user "bodhi" decides to login from the same client? Because the system can't keep track of and handle two identical connections, it has to decide whether to kick out the existing connection or stop the new connection from logging in.

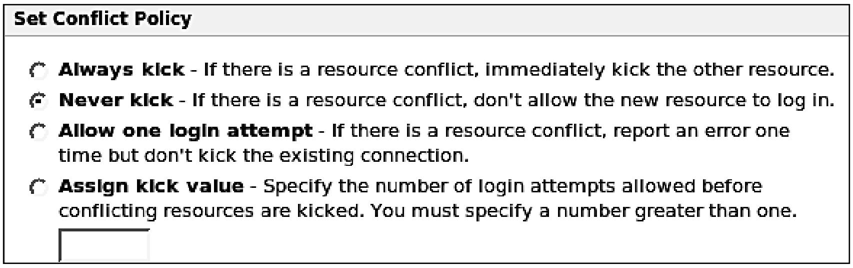
From under the Resource Policy under Server Settings, Openfire offers four different options to resolve resource conflicts (refer to the following screenshot too):

Option 1: **Always kick**—This is the default option. When Openfire detects a resource conflict, it kicks off the new connection.

Option 2: **Never kick**—Select this option if you want to block the new connection from logging into the server. Openfire refuses to establish the connection, with a message which says that the account is already signed in.

Option 3: **Allow one login attempt**—If you select this option, Openfire will report an error, but doesn't kick the existing connection.

Option 4: **Assign kick value**—This option allows you to specify the number of login attempts Openfire will allow before kicking the conflicting resources.



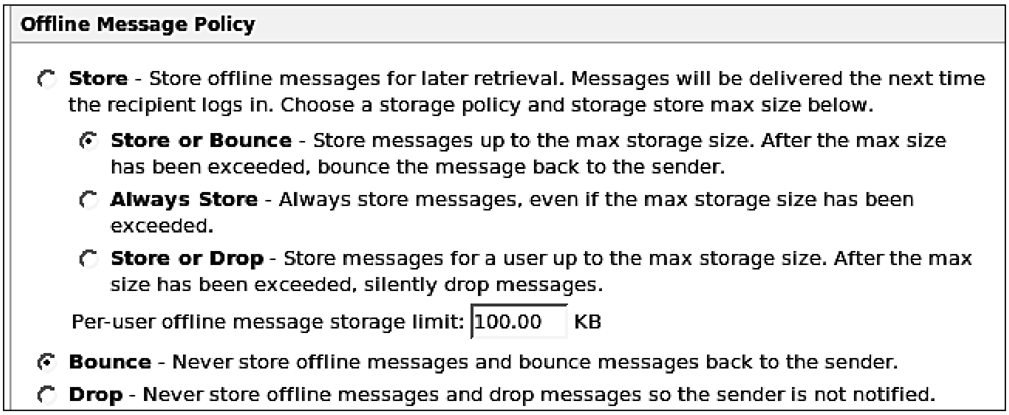
Set Conflict Policy

- **Always kick** - If there is a resource conflict, immediately kick the other resource.
- [✓ **Never kick** - If there is a resource conflict, don't allow the new resource to log in.]
- **Allow one login attempt** - If there is a resource conflict, report an error one time but don't kick the existing connection.
- **Assign kick value** - Specify the number of login attempts allowed before conflicting resources are kicked. You must specify a number greater than one.

Handling Offline Users

By default, Openfire is setup to store all offline messages. If a user on the network sends a message to another user who is currently offline, Openfire graciously stores the message in its offline cache. When the offline user logs in, Openfire opens a chat window and displays the offline messages for that user.

Under the Offline Messages section under Server Settings, you specify a maximum offline message storage limit per user, which by default is set to 100 KB. Offline messages are stored until they reach this limit. When Openfire can't store the messages, they are bounced back to the sender. This is Openfire's default **Store or Bounce** option (see the following screenshot). But since resources on the server are limited, you can choose to handle offline messages in a variety of ways.



Offline Message Policy

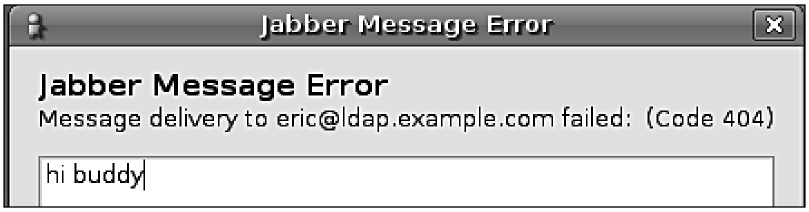
- **Store** - Store offline messages for later retrieval. Messages will be delivered the next time the recipient logs in. Choose a storage policy and storage store max size below.
  - [✓ **Store or Bounce** - Store messages up to the max storage size. After the max size has been exceeded, bounce the message back to the sender.]
  - **Always Store** - Always store messages, even if the max storage size has been exceeded.
  - **Store or Drop** - Store messages for a user up to the max storage size. After the max size has been exceeded, silently drop messages.

  Per-user offline message storage limit: 100.00 KB
- [✓ **Bounce** - Never store offline messages and bounce messages back to the sender.]
- **Drop** - Never store offline messages and drop messages so the sender is not notified.

Option 1: **Always Store**—If you select this option, Openfire will keep storing the messages even after the maximum limit has been reached.

Option 2: **Store or Drop**—This option is similar to the default option with a subtle difference. If you select this option, Openfire will store offline messages up to the maximum storage size. Once this has been exceeded, the server silently drops the messages and no-one is any the wiser.

Option 3: **Bounce**—If you're really short on resources, select the **Bounce** option and Openfire will bounce messages sent to offline users back to their senders. The screenshot below shows how a bounced message appears:


Jabber Message Error

Jabber Message Error

Message delivery to eric@ldap.example.com failed: (Code 404)

hi buddy

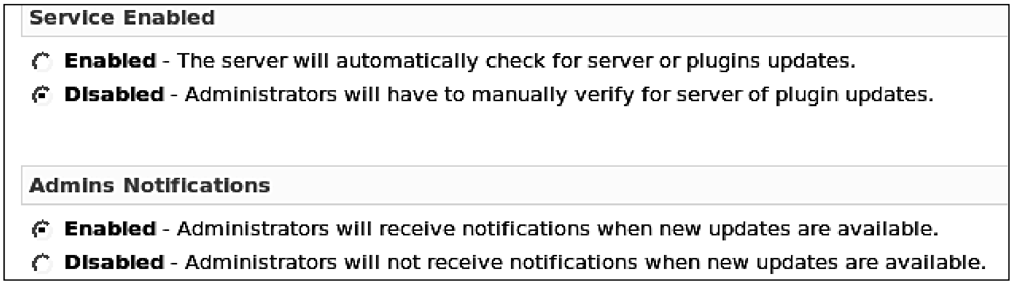
Option 4: **Drop**—Ok, so you're short on resources and evil. When you toggle the Drop option, not only are the offline messages not stored, they aren't even returned to the sender. The only evidence the message was ever sent but not delivered is in the server logs!

Nurturing The Server

Managing updates is all part of administering a server. Openfire is no different. The server is capable of receiving notifications when a new version is available. This behavior is selected from the Manage Updates section under Server Settings.

If you don't like the idea of the server listening for updates on the Internet, you can turn off this feature by selecting the **Disabled** option under **Service Enabled**. When this service is disabled, you'll have to manually check the Openfire website for new versions of the server and other components such as plugins, which we'll get to very shortly.

But if, on the other hand, you'd like to allow Openfire to keep you updated, keep the service enabled and select the way the server is connected to the Internet. Unless the server is behind a proxy server, you can safely leave the **Connection Method** option to its default value. If you are behind a proxy server, you'll have to select the **Proxy Connection** option and specify the location (hostname and port) of the proxy server in the fields provided.


Service Enabled

○ **Enabled** - The server will automatically check for server or plugins updates.

[✓ **Disabled** - Administrators will have to manually verify for server of plugin updates.]

Admins Notifications

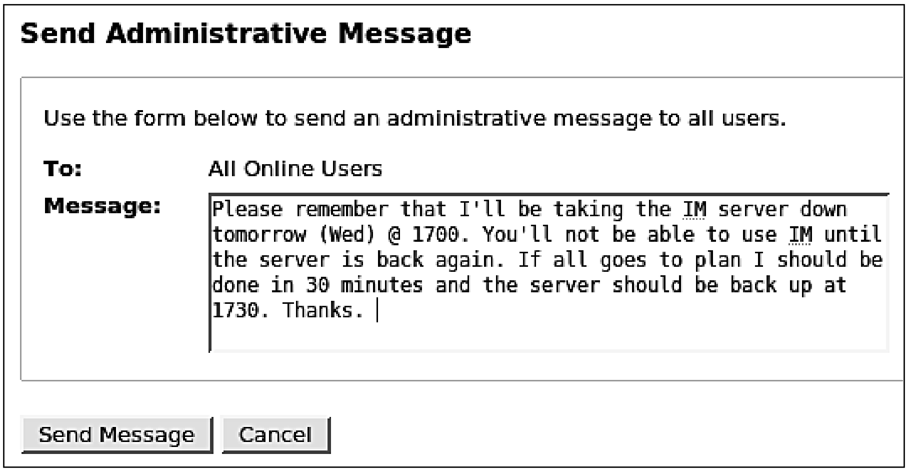
[✓ **Enabled** - Administrators will receive notifications when new updates are available.]

○ **Disabled** - Administrators will not receive notifications when new updates are available.

You can also tweak Openfire's behavior when it receives updates. By default, Openfire will send a notification to the administrator as and when a new update is available. But you can configure Openfire to spare you the notification. To do so, select the **Disabled** option under **Admins Notifications** (refer to the previous screenshot).

Talk to Users

One of the least-understood principles of successfully running a server, is taking it offline! In my experience, things such as updating, shifting, or migrating the server always seem to happen at the wrong time. Of course, I am saying this from a user's perspective, but my network admininistrator could have made my life a lot easier by talking to me, communicating their maintenance schedule.


Send Administrative Message

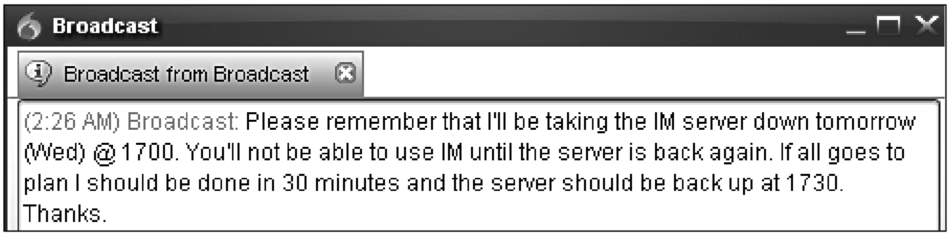
Use the form below to send an administrative message to all users.

To: All Online Users

Message:

```
Please remember that I'll be taking the IM server down
tomorrow (Wed) @ 1700. You'll not be able to use IM until
the server is back again. If all goes to plan I should be
done in 30 minutes and the server should be back up at
1730. Thanks.
```

An Openfire administrator has several tools at his or her disposal to communicate with the users on the network more efficiently. One of them is tucked inside the **Sessions** tab, appropriately listed under **Tools**. The **Send Message** tool has a simple interface—a **To** field and a **Message** box. The **To** field is preset to send message to all online users. All you need to do is type in a message and click the **Send Message** button (refer to the previous screenshot).



(2:26 AM) Broadcast: Please remember that I'll be taking the IM server down tomorrow (Wed) @ 1700. You'll not be able to use IM until the server is back again. If all goes to plan I should be done in 30 minutes and the server should be back up at 1730. Thanks.

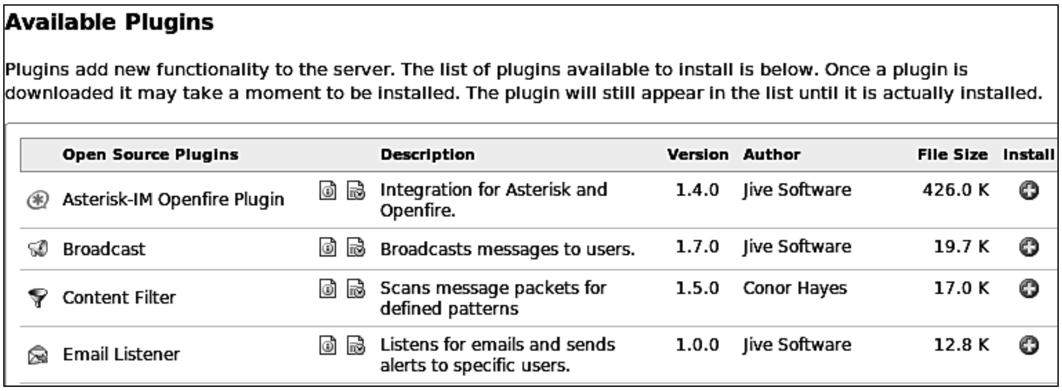
The message is delivered to all users including the administrator (see the previous screenshot). The only drawback is that such a broadcast doesn't follow the rules of a 'normal' offline message and will not be delivered to users that aren't online.

There are other communications tools, but we'll cover them when we know how to handle users and groups in Openfire.

Extending with Plugins

There's one thing I want in any server software I use—plugins. Thankfully some of the best ones I use, such as Apache, MySQL, Perl, etc., are all pluggable in their own way. Needless to say, Openfire, too, is pluggable. Because there's (yes, here I go again) hardly anything to installing plugins in Openfire, I'll show you one plugin all admins should use in this section. There are other powerful plugins as well that we'll cover in later chapters.

Installing a plugin is simple. Notice the set of tabs at the top of the Openfire administration console? There's one called Plugins. The page this tab takes you to lists all of the installed plugins. As we haven't installed any yet, there will be none listed. From the navigation menu on this page, click on **Available Plugins**. This section has several plugins, several open source ones, and a couple of commercial ones. Every plugin has a brief **Description** (refer to the following screenshot), a **Version** number, the name of its **Author**, and small buttons that you can use to read the plugin's accompanying `README` and `changelog` files. There's also a green "**+**" button, which, when pressed, installs that plugin.



Available Plugins

Plugins add new functionality to the server. The list of plugins available to install is below. Once a plugin is downloaded it may take a moment to be installed. The plugin will still appear in the list until it is actually installed.

Open Source Plugins	Description	Version	Author	File Size	Install
Asterisk-IM Openfire Plugin	Integration for Asterisk and Openfire.	1.4.0	Jive Software	426.0 K	⊕
Broadcast	Broadcasts messages to users.	1.7.0	Jive Software	19.7 K	⊕
Content Filter	Scans message packets for defined patterns	1.5.0	Conor Hayes	17.0 K	⊕
Email Listener	Listens for emails and sends alerts to specific users.	1.0.0	Jive Software	12.8 K	⊕

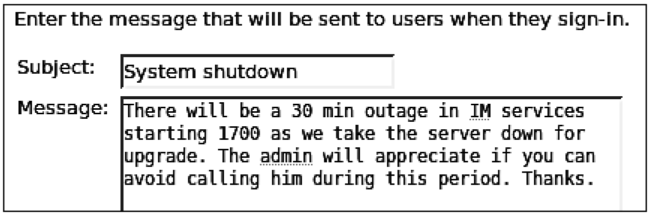
It really doesn't get any easier than this!

Message of The Day

The MotD plugin is another tool that'll help the administrator communicate with users on the network. I think of MotD as a recurring alarm. Once configured, the plugin will send messages to users as and when they login. You can use it to send long-term messages such as a link to the organization's IM usage guidelines or a list of intranet URLs and important settings. You can also use it to alert your users on special occasions such as an imminent system shutdown!

So let's go get it. I am sure that you know drill—head over to the Available Plugins section and click the MotD plugin's green 'Add' button. Wait for a while as the plugin downloads, and then watch it disappear from the list of available plugins! It now resides under the Plugins section, which keeps track of the installed plugins.

Once the plugin is installed, head over to the **Users/Groups** tab. Here, along with several other options for Users and Groups which we'll play with in later chapters, you'll notice the MotD properties section. Click on the checkbox **Enable MotD** and enter the **Subject** and **Message** that you want to send to all users when they log in (see the screenshot below).



Enter the message that will be sent to users when they sign-in.

Subject: System shutdown

Message: There will be a 30 min outage in IM services starting 1700 as we take the server down for upgrade. The admin will appreciate if you can avoid calling him during this period. Thanks.

The message will be broadcast to all users as and when they login until you decide to disable it by unchecking the **Enable MotD** checkbox. Remember that users already online will not receive this message until they log off and log in again.

Summary

We've covered a lot of ground in this chapter. You should now be well-equipped to run and administer an Openfire server in a single-office organization. You should also be comfortable moving around in the administration console. Tweaking ports, and checking up and adjusting system properties and various caches should no longer sound like a daunting task.

Some of the common network admin tasks that you should be well-rehearsed in are handling user registration and offline messages, checking for updates, and resolving login conflicts, while efficiently communicating with users on the network and installing plugins. Look at you go!

4
Organizing Users

Throughout this book, I've been mentioning how IM increases productivity; in this chapter the rubber hits the road.

But before we can get started, we'll have to understand how users are added and organized in Openfire. On the upside, learning to add, remove, and manage users is one of the last tasks that you will need to know if you're using Openfire in a small-to-medium sized single office. By the time you're done with this chapter, you'll be itching to roll out Openfire.

In this chapter, we will cover:

- Adding, editing, and deleting users
- Temporarily suspending users
- Organizing users into groups
- Editing and deleting groups
- Pre-populating rosters

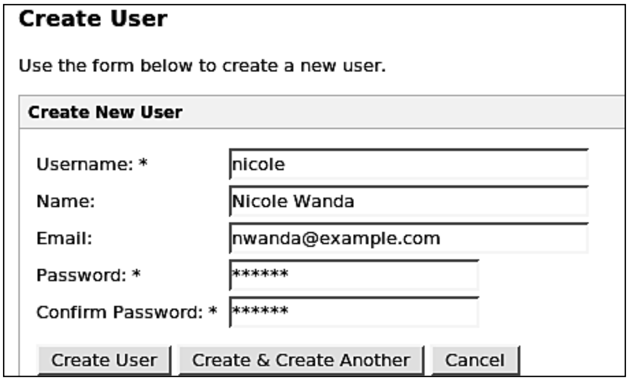
Adding Users

This chapter revolves around the Users/Groups tab. Currently, when you head over to that tab, it will list the default admin user and any other you might have created from within the clients, before we disabled that feature in the last chapter.

Ok, so if you want to add a user, under Users/Groups, you'll notice a self-descriptive **Create New User** option. This takes you to a simple web form (refer to the following screenshot) where you need to enter:

1. **Username** of the new user,
2. His/her real **Name**,
3. **Email** address, and
4. **Password**

You can then either click on the **Create User** button that'll create the user and take you to the user's properties for further editing, or you can use the **Create & Create Another** button which adds the user to the database and clears the form for you to create another user.



Create User

Use the form below to create a new user.

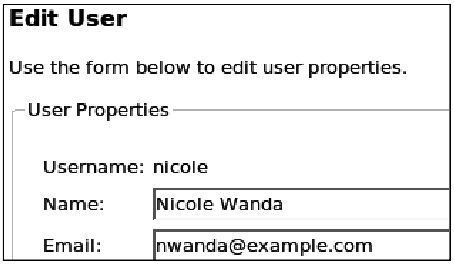
Create New User

Username: *	nicole
Name:	Nicole Wanda
Email:	nwanda@example.com
Password: *	******
Confirm Password: *	******

Note that two users on the network cannot have the same username. Openfire will ask you to change the username if an existing user with that username already exists.

Editing And Deleting Users

Openfire lets you correct a user's name and email address, if you've made a mistake while adding these details. Click on the user you want to edit from the User Summary section. This opens the **User Properties** section (refer to the following screenshot) section, which you might have already visited if you had used the Create User option to add a user. This page lists the details of the user, along with his or her status (Online, Offline, and so on), the date she was registered with Openfire, and the groups she belongs to. We'll come back to this section when this user is more integrated into the system. For now, click the **Edit Properties** button and make the changes in the page that opens. You can also get here by clicking the small button in the **Edit** column in the User Summary page itself.



Edit User

Use the form below to edit user properties.

User Properties

Username:	nicole
Name:	Nicole Wanda
Email:	nwanda@example.com

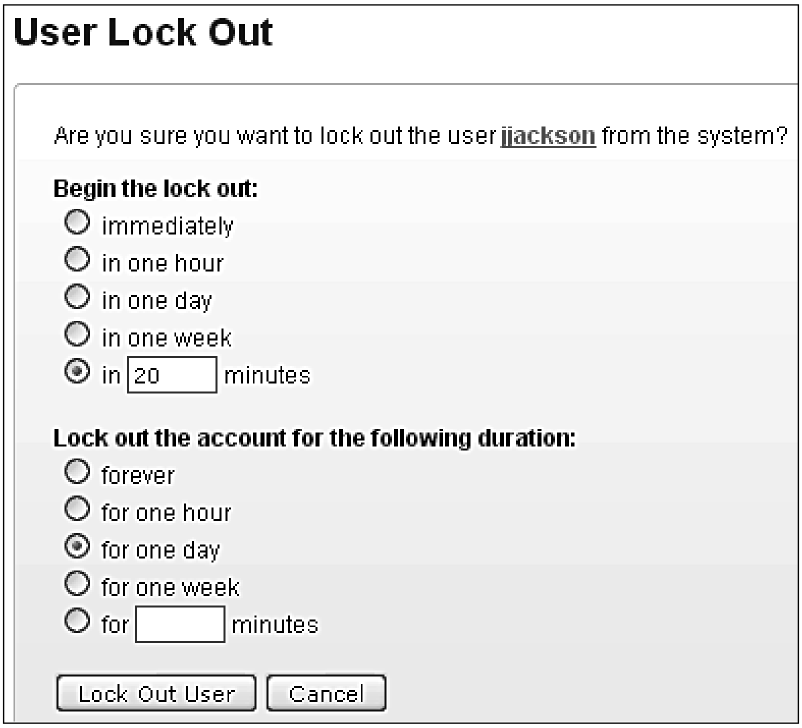
But what if you want to change the unchangeable—say the username. You'll have to take a longer route to change the username, which is to say that you'll have to delete the account and start afresh! In the **User Summary** section, there's a small red button under the **Delete** column. When this button is clicked, Openfire will ask you to confirm your action by clicking the **Delete User** button before actually sending the user to oblivion...or at least their digital self.

Temporarily Suspend Users

There are a number of reasons why you'd want to stop users from logging on to the Openfire chat server. In addition to disciplining users, you can also temporarily suspend accounts of users who are out of the office on tour or on vacation.

To suspend a user, you'll first have to identify the user you wish to "ban". You can do this using the User Summary section under the Users/Groups tab. After homing in on a user, click on his username to bring up the **User Properties** page.

In the left-hand navigation bar, one of the user properties that can be altered is the **Lock Out** option. Click on this to bring up the **User Lock Out** page (refer to the following screenshot), where you can set the lock out mechanism for that particular user.



User Lock Out

Are you sure you want to lock out the user jjackson from the system?

Begin the lock out:

- immediately
- in one hour
- in one day
- in one week
- [✓ in 20 minutes]

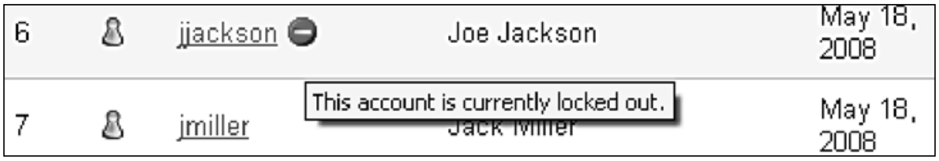
Lock out the account for the following duration:

- forever
- for one hour
- [✓ for one day]
- for one week
- for minutes

You'll be asked to set the following two options:

- Time when the lock out comes into play: Openfire lets you enforce the ban **immediately** (default option) or at a later time. It offers you preset options to ban the user after an hour, after a day, or after a week. If none of these options work for you, Openfire also lets you set a time (in minutes) after which the ban is enforced.
- The duration of the lock out: Once the ban is in effect, how long do you want it to last? By default, Openfire will ban the user forever. But you have the option to automatically un-ban the user after an hour, a day, or a week. Again, if none of these work, you can specify a custom ban duration (in minutes), after which access is restored to that user.

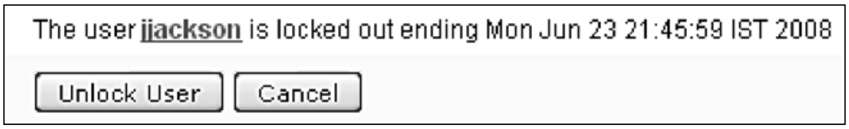
Once a user is locked out of the system, Openfire will mark that user with a red 'forbidden' icon next to the username (refer to the following screenshot) on the User Summary page.



6		jjackson ⛔	Joe Jackson	May 18, 2008
7		jmiller	Jack Miller	May 18, 2008

This account is currently locked out.

To unlock a user manually if you've locked him or her out forever, or if you want to restore access earlier than the duration you've set while banning him or her, you can do so easily.



The user jjackson is locked out ending Mon Jun 23 21:45:59 IST 2008

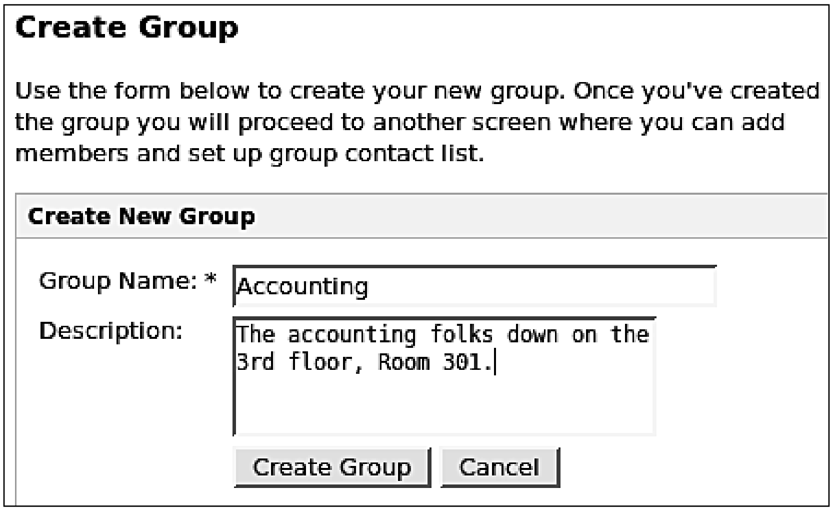
Just retrace your steps to the user's properties page and click on the lock out option as before. Because a lock is already in place, Openfire will inform you that the user is banned and will also give you a calculated date and time at which the user will be allowed back in. To override this preset, click on the **Unlock User** button (refer to the previous screenshot) button to restore access.

Organizing Users into Groups

Using Openfire, you can group users into virtual IM groups. It would make sense to create these groups based on an existing physical or logical basis. "Accounting", "Marketing", "IT", are all good examples of logical groups in an enterprise setting.

It's a good idea to take the pain of organizing users into groups, because Openfire is packed with tools to help members within these groups interact with each other more efficiently. And that's exactly what the doctor ordered, right?

Similarly to how you create a user, to create a group, use the Create New Group link on the **Users/Groups | Groups** tab. This simply requires (a) the name of the group, and (b) a brief description (refer to the following screenshot).



Create Group

Use the form below to create your new group. Once you've created the group you will proceed to another screen where you can add members and set up group contact list.

Create New Group

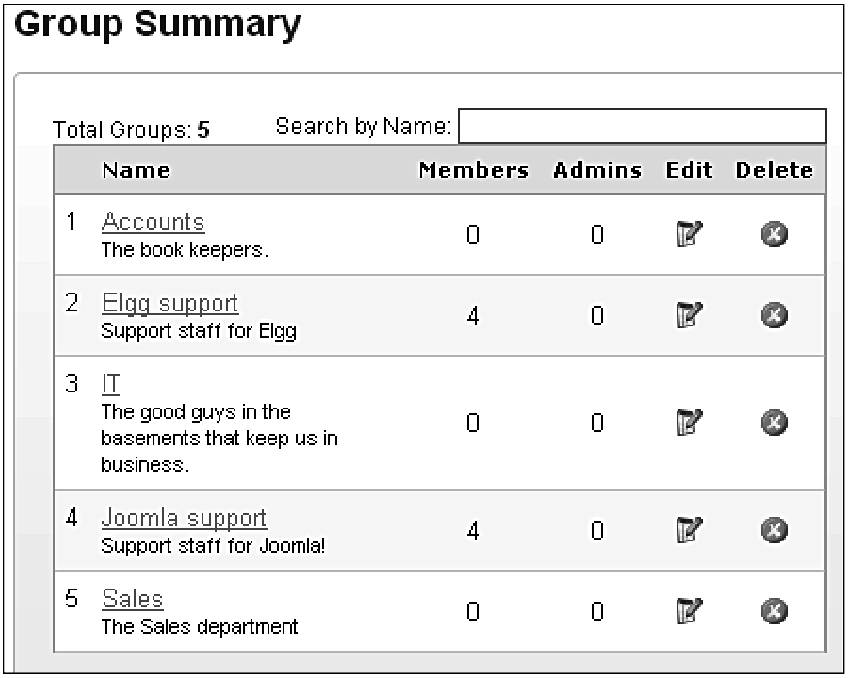
Group Name: *	Accounting
Description:	The accounting folks down on the 3rd floor, Room 301.

Once a group has been created, you are passed to the **Group Options** section where you can add users to this group. To do so, you'll have to specify the username of each and every user you want to add to the group, manually. This might sound tiring, especially if you have a lot of users, but there are easier ways—one we'll cover now and the other in the next chapter.

Editing and Deleting Groups

As with users, Openfire lets you edit and delete groups as well. There are two reasons why you'd want to edit a group. Firstly, and most importantly, you'd want to edit a group to add or remove users from within that group. Secondly, you'd want to edit a group to change its characteristics such as the name and description.

To edit a group head over to the **Users/Groups | Groups** tab, which will display a list of all groups. From the **Group Summary** page (refer to the following screenshot), click on the **Edit** button adjacent to the group you want to edit. This will take you to a page that'll let you add or remove members from the group, and control rosters, which we'll look at in detail in the next section.

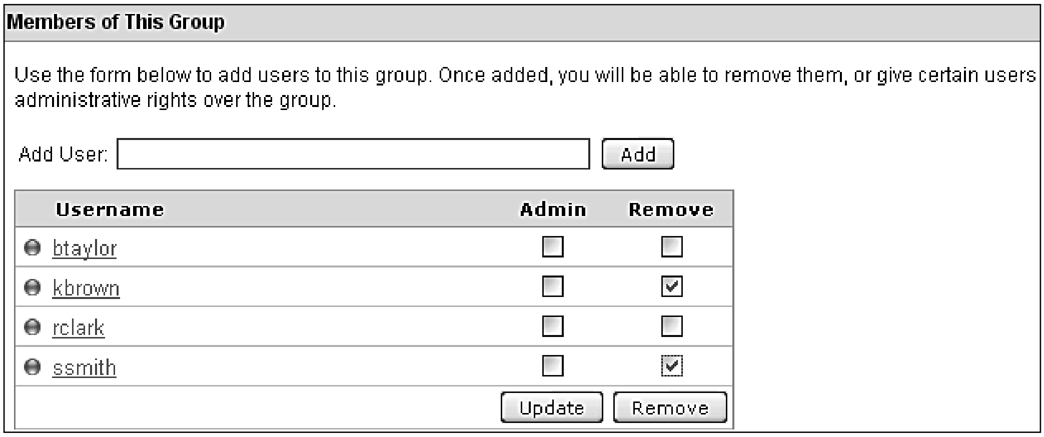


Group Summary

Total Groups: **5** Search by Name:

	Name	Members	Admins	Edit	Delete
1	Accounts The book keepers.	0	0		
2	Elgg support Support staff for Elgg	4	0		
3	IT The good guys in the basements that keep us in business.	0	0		
4	Joomla support Support staff for Joomla!	4	0		
5	Sales The Sales department	0	0		

The procedure to add users to a group from this page, is the same as it was back when you first created a group—you'll have to manually specify the usernames of users you want to add to the group. To remove users, select the checkbox adjacent to the user's name under the **Remove** column and click on the **Remove** button (refer to the following screenshot).



Members of This Group

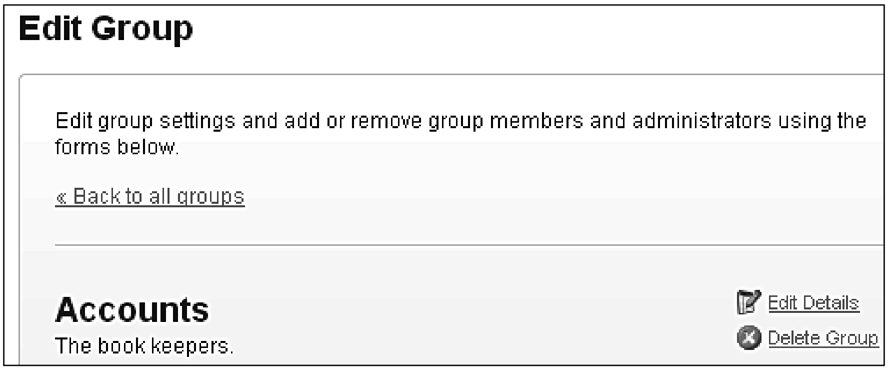
Use the form below to add users to this group. Once added, you will be able to remove them, or give certain users administrative rights over the group.

Add User: [ ] Add

Username	Admin	Remove
btaylor	☐	☐
kbrown	☐	✓
rclark	☐	☐
ssmith	☐	✓

Update Remove

While you are on this page, you'll also notice two links adjacent to the name of the group—**Edit Details** and **Delete Group** (refer to the following screenshot). With **Edit Details**, you can change the name of the group and its description, while the **Delete Group** link simply removes the group from the system, after confirming that this is what you want to do.



Edit Group

Edit group settings and add or remove group members and administrators using the forms below.

« Back to all groups

Accounts

The book keepers.

Edit Details

Delete Group

Note that users in the group that has been deleted are not removed from the system.

Pre-populating Rosters

A roster, in XMPP-speak, is your friends list. Now imagine that you log in to your IM network for the very first time and everyone you know is already on your contact list, neatly arranged into different groups such as 'Friends', 'Office', and so on. That's the magic of pre-populating rosters.

To get the most out of this section, I'll show you a couple of examples that you will be able to identify with. Let's picture a typical office scenario with several users arranged into three groups:

Accounting

James, Nicole, Rob, Dale, Carla

IT

Jimmy, Rick

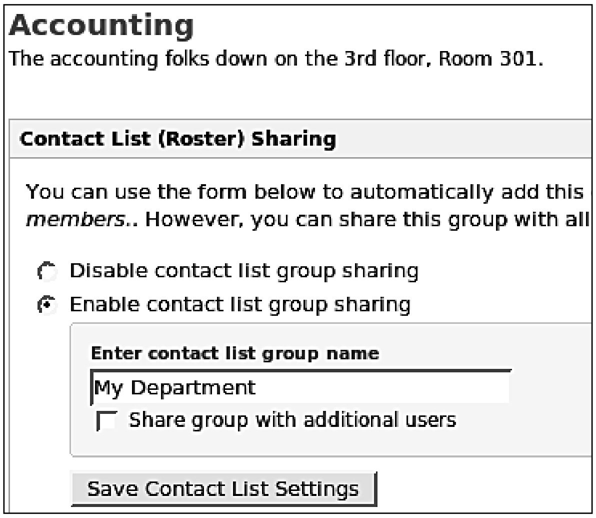
Board

Henry, Peter, Sally

Now let's use these users and groups in our hypothetical (but very practical) examples.

Scenario #1: See The Members of Your Department

This is the most logical course of action for a group. The first people you'd want to be connected with are the co-workers in your department. On the Group Summary page, click on any group. In the page that opens you'll notice the **Contact List (Roster) Sharing** section. Select the **Enable contact list group sharing** option button. This further opens up a text box where you need to enter the name of the group as you want it to appear in your IM program. To keep things simple, you can use your department name or something like **My Department** (refer to the following screenshot). Click the **Save Contact List Settings** button and you're done!



Accounting

The accounting folks down on the 3rd floor, Room 301.

Contact List (Roster) Sharing

You can use the form below to automatically add this
members.. However, you can share this group with all

Disable contact list group sharing [✓ Enable contact list group sharing]

Enter contact list group name

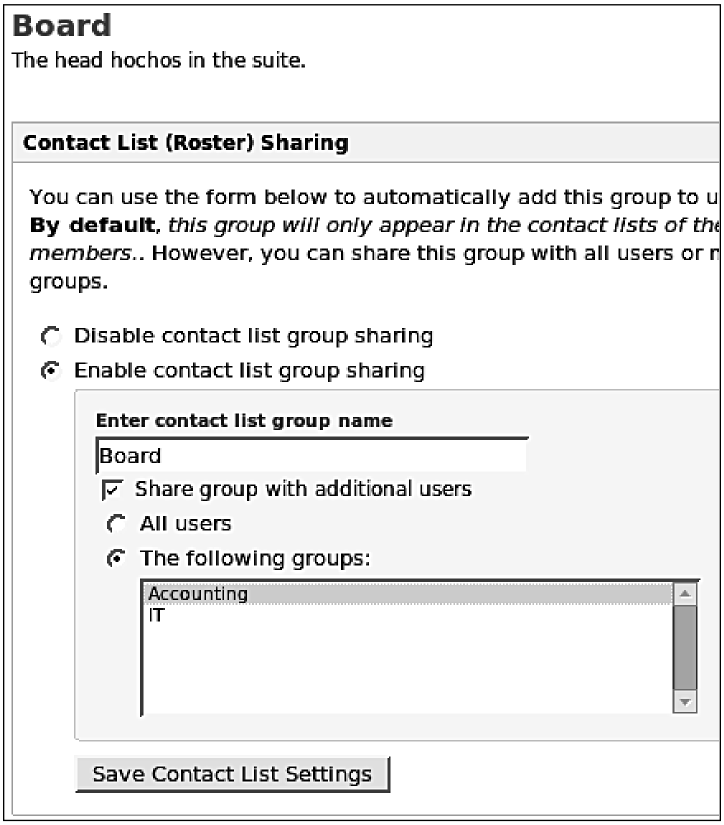
My Department

☐ Share group with additional users

When the people of that department log into their IM clients, they'll see everyone else from their department already in their contact list. Now repeat this step with all groups/departments.

Scenario #2: Accounting Reports to The Board

In addition to being connected with everyone in your department, a department would like to be connected with individuals or a group of individuals they report to. So if our Accounting group reports to the Board, we, follow the procedure above—open the **Board** group (refer to the following screenshot), select the **Enable contact list group sharing** option button, and enter a name for the group. Now, select the **Share group with additional users** checkbox which expands the space to give you a couple of additional options. Select the option to share the group with **The following groups** and select the **Accounting** group from the text box beneath the option. Voila! Now the Accounting group sees each other and the Board it reports too.


Board

The head hochos in the suite.

Contact List (Roster) Sharing

You can use the form below to automatically add this group to u
By default, *this group will only appear in the contact lists of th*
members.. However, you can share this group with all users or r
groups.

Disable contact list group sharing [✓ Enable contact list group sharing]

Enter contact list group name: Board

[✓] Share group with additional users

All users [✓ The following groups:]

Accounting (selected)
IT

To allow the board to see members of the Accounting group, you'll have to reverse the process and make sure that the Accounting group is also shared with the Board group. If there are various groups that would like to stay in touch with each other, for example Sales, Marketing, and Accounting, you can choose to share a group with multiple groups. Just hold down the *Ctrl* key while making the selection when sharing groups with additional groups.

Scenario #3: Everyone Wants IT on Their List

Yeah, right! Everyone wants the good guys to be on their contact list. That's easily done. Edit the IT group, enable contact list group sharing, specify a name, choose to share group with additional users, and select the **All users** option. Now everyone will have IT on their contact list.

On the other hand, if IT wants everyone on their list, make sure every group is shared with IT, in addition to the other group or groups with which they are already connected.

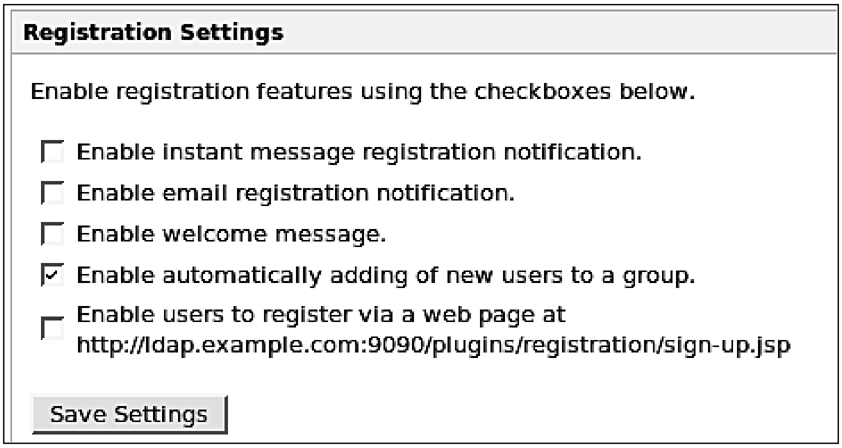
Scenario #4: Everyone on Everyone's List... Automatically

This is a bonus trick for small organizations that want everyone to be connected to everyone else. All you really need to do is modify the trick above, but this would be too laborious. What we're going to do differently is that we'll do it in a way that saves us the effort of manually adding every user to all groups. We'll add new users to everyone's rosters automagically!

First, create a new group called "Everyone" or "All users" or anything you feel like. Make sure you share this group with all users. Then download the Registration plugin (**Plugins | Available Plugins**). Now when you head back to the **Users/Groups** tab, there will be a new **Registration Properties** section.

[The Registration plugin comes into play when a new user is registered in the system. Depending on how it has been configured, the plugin then performs certain actions, such as displaying a welcome message, or sending an email notification to a particular email address.]

The registration property we are interested in is called the **Default Group** and is listed close to the bottom of the page. In the text field provided, enter the name of the group we just created and click on the **Save Group** button. Then head back to the top of the page and select the **Enable automatically adding of new users to a group** checkbox and click on **Save Settings** (refer to the following screenshot).



Registration Settings

Enable registration features using the checkboxes below.

- [ ] Enable instant message registration notification.
- [ ] Enable email registration notification.
- [ ] Enable welcome message.
- [x] Enable automatically adding of new users to a group.
- [ ] Enable users to register via a web page at http://ldap.example.com:9090/plugins/registration/sign-up.jsp

From this point on, whenever a new user is added to the system, the plugin will make sure that he or she gets listed under the group we specified. Because we've also configured the group to be shared with all users, the user will be automatically added to everyone's rosters!

Summary

In this chapter we added the last piece of our IM puzzle—users! And boy, we did it in style! We organized our users in groups, which we then used to populate the user's contact list in a variety of ways. Thanks to our efforts, all that our users need to do is just log on to the network and they'll be instantly connected with everyone in the organization, or more realistically, with everyone in their department, or to the department or person they report to, or to the good, ever-friendly IT guys.

Now here's the cherry on the cake. If you have a directory service running on your network, Openfire is quite capable of integrating with it. It'll save you a lot more than just time spent on adding users. Fasten your seat belts; we're just getting started.

5
Hooking up With a Directory Service

In the last chapter, we looked at how we could add users to the Openfire IM network, organize them into groups, and then pre-populate rosters. But if you have a huge network, adding all of the users manually is far from an optimal solution. Most networks these days have a directory service to store and organize information about their network's users and resources. This provides network administrators with a centralized mechanism to manage their users' access to the resources. It's a walk-in-the-park to configure Openfire to sync up with a directory service for authentication purposes and more!

There are various implementations of the Directory Service, including Apache Directory Server, Apple's Open Directory, Fedora Directory Server, OpenLDAP, and Microsoft's Active Directory. Networks often rely on the Lightweight Directory Access Protocol (LDAP) for authenticating users using their directory username and password. Openfire can outsource authentication to a directory service such as OpenLDAP and Active Directory. We'll cover the following in this chapter:

- Hooking Openfire with OpenLDAP
- Hooking Openfire with Active Directory
- Editing the Openfire config file

Remember that once you've integrated Openfire with a directory service, you won't be able to add users or groups from Openfire's administration console. Whatever changes are required will have to be done in the directory itself. These changes will then automatically be reflected in Openfire.

Adding Users Via OpenLDAP

As you may recall, when we first launched the administration console after installing the server, Openfire asked us whether we wanted to use a directory server. And that's the place where we can specify our directory server settings. But I am not going to undo all that we've done up to now and reinstall Openfire. I said Openfire is idiot-proof, and it is! Just tweak a simple variable and you can easily re-run the initial setup.

First, stop the Openfire server, if it's running. Then, open a terminal and navigate to the Openfire server installation directory. Navigate to the `conf/` directory, and open the "openfire.xml" file in your favorite text editor. Look for a line that reads "<setup>true</setup>" and change it to "<setup>false</setup>". Save the file, restart the server, and head back to the administration console. Presto! You're back at the setup interface.

Before we get into to setting up Openfire with OpenLDAP, it's a good idea to keep the following directory server settings handy:

- Host and Port: The IP address of the directory server and the port it's running on.
- BaseDN: This is the top-level, distinguished name of the LDAP directory tree. Check your LDAP configuration for this.
- Authentication information: You'll require the admin's complete DN and his or her password.

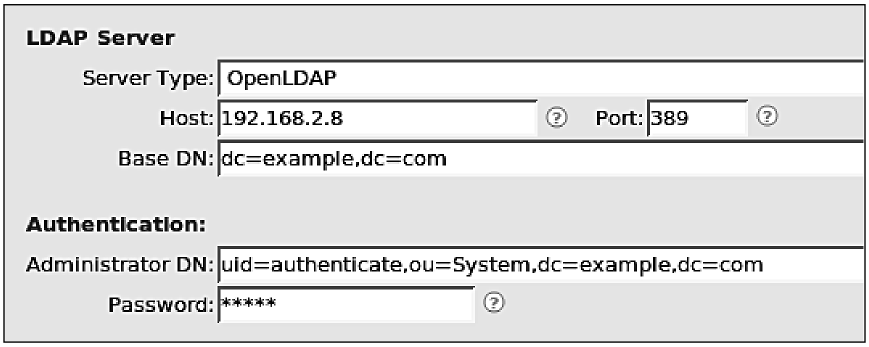
[

Installing OpenLDAP isn't difficult, and is well documented.
There are several books on the topic including Packt's Mastering OpenLDAP: Configuring, Securing and Integrating Directory Services (see `http://www.packtpub.com/OpenLDAP-Developers-Server-Open-Source-Linux/book`). You'll find easy-to-follow documentation for installing OpenLDAP on your favorite Linux distribution. A general installation guide that I recommend is available at Linux (see `http://www.linux.com/articles/113607`).

]

In the setup interface, continue with the options we used earlier until you get to **Profile Settings**. Unlike in our previous attempt, this time around, choose the **Directory Server** option and click **Continue**. This takes us to a three-step process at the end of which we will have synchronized our OpenLDAP server with Openfire.

1. In the first step of the process, you have to:
    - Select the type of directory server from the pull-down menu (choose OpenLDAP)
    - Enter the IP address and port of the directory server (on my network OpenLDAP is running at 192.168.2.8 via its default port of 389)
    - Enter the **Base DN** (**dc=example,dc=com**)
    - Enter the **Administrator DN** (**uid=authenticate,ou=System,dc=example,dc=com**)
    - Enter the password



LDAP Server

Server Type:	OpenLDAP
Host:	192.168.2.8
Port:	389
Base DN:	dc=example,dc=com

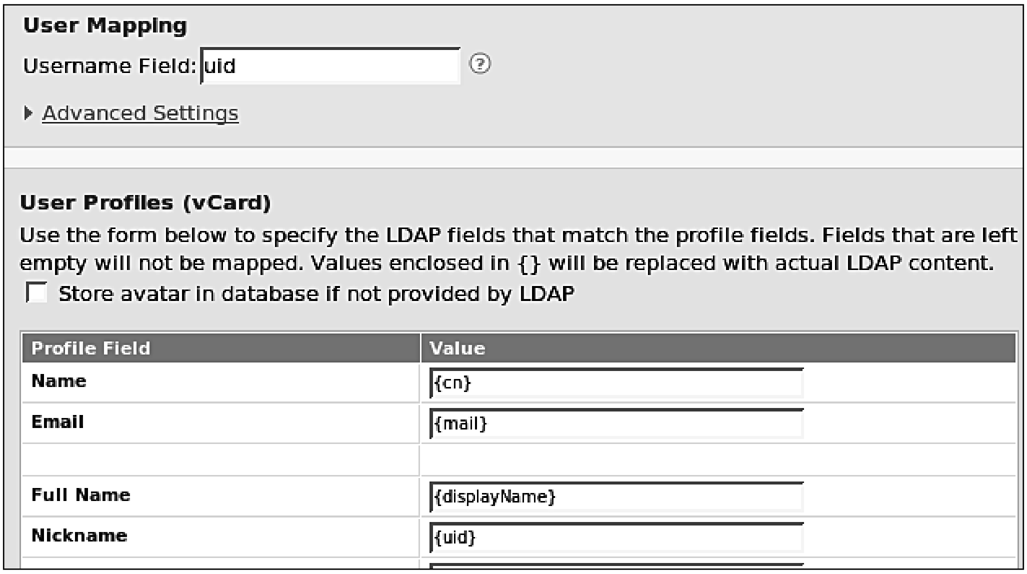
Authentication:

Administrator DN:	uid=authenticate,ou=System,dc=example,dc=com
Password:	*****

Once you've entered these settings, use the **Test Settings** button to make sure that the settings are correct. When you test your settings, Openfire tries to establish a connection with your OpenLDAP server based on the settings you've provided. If the settings are correct, Openfire will report that the test was a success. Close the test result pop-up window and move on to the next step.

2. The next step is a bonus step. Openfire already has all that it needs to connect to the OpenLDAP server. But because users already have their contact information and other details in the OpenLDAP database, which they'll also need in their IM profile, why not use that data as well? Openfire lets you map a user's information from their directory into their IM profiles! Now, how cool is that?

   So, start the step by filling in the mandatory **Username Field**. This will tell Openfire how to identify usernames for the other bits of information in the directory server. Leaving it set to **uid** should work for most OpenLDAP directory setups. In the **User Mapping** section, enter the values you'd specified when creating the user in the LDIF record.



User Mapping

Username Field: uid

Advanced Settings

User Profiles (vCard)

Use the form below to specify the LDAP fields that match the profile fields. Fields that are left empty will not be mapped. Values enclosed in {} will be replaced with actual LDAP content.

☐ Store avatar in database if not provided by LDAP

Profile Field	Value
Name	{cn}
Email	{mail}
Full Name	{displayName}
Nickname	{uid}

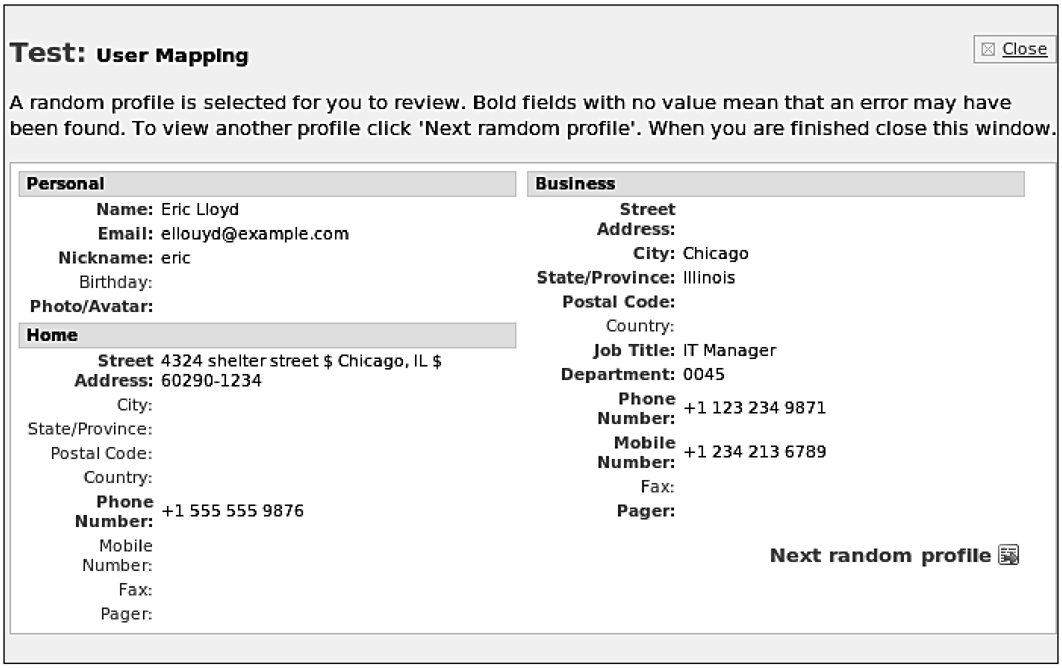
Here's a sample LDIF file that I used to initially populate my OpenLDAP:

```
# This is the root of the directory tree
dn: dc=example,dc=com
description: Example.Com, your trusted non-existent corporation.
dc: example
o: Example.Com
objectClass: top
objectClass: dcObject
objectClass: organization

# Subtree for users
```

```
dn: ou=Users,dc=example,dc=com
ou: Users
description: Example.Com Users
objectClass: organizationalUnit

##
## USERS
##
# Eric Lloyd
dn: uid=eric,ou=Users,dc=example,dc=com
ou: Users
# Name info:
uid: eric
cn: Eric Lloyd
sn: Lloyd
givenName: Eric
displayName: Eric Lloyd
# Work Info:
title: IT Manager
description: In-charge of IT for Example.Com
employeeType: Employee
departmentNumber: 0045
employeeNumber: 11-32-0045
mail: ellouyd@example.com
roomNumber: 3A
telephoneNumber: +1 123 234 9871
mobile: +1 234 213 6789
st: Illinois
l: Chicago
street: 1234 Cicero Ave.
# Home Info:
homePhone: +1 555 555 9876
homePostalAddress: 4324 shelter street $ Chicago, IL $ 60290-1234
# Misc:
userPassword: {SSHA}kOumZM4LZcmE0FKweaiy/EqCnLxFlwOe
preferredLanguage: en-us,en-gb
# Object Classes:
```

objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson

Use the values within curly braces {} in this file in the appropriate fields in the web form. For example, in the **Full Name** field on the form, enter {displayName} as mentioned in the sample LDIF file. This will map the user's full name into his profile. You can ask Openfire to test these values by fetching a sample record from the OpenLDAP directory.



Test: User Mapping

Close

A random profile is selected for you to review. Bold fields with no value mean that an error may have been found. To view another profile click 'Next ramdom profile'. When you are finished close this window.

Personal	
Name:	Eric Lloyd
Email:	ellouyd@example.com
Nickname:	eric
Birthday:	
Photo/Avatar:	
Home	
Street Address:	4324 shelter street $ Chicago, IL $ 60290-1234
City:	
State/Province:	
Postal Code:	
Country:	
Phone Number:	+1 555 555 9876
Mobile Number:	
Fax:	
Pager:	

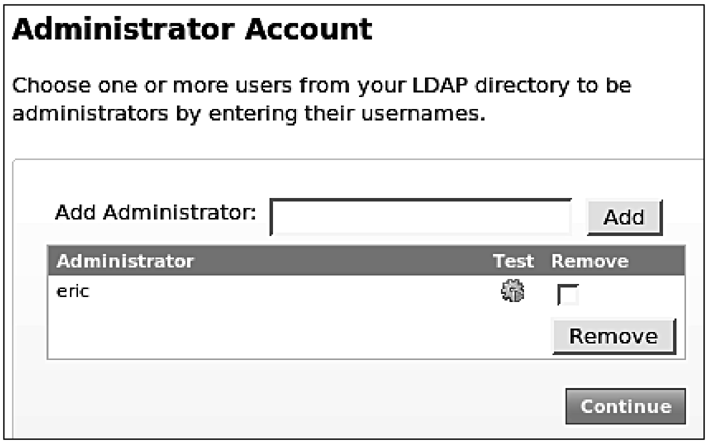
Business	
Street Address:	
City:	Chicago
State/Province:	Illinois
Postal Code:	
Country:	
Job Title:	IT Manager
Department:	0045
Phone Number:	+1 123 234 9871
Mobile Number:	+1 234 213 6789
Fax:	
Pager:	

Next random profile

Incorrect and mismatched values and empty fields are not mapped. When you're satisfied with the output, click on **Save and Continue**.

3. The last step of the OpenLDAP integration process involves instructing Openfire how to map groups from OpenLDAP. Yes, groups! Of course groups, like users, are defined in the LDIF format when adding users. Again, the **Test Settings** option will give you a sample output of the groups as mapped from the OpenLDAP directory.

And we're done... At least with the "hooking up" part. If you notice, we're at the last **Administrator Account** step of the Openfire setup process. But unlike the last time, when we had to specify a password for the administrator and his email address, all we need to do now is to specify the username of the lucky user who gets to play administrator! That's because, Openfire is now aware of all the users in the organization, their passwords, and their email addresses.


Administrator Account

Choose one or more users from your LDAP directory to be administrators by entering their usernames.

Add Administrator: [ ] Add

Administrator	Test	Remove
eric		☐

Once you've connected OpenLDAP to Openfire, head over to the **User/Groups** tab. Instead of the solitary **admin** user that we saw when we first set up Openfire, we see each and every member of the organization! Similarly, the **Group Summary** tab lists all of our groups, along with their descriptions and members. Almost makes you wonder, "who dunnit"?

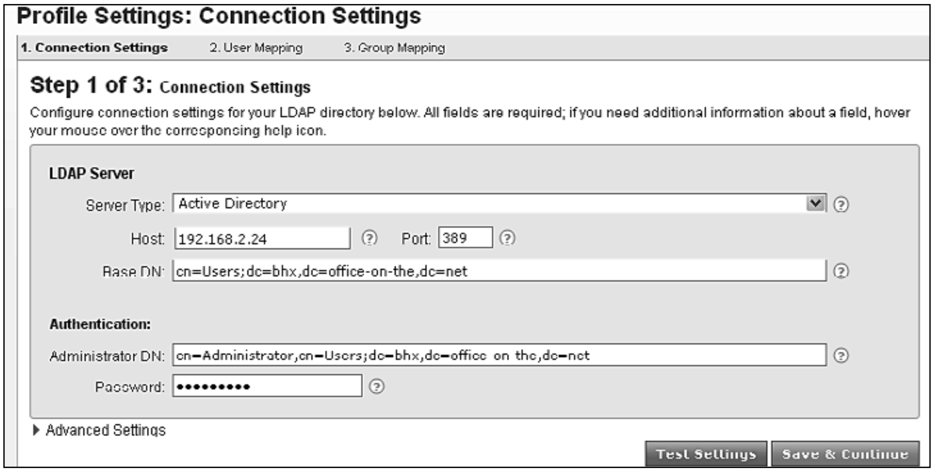
Adding Users via Active Directory

If you are running Microsoft Windows Server, you are probably using Microsoft's Active Directory service. No sweat! Adding users via Active Directory isn't a wee bit different from adding them via OpenLDAP.

As with OpenLDAP, setting up an Active Directory server on Microsoft Windows Server 2000/2003 is pretty straightforward. Here is a step-by-step guide to setting up Active Directory (see `http://web2.blogtells.com/2008/04/28/installing-active-directory-on-a-windows-server-2003-computer/`).

Let's quickly go over the steps for hooking up Openfire with OpenLDAP as described above, but this time we'll connect it with Active Directory.

1. Stop Openfire, if its running, and edit **openfire.xml**. To rerun the setup, look for the line that reads **<setup>true</setup>** and change it to **<setup>false</setup>**. Skip this step if you're installing a new Openfire server.
2. Continue with the Openfire installation as described in the opening chapters, until you get to the **Profile Settings** step for adding users to Openfire. Here, choose to use a **Directory Server** and select **Active Directory** from the drop-down list.
3. Enter your Active Directory server details in the space provided. You need the following information:
   - Host Name or IP address of the server running the directory server.
   - The Port on which the directory server is running, which is usually 389.
   - The **Base DN** guides Openfire to the location where the directory server keeps your users list. If your domain is **bhx.office-on-the.net**, and your users are under the **Users** folder, the Base DN will be **cn=Users;dc=bhx,dc=office-on-the,dc=net**.
   - **Administration DN** is the location of the administrator user. If this user is named **Administrator,** and is under the **Users** folder, the **Administration DN** will be **cn=Administrator,cn=Users;dc=bhx,dc=office-on-the,dc=net**.
   - **Password** is the password for the user specified in the **Administrator DN** field.


Profile Settings: Connection Settings

1. Connection Settings 2. User Mapping 3. Group Mapping

Step 1 of 3: Connection Settings

Configure connection settings for your LDAP directory below. All fields are required; if you need additional information about a field, hover your mouse over the corresponding help icon.

LDAP Server

Server Type: Active Directory

Host: 192.168.2.24 Port: 389

Base DN: cn=Users;dc=bhx,dc=office-on-the,dc=net

Authentication:

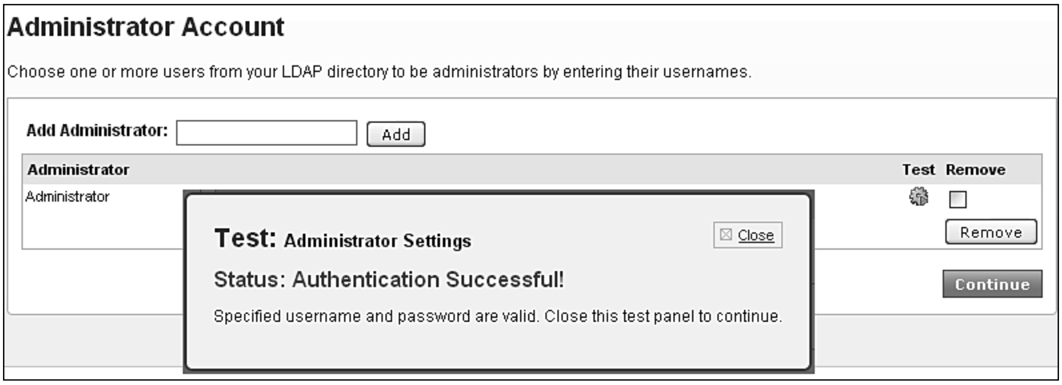
Administrator DN: cn=Administrator,cn=Users;dc=bhx,dc=office-on-the,dc=net

Password: •••••••••

Once the settings have been entered, use the **Test Settings** button to let Openfire establish a link with the Active Directory server. Once it's successful, continue with the next step.

1. In the **User Mapping** step, configure how the users are mapped from the directory server. Information provided within curly braces {} are replaced by real content from the directory server.
2. The last step of the hooking-up process is to map users from the directory server. To test if groups are mapped correctly, you can ask Openfire to display a list of groups as it reads from the Active Directory. The image that follows shows a list of default Windows groups as mapped by Openfire.

This completes the process of hooking up Active Directory with Openfire. Because we can't add users to Openfire now, we'll have to specify one of the existing users as an administrator.


Administrator Account

Choose one or more users from your LDAP directory to be administrators by entering their usernames.

Add Administrator:

Administrator	Test	Remove
Administrator		

Test: Administrator Settings

Status: Authentication Successful!

Specified username and password are valid. Close this test panel to continue.

In most circumstances, the administrator for both the directory server and Openfire will be the same person (you!). You can safely select the default Windows administrator user as the Openfire administrator as well.

Editing the config File

Behind the scenes, all Directory Server settings are stored in the `openfire.xml` file found in the `/conf` directory. Although it's absolutely fine to use the web-based user interface provided by Openfire's setup, you can also add these settings directly in the Openfire config file. This is especially useful when there's a change in your Directory Server settings and you don't want to re-run the whole Openfire setup.

Here's a section of the **openfire.xml** file pertaining to the directory server.

```
<ldap>
    <host>192.168.2.24</host>
    <port>389</port>
    <baseDN>cn=Users;dc=bhx,dc=office-on-the,dc=net</baseDN>
    <adminDN>cn=Administrator,cn=Users;dc=bhx,
                dc=office-on-the,dc=net</adminDN>
    <adminPassword>something-more-complex-that-this</adminPassword>
    <usernameField>sAMAccountName</usernameField>
    <searchFilter>(objectClass=organizationalPerson)</searchFilter>
<vcard-mapping><![CDATA[
<vCard xmlns="vcard-temp">
  <N>
    <GIVEN>{cn}</GIVEN>
  </N>
  <EMAIL>
    <USERID>{mail}</USERID>
  </EMAIL>
  <FN>{displayName}</FN>
  <PHOTO>
    <TYPE>image/jpeg</TYPE>
    <BINVAL>{jpegPhoto}</BINVAL>
  </PHOTO>
  <ADR>
    <WORK/>
    <STREET>{streetAddress}</STREET>
    <LOCALITY>{l}</LOCALITY>
    <REGION>{st}</REGION>
    <PCODE>{postalCode}</PCODE>
    <CTRY>{co}</CTRY>
  </ADR>
  <TEL>
    <WORK/>
    <VOICE/>
    <NUMBER>{telephoneNumber}</NUMBER>
  </TEL>
  <TITLE>{title}</TITLE>
  <ORG>
    <ORGUNIT>{department}</ORGUNIT>
  </ORG>
</vCard>]]></vcard-mapping>
```

```
        <nameField>cn</nameField>
        <emailField>mail</emailField>
        <groupNameField>cn</groupNameField>
        <groupMemberField>member</groupMemberField>
        <groupDescriptionField>description</groupDescriptionField>
        <posixMode>false</posixMode>
        <groupSearchFilter>(objectClass=group)</groupSearchFilter>
    </ldap>
    <provider>
        <vcard>
          <className>org.jivesoftware.openfire.ldap.LdapVCardProvider
           </className>
        </vcard>
        <user>
          <className>org.jivesoftware.openfire.ldap.LdapUserProvider
           </className>
        </user>
        <auth>
          <className>org.jivesoftware.openfire.ldap.LdapAuthProvider
           </className>
        </auth>
        <group>
          <className>org.jivesoftware.openfire.ldap.LdapGroupProvider
           </className>
        </group>
    </provider>
```

The tags in the file are self explanatory. In between the `<ldap>` and `</ldap>` tags we first provide the connection details as shown in the examples above. We then provide the user details and VCard settings that Openfire maps from the directory server. In between the `<provider>` and `</provider>` tags, we list the Openfire classes that provide the VCard, User, Authentication, and Group services.

Stop for a while and take a deep breath. Not only have we saved time and effort that would have been wasted if we'd had to manually add all our users to the network, we have also drafted them into groups! All we need to do now is populate rosters to share them as we want, and we're in business. Three cheers to the directory service!

Summary

In this chapter, we acted smart. We decided to let our directory service step in and validate our users. If you have your directory service in order, it's just a three-step process to sync it with Openfire. The cherry on top is that Openfire will also let you migrate the users' contact information and other details, and use it to populate their profiles, as well as create groups based on the directory service information. Pre-populating groups is just a matter of mapping the intra-group relationships.

Of course, now that we have users on our network, we'll see how we can manage them efficiently, in the next chapter.

6
Effectively Managing Users

The difference between a hassled always-on-the-job admin and an admin who has plenty of time to dress for a lovely date in the evening, is how effectively they manage their users. As any admin knows, managing users is one thing, but managing users effectively takes something special—from both you and the server software. In this chapter, I'll show you some tips and tricks that you can use to become a happy-go-lucky system admin. But finding a beautiful date is beyond the scope of this book!

Despite the way it sounds, managing users isn't an all-involving activity—at least it shouldn't be. Most system administrators tend to follow the "install-it-forget-it" methodology to running their servers. You can do so with Openfire as well, but with a user-centeric service such as an IM server, keeping track of things isn't a bad idea. Openfire makes your job easier with its web-based admin interface.

There are several things that you can setup via the web interface that'll help you manage the users. You can install some plugins that'll help you run and manage the server more effectively, such as the plugin for importing/exporting users, and dual-benefit plugins such as the search plugin, which help users find other users in the network, and also let you check up on users using the IM service.

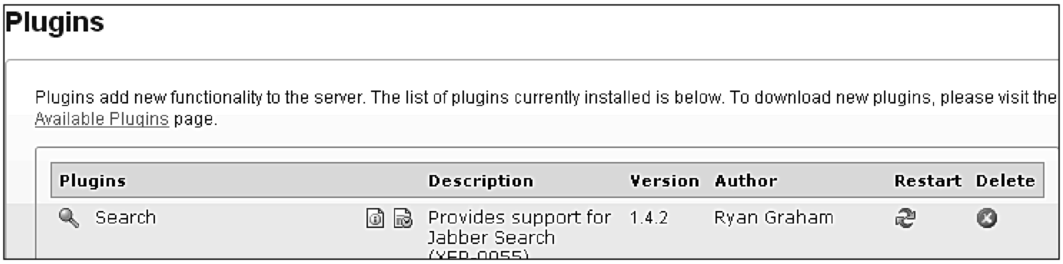
In this chapter, we will cover:

- Searching for users
- Getting email alerts via IM
- Broadcasting messages to all users
- Managing user clients
- Importing/exporting users

Searching for Users with the Search Plugin

Irrespective of whether you have pre-populated user rosters, letting users find other users on the network is always a good idea. The Search Plugin works both ways—it helps your users find each other, and also helps you, the administrator, to find users and modify their settings if required.

To install the plugin, head over to the **Plugins** tab (refer to the following screenshot). The **Search** plugin is automatically installed along with Openfire, and will be listed as a plugin that is already installed. It's still a good idea to restart the plugin just to make sure that everything's a-ok. Locate and click the icon in the **Restart** column that corresponds to the **Search** plugin. This should restart the plugin.



Plugins

Plugins add new functionality to the server. The list of plugins currently installed is below. To download new plugins, please visit the Available Plugins page.

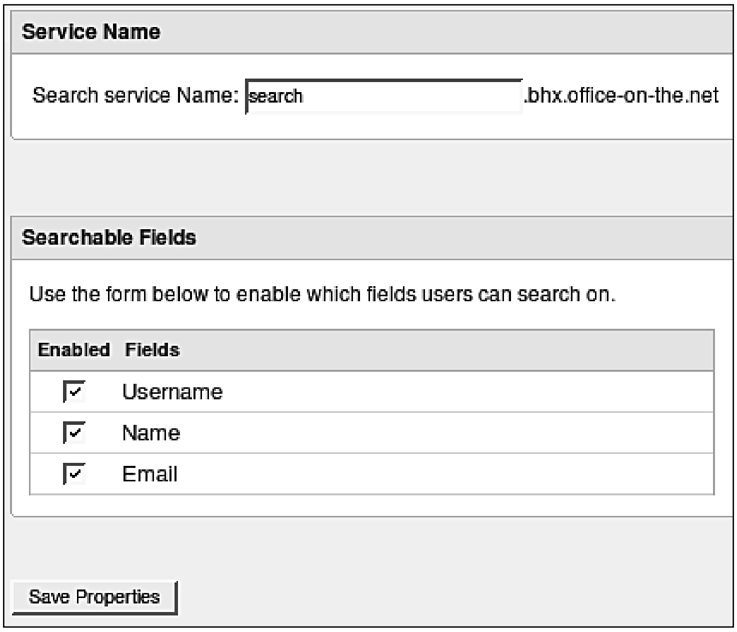
Plugins	Description	Version	Author	Restart	Delete
Search	Provides support for Jabber Search (XEP-0055)	1.4.2	Ryan Graham		

The Search plugin has various configurable options, but by default the plugin is deployed with all of its features enabled. So your users can immediately start searching for users.

To tweak the Search plugin options, head over to the **Server | Server Settings | Search Service Properties** in the Openfire admin interface. From this page, you can enable or disable the service. Once enabled, users will be able to search for other users on the network from their clients. Not all clients have the Search feature but Spark, Exodus, Psi, and some others do.

> Even if you disable this plugin, you, the admin, will still be able to search for users from the Openfire admin interface as described in the following section.

In addition to enabling the Search option, you'll have to name it. The plugin is offered as a network "service" to the users. The Openfire server offers other services and also includes the group chat feature which we will discuss in the Appendix. Calling the search service by its default name, search.<your-domain-name> is a good idea. You should only change it if you have another service on your network with the same name.



Service Name

Search service Name: search .bhx.office-on-the.net

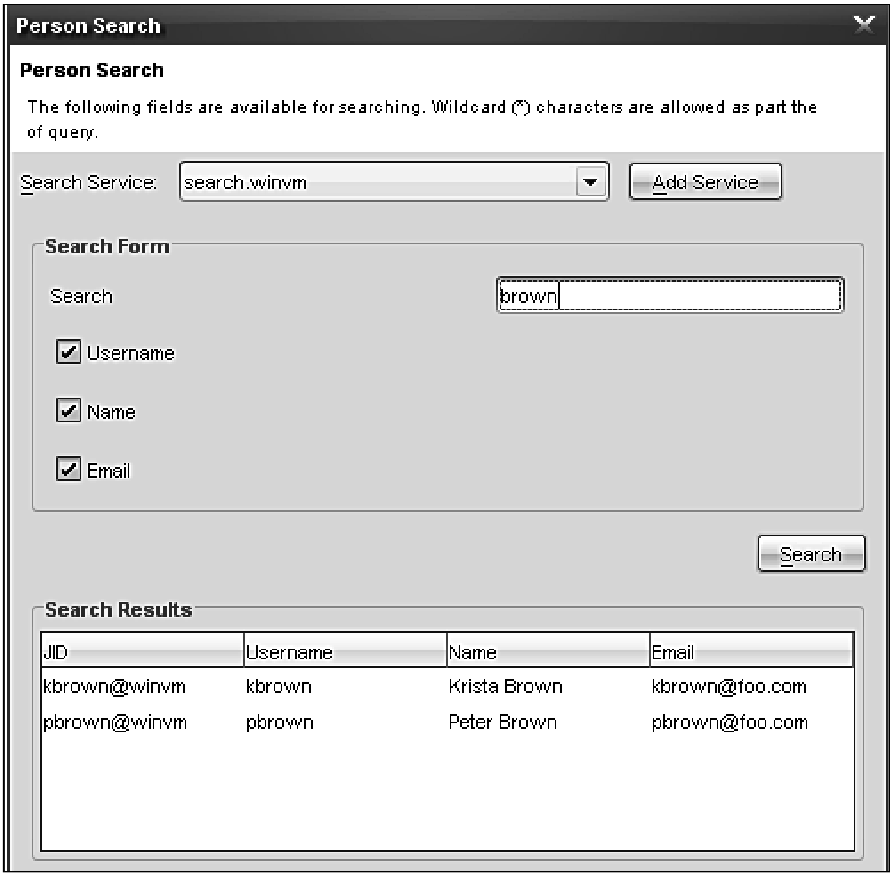
Searchable Fields

Use the form below to enable which fields users can search on.

Enabled	Fields
✓	Username
✓	Name
✓	Email

Finally, you'll have to select the fields users can search on. The three options available are **Username**, **Name**, and **Email** (refer to the previous screenshot). You can enable any of these options, or all the three for a better success rate. Once you're done with setting up the options, click the **Save Properties** button to apply them.

To use the plugin, your users will have to use their clients to query the Openfire server and then select the search service from the ones listed. This will present them with a search interface through which they'll be able to search for their peers (refer to the following screenshot) using one or more of the three options (**Username**, **Name**, **Email**), depending on what you have enabled.


Person Search
Person Search
The following fields are available for searching. Wildcard (*) characters are allowed as part the of query.
Search Service:
Add Service
Search Form
Search
Search Results

Search Service: search.winvm

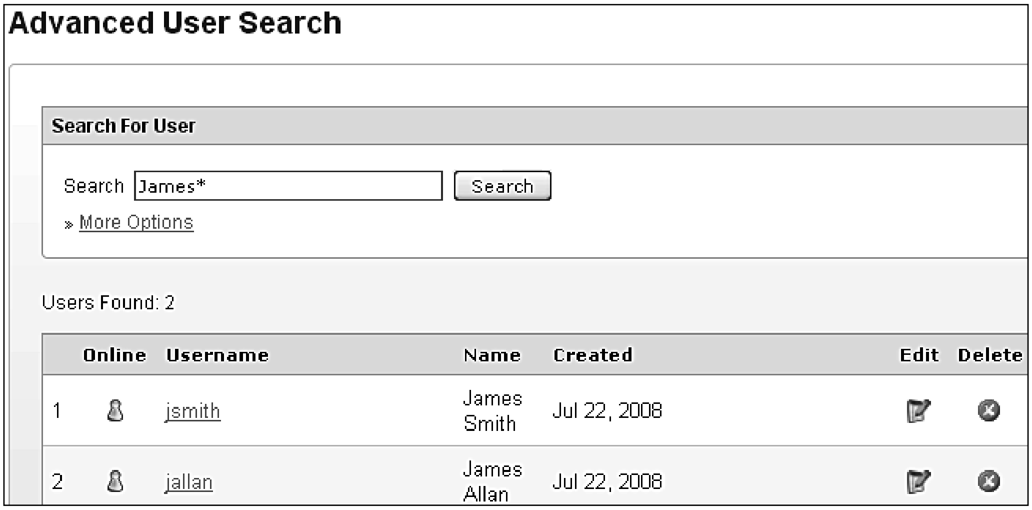
Search: brown

- [✓] Username
- [✓] Name
- [✓] Email

JID	Username	Name	Email
kbrown@winvm	kbrown	Krista Brown	kbrown@foo.com
pbrown@winvm	pbrown	Peter Brown	pbrown@foo.com

Searching for Users from Within The Admin Interface

So we've let our users look for their peers, but how do you, the Openfire admin, look for users? You too can use your client, but it's better to do it from the interface since you can tweak the user's settings from there as well. To search for users from within the admin interface, head over to the Users/Groups tab. You'll notice an **Advanced User Search** option in the sidebar.



Advanced User Search

Search For User

Search: James*

	Online	Username	Name	Created	Edit	Delete
1		jsmith	James Smith	Jul 22, 2008		
2		jallan	James Allan	Jul 22, 2008		

Users Found: 2

When you click on this option, you'll be presented with a single text field with three checkboxes (refer to the previous screenshot). In the textfield, enter the user's **Name**, **Username**, and **Email** that you want to find. The plugin can also handle the * wildcard character so that you can search using a part of the user's details as well. For example, if you want to find a user "James", but don't know if his last name is spelled "Allen" or "Allan", try entering "James A*" in the search field and make sure that the **Name** checkbox is selected. Another example would be "* Smith", which looks for all the users with the last name "Smith".

The search box is case-sensitive.

So why were you looking for "James Allan", the guy with two first names? It was because his last name is in fact "Allen" and he wants to get it corrected. So you find his record with the plugin and click on his username. This brings up a summary of his properties including his status, the groups he belongs to, when he was registered on the network, and so on. Find and click the **Edit Properties** button below the details, make the required changes, and click the **Save Properties** button.

Get Email Alerts via IM

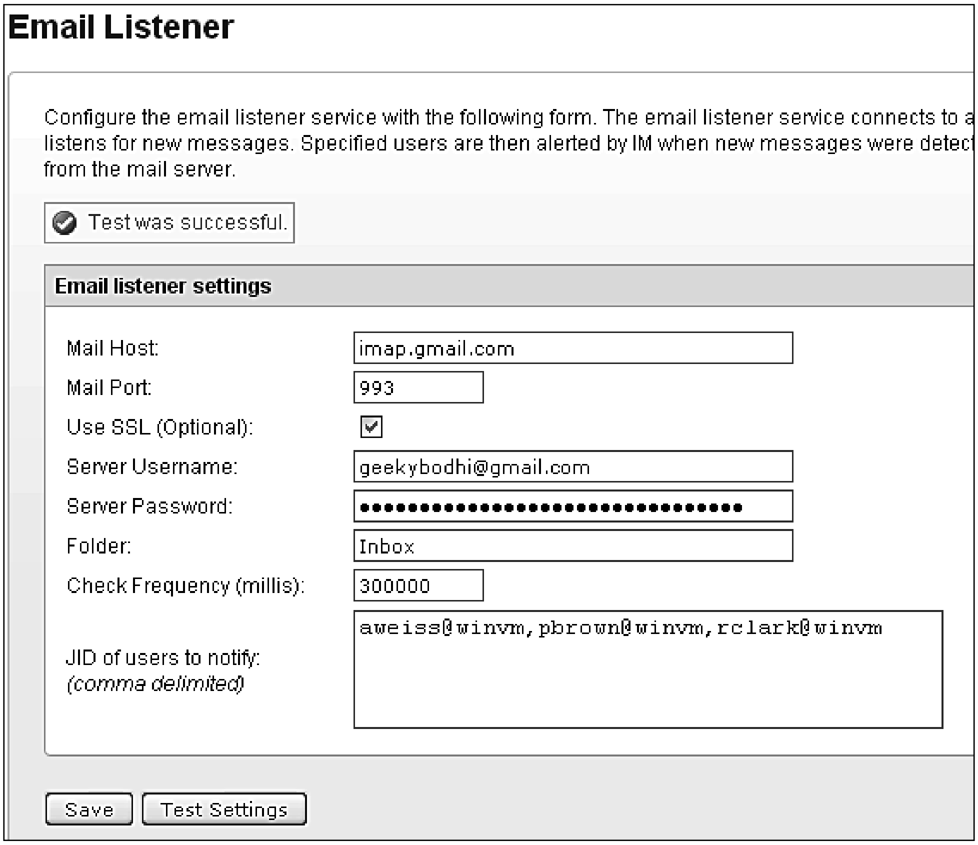
As I said in the opening chapter, Instant Messaging is an alternate line of enterprise communication, along with electronic ones such as email and traditional ones such as the telephone. Some critical tasks require instant notification and nothing beats IM when it comes to time-critical alerts.

For example, most critical server software applications, especially the ones facing outwards on to the Internet, are configured to send an email to the admin in case of an emergency—for example, a break-in attempt, abnormal shutdown, hardware failure, and so on. You can configure Openfire to route these messages to you as an IM, if you're online. If you're a startup that only advertises a single info@cool-startup.com email address which is read by all seven employees of the company, you can configure Openfire to send IMs to all of you when the VCs come calling!

Setting this up isn't an issue if you have the necessary settings handy. The email alert service connects to the email server using IMAP and requires the following options:

- Mail Host: The host running the email service. Example: imap.example.com
- Mail Port: The port through which Openfire listens for new email. SSL can also be used if it is enabled on your mail server. Example: 993.
- Server Username: The username of the account you want to monitor. Example: info@cool-startup.com.
- Server Password: The accounts password.
- Folder: The folder in which Openfire must look for new messages. Typically this will be the "Inbox" but if your server filters email that meet a preset criteria into a particular folder, you need to specify it here.
- Check Frequency: How frequently Openfire should check the account for new email. The default value is 300000 ms which is equal to 5 minutes.
- JID of users to notify: This is where you specify the Openfire Jabber IDs (userids) of the users you want to notify when a new email pops up. If you need to alert multiple users, separate their JID's with commas.

But first head over to the Plugins tab and install the Email Listener plugin from the list of available plugins. Once you have done this, head back to the Server tab and choose the **Email Listener** option in the sidebar and enter the settings in the form that pops up (refer to the following screenshot). Click the **Test Settings** button to allow Openfire to try to connect to the server using the settings provided. If the test is successful, finish off the setup procedure by clicking the **Save** button to save your settings. If the test fails, check the settings and make sure that the email server is up and running. You can test and hook them with your Gmail account as well.



Email Listener

Configure the email listener service with the following form. The email listener service connects to a
listens for new messages. Specified users are then alerted by IM when new messages were detec
from the mail server.

✓ Test was successful.

Email listener settings

Mail Host:	imap.gmail.com
Mail Port:	993
Use SSL (Optional):	✓
Server Username:	geekybodhi@gmail.com
Server Password:	••••••••••••••••••••••••••••••••
Folder:	Inbox
Check Frequency (millis):	300000
JID of users to notify: *(comma delimited)*	aweiss@winvm,pbrown@winvm,rclark@winvm

That's it. Now close that email client you have running in the background, and let Openfire play secretary, while you write your world domination application!

Broadcasting Messages

Since Openfire is a communication tool, it reserves the coolest tricks in the bag for that purpose. The primary purpose of Openfire remains one-to-one personal interactions and many-to-many group discussion, but it can also be used as a one-to-many broadcasting tool.

This might sound familiar to you. But don't sweat, I'm not repeating myself. The one-to-many broadcasting we cover in this section is different from the Send Message tool we've covered in the last chapter. The Send Message tool from the web-based Openfire administration console is available only to the Openfire administrator. But the plugin we cover in this section has a much broader perspective.

For one, the Broadcast plugin can be used by non-admin users, though of course, you can limit access. Secondly, the Broadcast plugin can be used to send messages to a select group of users which can grow to include everyone in the organization using Openfire.

One use of the broadcast plugin is for sending important reminders. Here are some examples:

- The Chief Accounts Officer broadcasts a message to everyone in the organization reminding them to file their returns by a certain date.
- The CEO broadcasts a message explaining the company's plans to merge with or acquire another company, or just to share a motivational message.
- You, the Openfire administrator, use the plugin to announce system outages.
- The Sales Department Head is upset because sales targets haven't been met and calls for a group meeting at 10:00 a.m. on the day after tomorrow and informs everyone in the Sales department via the plugin.
- The intern in the advertisement department sends a list of his accounts to everyone in the department before returning to college and saves everyone a lot of running around, thanks to the plugin.

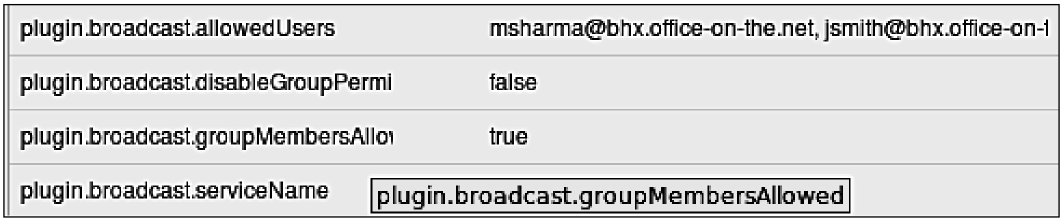
Setting up the Plugin

To reap the benefits of the Broadcast plugin, begin by installing it from under the Available Plugins list on the Plugins tab. This plugin has a few configuration options which should be set carefully—using a misconfigured broadcast plugin, the new guy in the purchase department could send a message of "Have you seen my stapler?" to everyone in the organization, including the CEO!

The broadcast plugin is configured via the Openfire system properties. Remember these? They are listed under the Server tab's **System Properties** option in the sidebar. You'll have to manually specify the settings using properties (refer to the following screenshot):

- `plugin.broadcast.serviceName`—This is the name of the broadcast service. By default the service is called "broadcast", but you can call it something else, such as "shout", or "notify".
- `plugin.broadcast.groupMembersAllowed`—This property accepts two values—true and false. If you select the "true" option, all group members will be allowed to broadcast messages to all users in the group they belong to. If set to "false", only group admins can send messages to all members of their groups. The default value is "true".

- `plugin.broadcast.disableGroupPermissions`—Like the previous property, this property also accepts either true or false values. By selecting the "true" option, you will allow any user in the network to broadcast messages to any group and vice versa, the "false" option restricts the broadcasting option to group members and admins. The default value of this group is "false". As you can imagine, if you set this value to "true" and allow anyone to send broadcast messages to a group, you effectively override the restrictive value of the previous setting.
- `plugin.broadcast.allowedUsers`—Do not forget to set this property! If it is not set, anyone on the network can send a message to everyone else on the network. There are a only a few people you'd want to have the ability to broadcast a message to everyone in the organization. This list of users who can talk to everyone should be specified with this property by a string of comma-separated JIDs.

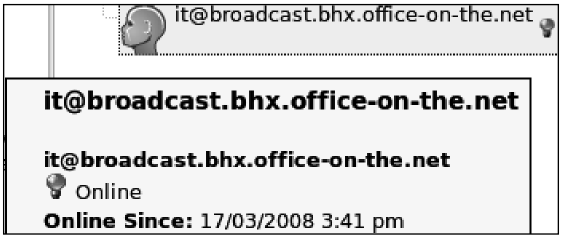


plugin.broadcast.allowedUsers	msharma@bhx.office-on-the.net, jsmith@bhx.office-on-t
plugin.broadcast.disableGroupPermi	false
plugin.broadcast.groupMembersAllo	true
plugin.broadcast.serviceName	plugin.broadcast.groupMembersAllowed

In most cases, the default options of these properties should suffice. If you don't change any variables, your service will be called "broadcast" and will allow group members to broadcast messages to their own groups and not to anyone else. You should also add the JIDs of executive members of the company (CEO, MD, etc.) to the list of users allowed to send messages to everyone in the organization.

Using The Plugin

Once you have configured the plugin, you'll have to instruct users on how to use the plugin according to the configuration. To send a message using the broadcast plugin, users must add a user with the JID in the following format <group>@<servicename>.<server-name> (refer to the following screenshot).



it@broadcast.bhx.office-on-the.net

it@broadcast.bhx.office-on-the.net

it@broadcast.bhx.office-on-the.net
Online
Online Since: 17/03/2008 3:41 pm

If the CEO wants to send a message to everyone, he has to send it to a user called `all@broadcast.serverfoo`, assuming that you kept the default settings, and that your Openfire server is called serverfoo. Similarly, when members of the sales department want to communicate with their departmental collegues, they have to send the message to `sales@broadcast.serverfoo`.

Managing User Clients

There's no dearth of IM clients. It's said that if you have ten users on your network, you'll have at least fifteen different clients. Managing user's clients is like bringing order to chaos. In this regard you'll find that Openfire is biased towards its own IM client, Spark. But as it has all the features you'd expect from an IM client and runs on multiple platforms as well, one really can't complain.

So what can you control using the client control features? Here's a snapshot:

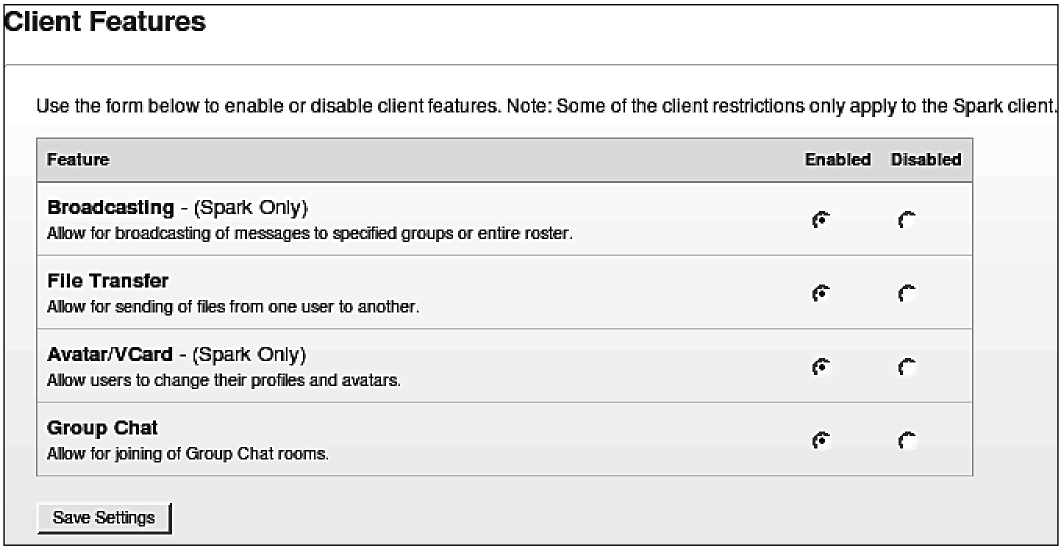
1. Don't like users transferring files? Turn it off, irrespective of the IM client.
2. Don't like users experimenting with clients? Restrict their options.
3. Don't want to manually install Spark on each and every user's desktop? Put it on the network, and send them an email with a link, along with installation and sign-in instructions.
4. Do users keep forgetting the intranet website address? Add it as a bookmark in their clients.
5. Don't let users bug you all the time asking for the always-on "hang-out" conference room. Add it as a bookmark to their client!

Don't these features sound as if they can take some of the work off your shoulders? Sure, but you'll only truly realize how cool and useful they are when you implement them! So what are you waiting for? Head over to the **Plugins** tab and install the **Client Control** plugin. When it is installed, head over to the **Server | Client Management** tab. Here you'll notice several options.

The first option under client management, **Client Features**, lets you enable or disable certain client features (refer to the following screenshot). These are:

1. **Broadcasting**: If you don't want your users to broadcast messages, disable this feature. This applies only to Spark.
2. **File Transfer**: Disabling this feature will stop your users from sharing files. This applies to all IM clients.
3. **Avatar/VCard**: You can turn off indiscriminate changes to a user's avatar or virtual visiting card by disabling this experimental feature which only applies to Spark.
4. **Group Chat**: Don't want users to join group chat rooms? Then disable this feature which will prevent all the users from joining discussion groups, irrespective of the IM client they are using.

By default, all of these features are enabled. When you've made changes as per your requirements, remember to save the settings using the **Save Settings** button.

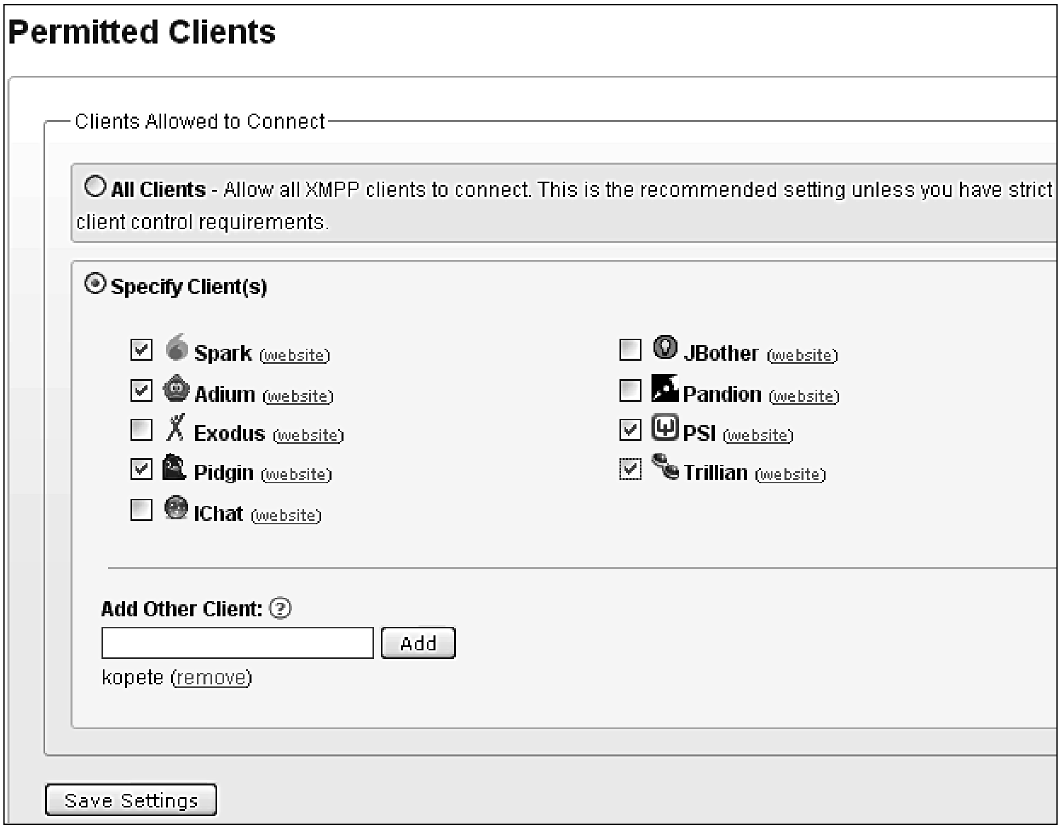


Client Features

Use the form below to enable or disable client features. Note: Some of the client restrictions only apply to the Spark client.

Feature	Enabled	Disabled
Broadcasting - (Spark Only) Allow for broadcasting of messages to specified groups or entire roster.	●	○
File Transfer Allow for sending of files from one user to another.	●	○
Avatar/VCard - (Spark Only) Allow users to change their profiles and avatars.	●	○
Group Chat Allow for joining of Group Chat rooms.	●	○

Next, head over to the **Permitted Clients** option (refer to the following screenshot) to restrict the clients that users can employ. By default, Openfire allows all XMPP clients to connect to the server. If you want to run a tight ship, you can decide to limit the number of clients allowed by selecting the **Specify Clients** option button. From the nine clients listed for the three platforms supported by Openfire (Windows, Linux, and Mac), choose the clients you trust by selecting the checkbox next to them. If your client isn't listed, use the **Add Other Client** text box to add that client. When you've made your choices, click on the **Save Settings** button to save and implement the client control settings.



Permitted Clients

Clients Allowed to Connect

○ **All Clients** - Allow all XMPP clients to connect. This is the recommended setting unless you have strict client control requirements.

● **Specify Client(s)**

☑ **Spark** (website)	☐ **JBother** (website)
☑ **Adium** (website)	☐ **Pandion** (website)
☐ **Exodus** (website)	☑ **PSI** (website)
☑ **Pidgin** (website)	☑ **Trillian** (website)
☐ **IChat** (website)	

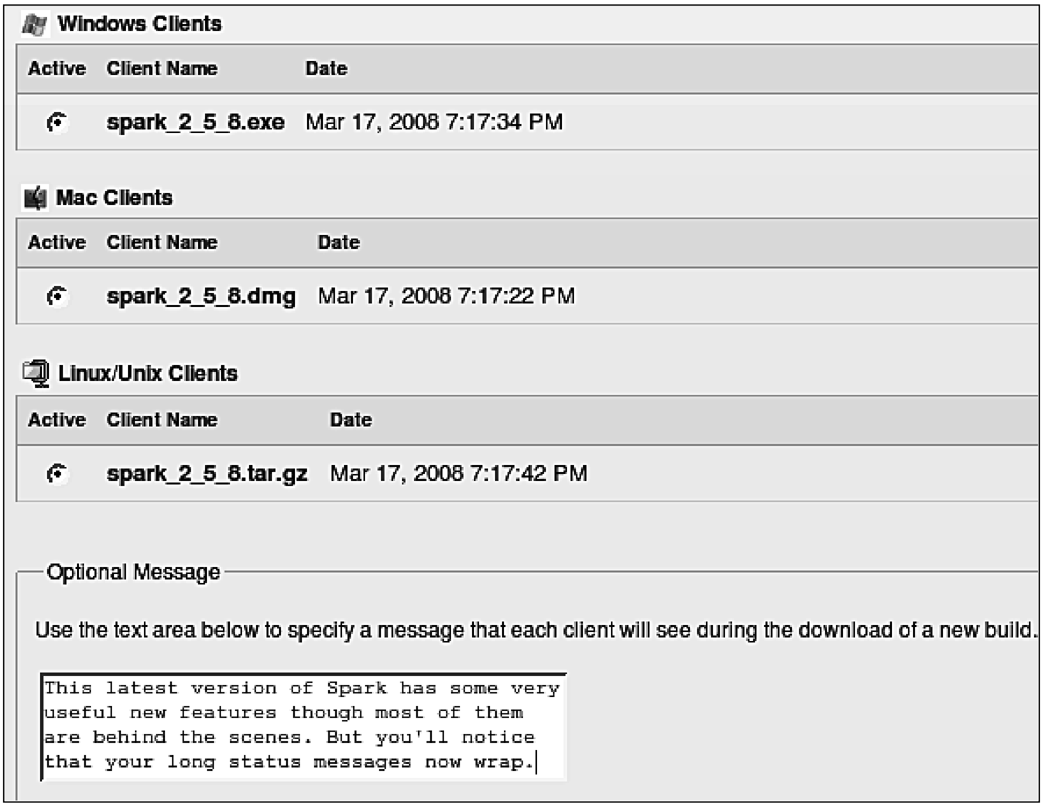
Add Other Client: [ ] Add

kopete (remove)

Save Settings

[The manually-added clients are automatically added to the list of allowed clients. If you don't trust them, why add them? The **remove** link next to these clients will remove them from the list of clients you trust.]

If Spark is one of the IM clients your users are allowed to use, Openfire will help you manage and roll out new versions of Spark for various plantforms as they are relaeased. From the **Spark Version** option in the sidebar, you can upload versions of Spark for each of the three platforms it supports—Linux, Windows, and Mac. You can also place them in the directory specified on that page (such as `C:\Program Files\Openfire\enterprise\spark`).


Windows Clients

Active	Client Name	Date
●	spark_2_5_8.exe	Mar 17, 2008 7:17:34 PM

Mac Clients

Active	Client Name	Date
●	spark_2_5_8.dmg	Mar 17, 2008 7:17:22 PM

Linux/Unix Clients

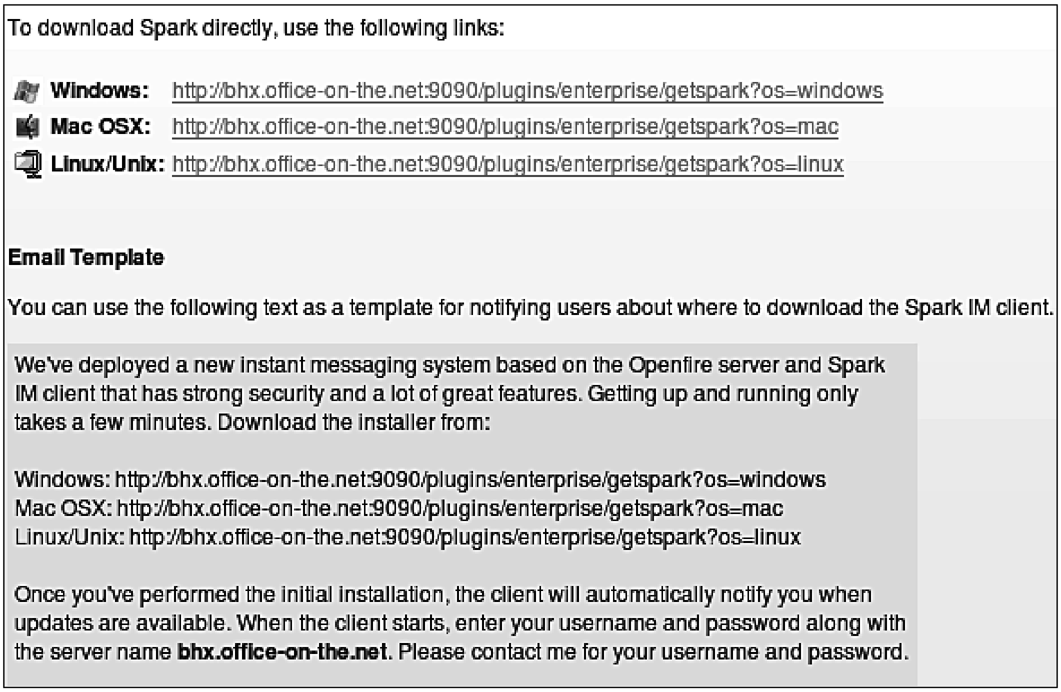
Active	Client Name	Date
●	spark_2_5_8.tar.gz	Mar 17, 2008 7:17:42 PM

Optional Message

Use the text area below to specify a message that each client will see during the download of a new build.

```
This latest version of Spark has some very
useful new features though most of them
are behind the scenes. But you'll notice
that your long status messages now wrap.
```

In case you have multiple Spark versions for a particular platform, such as when a new version is released, you have the option of selecting an **Active** client (refer to the previous screenshot). If the "Active" version is a newer version the one currently deployed, users on the network will automatically be notified that a new version is available for download. The page also has an **Optional Message** text box that you can use to specify a short message that users will see when downloading the client from the Openfire server.



To download Spark directly, use the following links:

Windows: http://bhx.office-on-the.net:9090/plugins/enterprise/getspark?os=windows
Mac OSX: http://bhx.office-on-the.net:9090/plugins/enterprise/getspark?os=mac
Linux/Unix: http://bhx.office-on-the.net:9090/plugins/enterprise/getspark?os=linux

Email Template

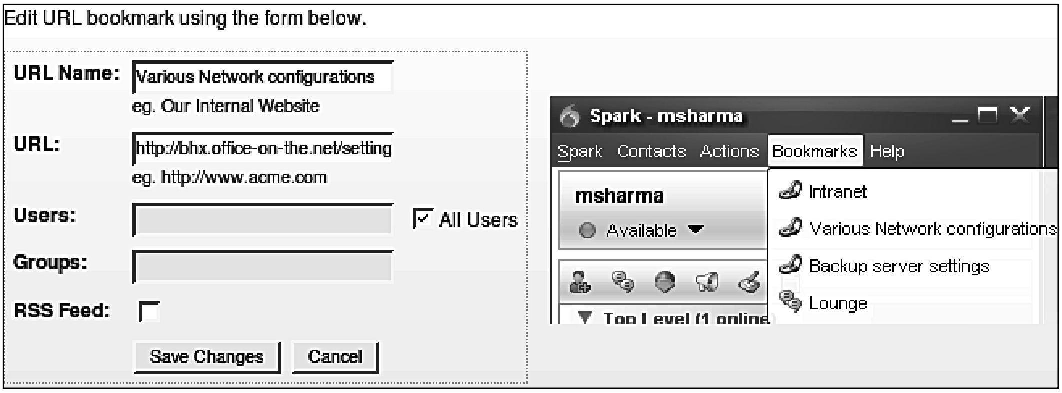
You can use the following text as a template for notifying users about where to download the Spark IM client.

We've deployed a new instant messaging system based on the Openfire server and Spark IM client that has strong security and a lot of great features. Getting up and running only takes a few minutes. Download the installer from:

Windows: http://bhx.office-on-the.net:9090/plugins/enterprise/getspark?os=windows
Mac OSX: http://bhx.office-on-the.net:9090/plugins/enterprise/getspark?os=mac
Linux/Unix: http://bhx.office-on-the.net:9090/plugins/enterprise/getspark?os=linux

Once you've performed the initial installation, the client will automatically notify you when updates are available. When the client starts, enter your username and password along with the server name **bhx.office-on-the.net**. Please contact me for your username and password.

Unlike an update, when you deploy the Openfire server for the first time, sending IM updates to users wouldn't do them any good. In this case, you can use Openfire's direct download links to download Spark. Under the **Download Spark** option in Openfire Enterprise, there are three sets of download links, one each for the three platforms on which Spark can run (refer to the screenshot above). Below the download links is a template email blurb that you can send to all of your users. The email informs the users of the new IM server that you have deployed and has instructions on downloading a client (Spark) for their platform as well as information on logging on.


Edit URL bookmark using the form below.

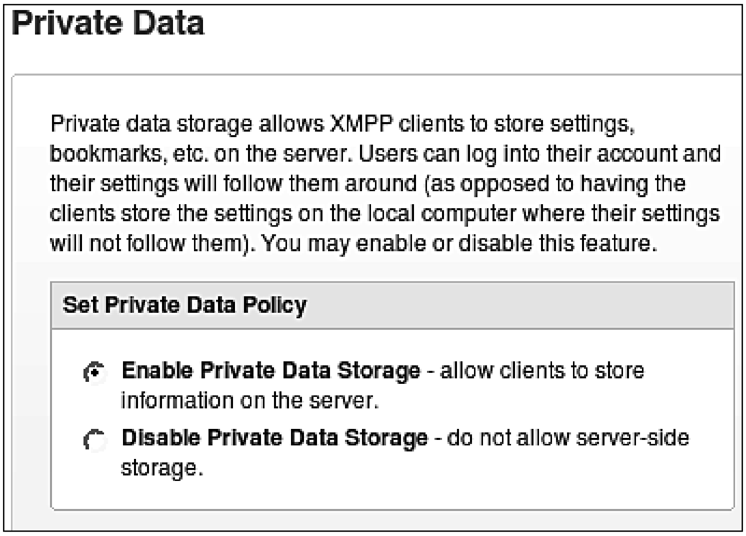
URL Name:	Various Network configurations
	eg. Our Internal Website
URL:	http://bhx.office-on-the.net/setting
	eg. http://www.acme.com
Users:	[✓ All Users]
Groups:	
RSS Feed:	☐

Before the users get their hands on Spark, it's a good idea to load them with important bookmarks (refer to the screenshot above). Then you can include this information in the introductory email as well. Wouldn't your users love you if you told them that all of the important intranet URLs and the settings page are bookmarked in their clients and that if they ever run into any problems they should join and ask for solutions in the bookmarked troubleshoot chat room, the company's always-on IT helpline. You'll be a demigod! All hail the Openfire admin!

Private Data Storage

Do your users move around a lot? If yours is a terminal server environment where users don't have a fixed computer but rather they log in to the server (which stores their files) from any machine, it doesn't make sense to store the user's settings on a particular machine. Openfire can support this setup by aping the network settings and storing user files on the server instead of a local machine. This is done via the Private Data option.

Using private data storage, all XMPP/Jabber clients store their settings, group and web bookmarks, and so on, for every user on the Openfire server. This allows users the freedom to log into their account from any machine on the network, and their settings will follow them around.



Private Data

Private data storage allows XMPP clients to store settings, bookmarks, etc. on the server. Users can log into their account and their settings will follow them around (as opposed to having the clients store the settings on the local computer where their settings will not follow them). You may enable or disable this feature.

Set Private Data Policy

[✓ **Enable Private Data Storage** - allow clients to store information on the server.]

Disable Private Data Storage - do not allow server-side storage.

To enable this feature, head over to **Server | Server Settings | Private Data Storage** (refer to the previous screenshot). All you have to do now is choose the **Enable Private Data Storage** option button and save the settings. If, on the other hand, you'd rather bind a user's settings to the machine from which they first log into the server, choose the **Disable Private Data Storage** option.

Importing/Exporting Users

Coming back to reality, there are only very few things in an admin's list of things that come close to a server migration. My 10-year veteran admin friend thinks there's nothing like a server migration to make your day! For him, migration involves a huge checklist and a wasted weekend. But if you use Openfire, you can retain your happy-go-lucky charm and get the job done before your users return from lunch.

Migration is actually a small subset of a bigger problem. There are a couple of situations that can very well be described as an admin's worst nightmare:

1. Moving over to Openfire: Ok, so this isn't totally a waste of effort—you do get a quality IM server at the end of the day. But exporting all of the users, their settings, and their rosters and then importing them into the new server can be quite a hassle.
2. Moving from one Openfire server to another: So your old server is slowly bleeding to death. You love the beast but you've got to cut it loose. Not only do you now have to export data from the old server and then import it into the new server, but you also have to take into account the new domain/server name. You shiver in fear—or excitement!

It might sound retro, but whatever the case maybe, have no fear, Openfire is here. Irrespective of whether you are switching servers, or moving users from an existing XMPP/Jabber IM server into Openfire, you're doing the right thing and you *know* you're doing the right thing because Openfire makes the process a walk-in-the-park—thanks to the user Import/Export plugin.

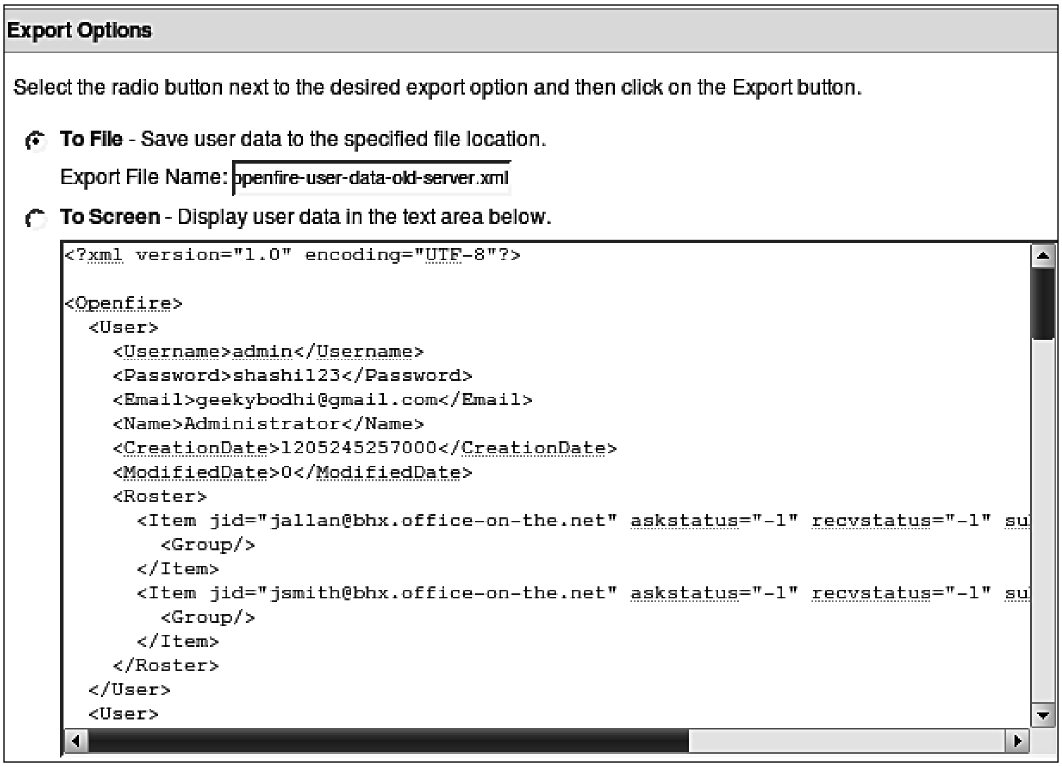
The user Import/Export plugin lets you import and export Openfire user data via the Admin Console. This user data includes usernames, passwords, names, email addresses, creation and modification dates, and roster lists. Also, as I said, this plugin helps you when migrating users from other Jabber/XMPP based systems to Openfire.

Using The Plugin

For all the tough work it does, the plugin is relatively simple to use. Install it from the list of Available Plugins under the Plugins tab. Then head over to the Users/Groups tab where you'll notice a new **Import & Export** sub-tab. Clicking on this brings you to a simple page with two links—one to import user data, and the other to export user data.

Openfire data is exported as an XML file. There are two options when exporting user data. You can either save the export data to a file name or have it displayed on the screen and then copy-paste it to wherever you want to (refer to the following screenshot).

Select the **To File** option by clicking on the option button adjacent to it and specify a file name in the **Export File Name**: text box. When you click on the **Export** button, the Openfire user XML data will be saved to the file you specified.



Export Options

Select the radio button next to the desired export option and then click on the Export button.

[✓ **To File** - Save user data to the specified file location.]

Export File Name: openfire-user-data-old-server.xml

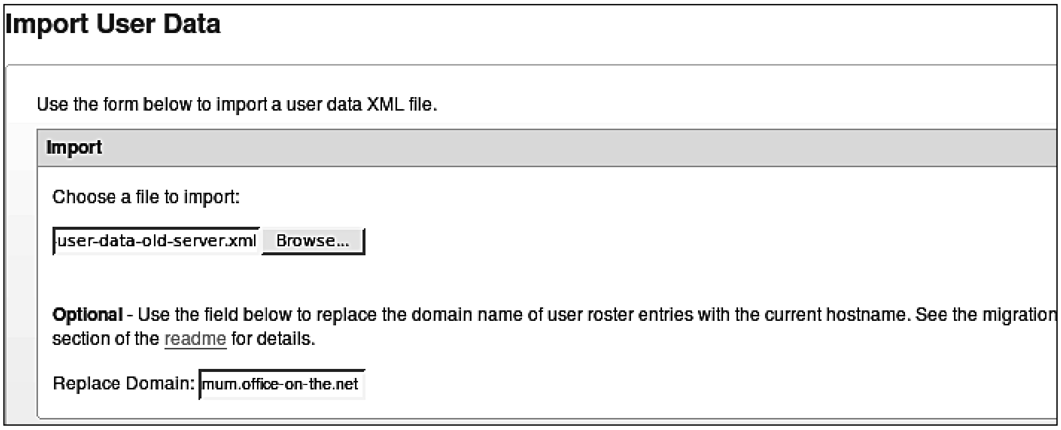
To Screen - Display user data in the text area below.

```
<?xml version="1.0" encoding="UTF-8"?>

<Openfire>
  <User>
    <Username>admin</Username>
    <Password>shashil23</Password>
    <Email>geekybodhi@gmail.com</Email>
    <Name>Administrator</Name>
    <CreationDate>1205245257000</CreationDate>
    <ModifiedDate>0</ModifiedDate>
    <Roster>
      <Item jid="jallan@bhx.office-on-the.net" askstatus="-1" recvstatus="-1" su
        <Group/>
      </Item>
      <Item jid="jsmith@bhx.office-on-the.net" askstatus="-1" recvstatus="-1" su
        <Group/>
      </Item>
    </Roster>
  </User>
  <User>
```

The other option is to copy-paste the data yourself. Some administrators might prefer this option to save the data in multiple places or on remote machines. To export the data manually, choose the **To Screen** option button and click on the **Export** button. This displays the data in the text box provided.

> Be careful when exporting user data, because all of the user passwords in the exported XML file are displayed unencrypted as plain text and can be used by anyone. It's not a bad idea to ask users to change their passwords immediately after you have completed the migration process.

You can import data into Openfire via an XML file imported from an XMPP/Jabber server. On the import page (refer to the following screenshot), the **Browse** button will help you locate the file that contains the user information. When you have the file, click on the **Import** button. This starts the import process.


Import User Data

Use the form below to import a user data XML file.

Import

Choose a file to import:

user-data-old-server.xml

Optional - Use the field below to replace the domain name of user roster entries with the current hostname. See the migration section of the readme for details.

Replace Domain: mum.office-on-the.net

When the plugin has successfully imported all user data, it will display a message stating **All users added successfully**. However, if the plugin was not successful in importing all user data, it will display an error message indicating what might have gone wrong. You can resolve the error and try again. Don't worry about duplicate data when resuming a broken import. If the plugin detects a user that already exists in the system, it will skip importing that user or its roster information.

During the import process, you also have the option of changing the domain name of the data being imported from the old domain to the new domain. To do this, after selecting the file to import the data from, in the **Import User Data** section, specify the old domain name as mentioned in the import file in the **Replace Domain** text field. Now when Openfire imports the data, it will replace all instances of the old domain, such as in roster information, with the domain name of the Openfire server you are importing the data into.

Here is a sample of an exported user list from the plugin's `readme` file. It contains two users, Joe and Sally, who have added each other to their respective rosters.

```
<?xml version="1.0" encoding="UTF-8"?>

<Openfire>
 <User>
   <Username>joe</Username>
   <Password>joepwd</Password>
   <Email></Email>
   <Name></Name>
   <CreationDate>1125601449177</CreationDate>
   <ModifiedDate>1125601449177</ModifiedDate>
   <Roster>
     <Item jid="sally@localhost" askstatus="-1" recvstatus="-1"
```

```
                                            substatus="3" name="Sally">
          <Group/>
        </Item>
      </Roster>
    </User>
    <User>
      <Username>sally</Username>
      <Password>sallypwd</Password>
      <Email></Email>
      <Name></Name>
      <CreationDate>1125601471848</CreationDate>
      <ModifiedDate>1125601471848</ModifiedDate>
      <Roster>
        <Item jid="joe@localhost" askstatus="-1" recvstatus="-1"
   substatus="3">
          <Group/>
        </Item>
      </Roster>
    </User>
   </Openfire>
```

Notice the different status types? Here is a list of all of the different status types, with a brief description, also from the plugin's `readme` file.

askstatus

- -1—The roster item has no pending subscription requests.
- 0—The roster item has been asked for permission to subscribe to its presence but no response has been received.
- 1—The roster owner has asked the roster item to be unsubscribed from its presence notifications but hasn't yet received confirmation.

recvstatus

- -1—There are no subscriptions that have been received but not presented to the user.
- 1—The server has received a subscribe request, but has not forwarded it to the user.
- 2—The server has received an unsubscribe request, but has not forwarded it to the user.

substatus

- -1—Indicates that the roster item should be removed.
- 0—No subscription is established.
- 1—The roster owner has a subscription to the roster item's presence.
- 2—The roster item has a subscription to the roster owner's presence.
- 3—The roster item and the owner have a mutual subscription.

Summary

In this chapter, we've explored several ways of effectively administrating users. The tips and tricks covered in this chapter will help you no matter how many users you have on the network— three or three thousand. Openfire can scale effortlessly as your organization grows, and thanks to some very useful accessories, you'll be saved a lot of headache when it does.

If you are using an existing IM server, you can start enjoying the benefits of Openfire right from the word go. This chapter has shown how you can import user data into their roster lists from any XMPP server into Openfire. The import tool can also be used to export user data when switching servers. It will automatically make changes to the domain name in the old user data when migrating the server to another domain.

We've then seen how easily you can maintain decorum in conversations. Openfire can also work with emails to alert you of important messages. But it's in a class of its own when it comes to doing what it's designed to do—help you communicate. We've covered tools that'll help members of the organization broadcast messages to their peers based on preset company rules.

We have also discussed the client control features of the Enterprise version. If you want to run a tight ship, the Enterprise version can let you limit the IM clients authorized to interact with the Openfire server, in addition to disabling certain features. You can also use the Enterprise tools to place versions of Spark (Openfire's official IM client) on the network and alert users to grab the latest version from the intranet, saving valuable bandwidth and leaving you in charge. Ah, every admin's dream come true!

7 Connecting to Other Services

There are several organizations in the world for which keeping communication behind closed doors is paramount to their existence—the kind whose employees sleep with Colt .45s under their pillows. If you look closely, our current Openfire setup is perfectly suitable for such an organization. All current Openfire communication happens behind what we call a walled garden—Openfire lets users interact with others connected to the server, but not with anyone else. Unless your organization operates in a cocoon, you'll want to interact with the external world—after all, even spies have external contacts!

As you might have guessed, Openfire does let you talk to your contacts outside the network. In this chapter, we'll configure Openfire to allow users on the local network to connect to their contacts who use IM services provided by external, publically-accessible networks such as Yahoo! Messenger, AOL, and Google Talk.

But before we get to the configuration bit, aren't you wondering why you can't just allow your users to use the appropriate client to connect to these free services? As you might have guessed, going via an internal IM server, like Openfire, has clear advantages, as explained in the following section.

In this chapter, we will cover:

- Connecting with users on external IM networks
- Controlling and monitoring external conversations
- Connecting with an external Voice Over IP solution

Why Connect via Openfire?

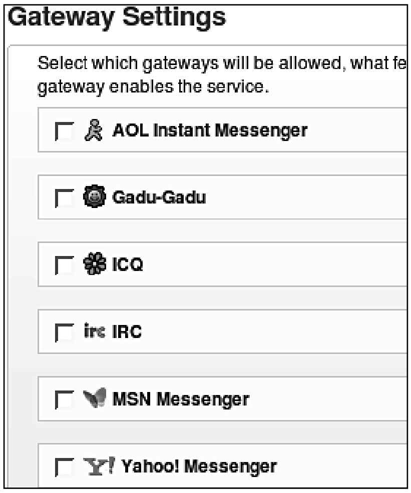
Using Openfire as a gateway to these publically-accessible networks is one Openfire feature that could be counter-productive as well. Although it helps users connect to clients and their business interests outside the network, it also exposes them to people who can distract them from deadlines. You wouldn't want your users finalizing their weekend plans as well as the weekly report at the same time now, would you?

By routing access to the public networks through the Openfire server, you automatically bring these users under the purview of your corporate IM policies. It makes conversations easy to monitor and control, in a sense forcing people to stick to business correspondence, unless of course they don't mind explaining why they were exchanging notes with funny_dude86 on how they like their martini for well over an hour!

This might sound a little harsh and unfair. Most companies ignore brief periods of indulgence, but any conversation that goes over an hour, even within the organization, is bound to raise eyebrows. Some companies relax these guidelines during lunch hours and on weekends, but this is something that varies from organization to organization and you have to decide what policy works best for yours.

The Openfire Gateway

So far, we've only used Openfire to allow users connected to it to communicate with each other. To get it to act as a gateway for providing access to external services, you need the **IM Gateway** plugin. You'll find it under the list of available plugins on the **Plugins** tab.



Gateway Settings

Select which gateways will be allowed, what fe
gateway enables the service.

☐ AOL Instant Messenger

☐ Gadu-Gadu

☐ ICQ

☐ IRC

☐ MSN Messenger

☐ Yahoo! Messenger

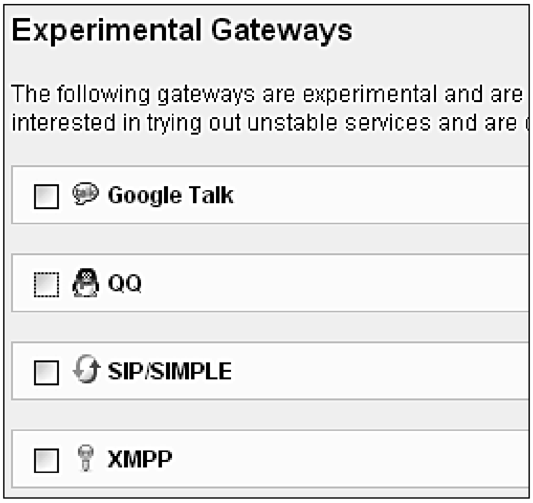
When installed, the plugin creates its own tab that is accessible via **Server | Gateways**. Using the plugin, you can allow your users access to the following gateways (refer to the previous screenshot):

- **AOL Instant Messenger**
- **Gadu-Gadu**
- **ICQ**
- **IRC**
- **MSN Messenger**
- **Yahoo! Messenger**

The plugin is under constant development and you might see new networks being added as it progresses through versions. In addition to the above list of well-tested gateways, the plugin also supports a few "experimental" gateways (refer to the following screenshot) including:

- **Google Talk**
- Tencent **QQ**
- **SIP/SIMPLE**
- **XMPP**

The only difference between these experimental gateways and the ones listed earlier is that the latter haven't been tested enough. Openfire developers advise that the experimental gateways shouldn't be used on production systems as they might not work correctly.


Experimental Gateways

The following gateways are experimental and are
interested in trying out unstable services and are

In my testing, I've used two of these experimental gateways and they seem stable enough to me. Neither of them crashed or failed to log the conversation. There were also no noticeable lags between messages, and none of them were lost in transit. However, I haven't tested them under a heavy load, and too much stress might break them. Use them with caution!

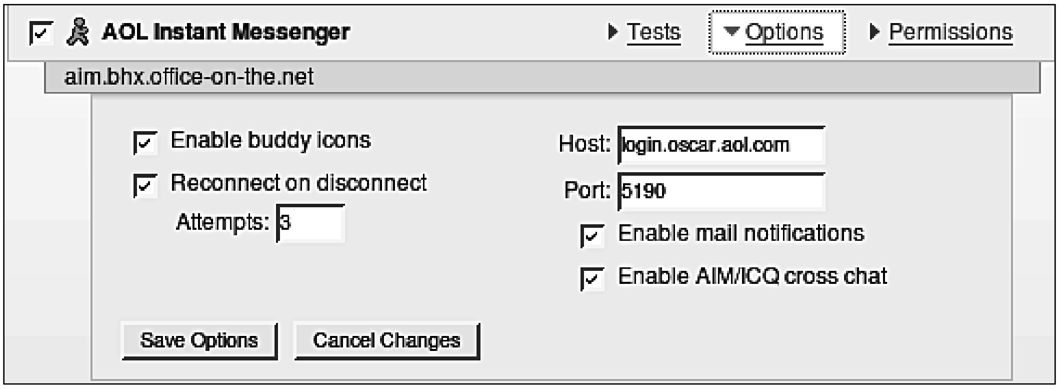
Controlling Access

The Gateway plugin is more than simply a mechanism for connecting Openfire users with publically accessible IM networks. Remember how we discussed limiting access to external networks earlier in the chapter? The Gateway plugin is every bit as clever as the platform it's designed for. In addition to allowing users access to their external IM accounts, the plugin gives you power to let them do so either in a more controlled fashion or in a no-holds-barred fashion if that's what you really want!

Although the options vary from gateway to gateway, between them all you'll be able to:

- Control the number of reconnect attempts.
- Change the server address and port of the gateway. This is particularly useful when selecting an IRC server to connect to.
- Notify on new email to the gateway account, since most gateway services also provide email.
- Display buddy icons, if supported.
- Enable cross-chat with another gateway (such as AIM/ICQ), if available.

Most of these options are enabled by default, but they can be tweaked easily as we'll see later in this chapter. They might sound too trivial to spend much time on, but you'll have to use the options listed above carefully. Depending on the number of users you have on your network, providing access to gateways that display icons can end up choking the network by enabling buddy icons (refer to the following screenshot), especially if you are on a low bandwidth connection. The same goes for the email in the external email account. Although most of these services have decent spam protection, you might not want users to transgress your organization's email policy by opening email on their external accounts.



aim.bhx.office-on-the.net

☑ Enable buddy icons
☑ Reconnect on disconnect
Attempts: 3
Host: login.oscar.aol.com
Port: 5190
☑ Enable mail notifications
☑ Enable AIM/ICQ cross chat

But these options are user-centric, that is, they are designed to let you configure a better service for your users. But what about you, the admin? Saving a little bandwidth and enforcing your email policy is good, but you need something more concrete if you wish to control indiscriminate use of the gateways. This very perceptive plugin has a few options that you can use to restrict permission for using the plugin to:

- A limited list of users
- Users belonging to a list of groups
- All users

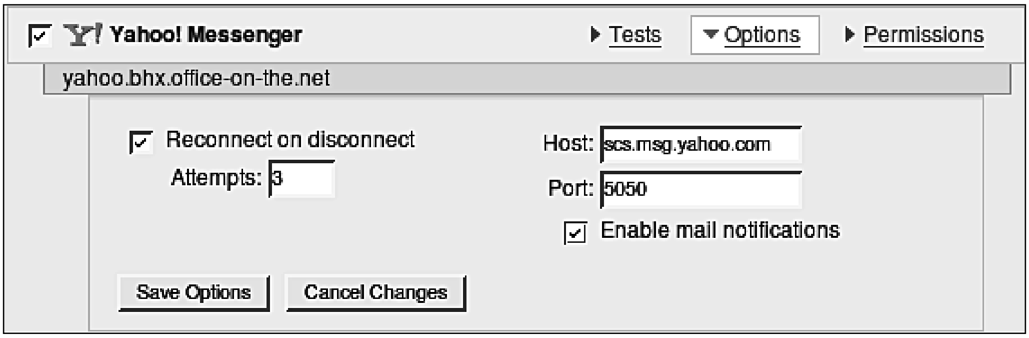
Depending on which option you've selected, the users will be able to register themselves in the specified gateway, or you can manually register users to the gateway to which they want access (they'll have to tell you their username and password), if you are really that paranoid. Also, unlike the options discussed earlier, the permission settings for all gateways remain the same.

Enough talk! It's time we see the Gateway plugin in action.

Scenario #1: Enabling Yahoo! Messenger for All Users

So let's start exploring the plugin with the simplest of requirements—enabling the Yahoo! gateway for all users. To do so, assuming that you've installed the gateway plugin, head over to the **Server | Gateways** tab. From the list of gateways under the **Gateway Settings** section, select the checkbox next to **Yahoo! Messenger** (refer to the following screenshot). This displays a small bar at the bottom with the name of the service in the format servicename.servername (something like yahoo.openfire.example.com).

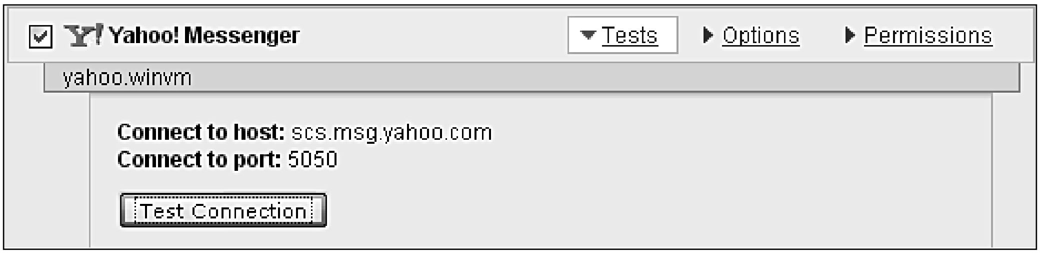
You'll also see three buttons that'll help you configure and test the gateway. To set some gateway specific settings, click the **Options** slider. From the panel that slides down, you can change the address and port of the gateway host. Unless you're having trouble connecting, you should do just fine without any modification to this field. You'll find the option to enable mail alerts below the address. This is enabled by default, but you can choose to disable it.



yahoo.bhx.office-on-the.net

✓ Reconnect on disconnect
Attempts: 3
Host: scs.msg.yahoo.com
Port: 5050
✓ Enable mail notifications

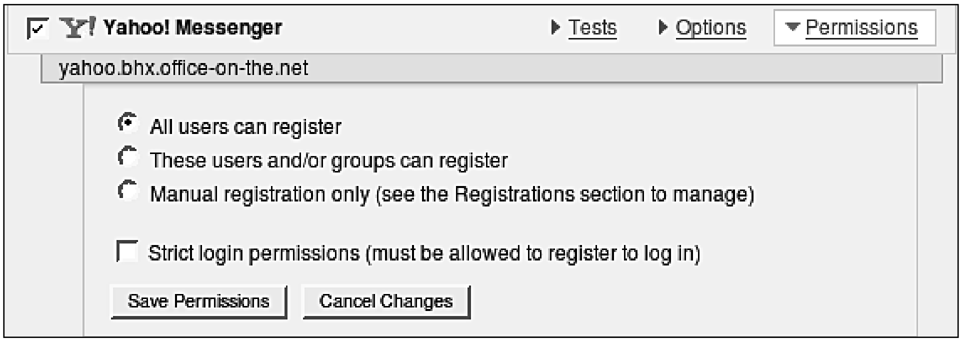
Once you've reviewed the options and made any required changes, click on the **Save Options** button to save them. It's a good idea to test the settings, especially if you have changed the server address or port number. You can do so from the **Tests** button (refer to the following screenshot).



yahoo.winvm

Connect to host: scs.msg.yahoo.com
Connect to port: 5050

When clicked, this button displays the server address and port number which are followed by the **Test Connection** button. If the test is successful, you'll be informed that it was; else you'll have to make changes to the address and port to the gateway server from the **Options** section.



yahoo.bhx.office-on-the.net

[✓ All users can register]
These users and/or groups can register
Manual registration only (see the Registrations section to manage)

☐ Strict login permissions (must be allowed to register to log in)

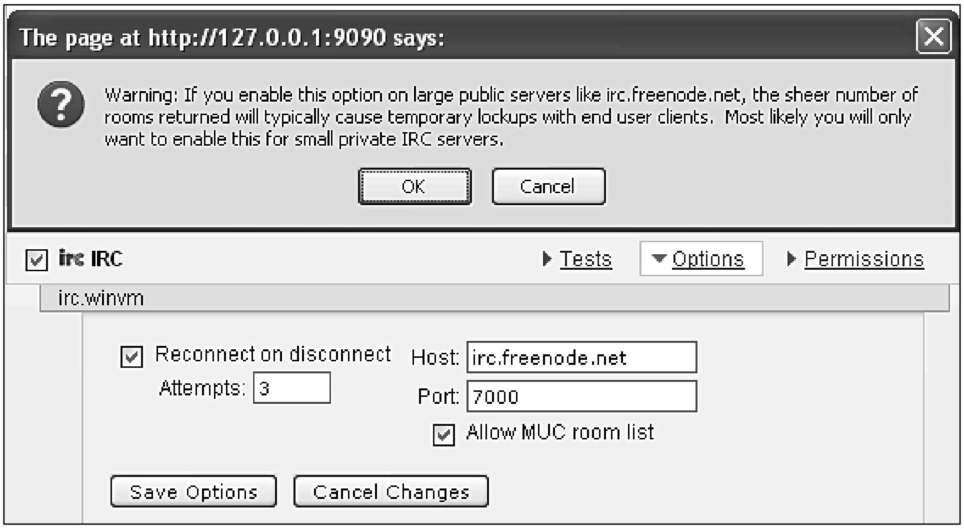
So now we've enabled the Yahoo! Messenger gateway. Next, we head over to the most critical section of the setup—user permissions. Our goal is to enable the Yahoo! Messenger gateway for all users. To do this, click the **Permissions** slider (refer to the following screenshot). From the options that slide down, make sure that the **All users can register** option is selected. Follow this up by saving the settings by clicking the **Save Permissions** button. Presto! All your users will now be able to connect to their contacts on Yahoo! Messenger.

Scenario #2: Enabling IRC for IT and Devs

IRC, or Internet Relay Chat, is a very popular means of multi-user communication often preferred by developers. Conversation on IRC is segregated into channels with several channels on one IRC server. There are many popular IRC servers, but Freenode seems to host the most popular and geeky channels. I write for Linux.com and we host a channel for our readers on OFTC.net.

[You can connect to only one IRC server using the IRC gateway. You cannot have some users on one network and others on another.]

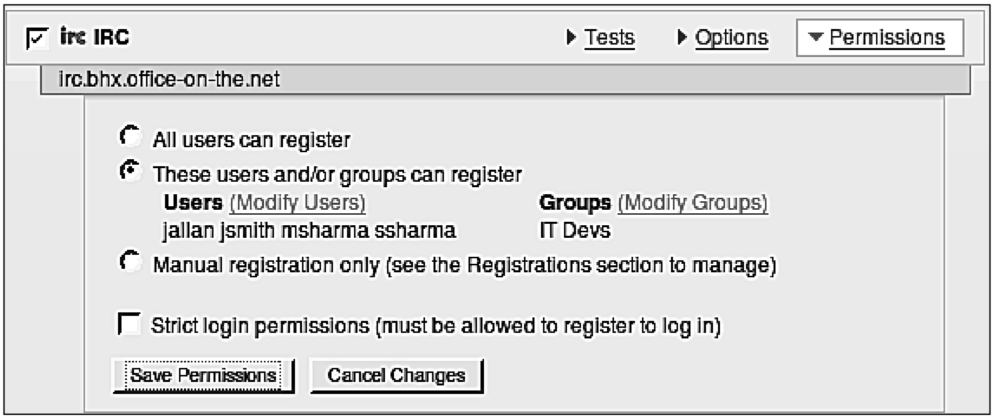
You can change the channel you want to log into from the Options slider. By default, it's set to log in into irc.freenode.net. If the channel you want to connect to is on another server, you'll have to specify it here. Because an IRC server has several channels, the IRC gateway has an option to view the list of channels on a server once a user is connected. To see a list of channels, select the **Allow MUC room list** checkbox (refer to the following screenshot).


The page at http://127.0.0.1:9090 says:

Warning: If you enable this option on large public servers like irc.freenode.net, the sheer number of rooms returned will typically cause temporary lockups with end user clients. Most likely you will only want to enable this for small private IRC servers.

[✓] Reconnect on disconnect
Attempts: 3
Host: irc.freenode.net
Port: 7000
[✓] Allow MUC room list

Although it will be convenient for your users, it might inversely degrade performance. As I said earlier, some popular servers such as FreeNode have a large number of channels and it could take several minutes to fetch this list which could "hang" the client. As before, after making any changes, it's a good idea to test these settings via the Tests slider.



All users can register [✓ These users and/or groups can register] Manual registration only (see the Registrations section to manage)

Users (Modify Users): jallan jsmith msharma ssharma
Groups (Modify Groups): IT Devs

[ ] Strict login permissions (must be allowed to register to log in)

Now comes the tricky bit. Unlike the Yahoo! Gateway, we want to restrict IRC access to users in the **IT** and **Devs** groups. From under the **Permissions** slider (refer to the previous screenshot), select the **These users and/or groups can register** radio button. This further expands the slider to let you restrict access to a list of users or groups. Click on the **Modify Groups** option and, in the text box, enter the name of the groups to which you want to allow access to the gateway. Now when you click on **Save Permissions**, Openfire will check if these groups exist and if they do, will allow members from these groups to access the gateway. Pretty slick, eh?

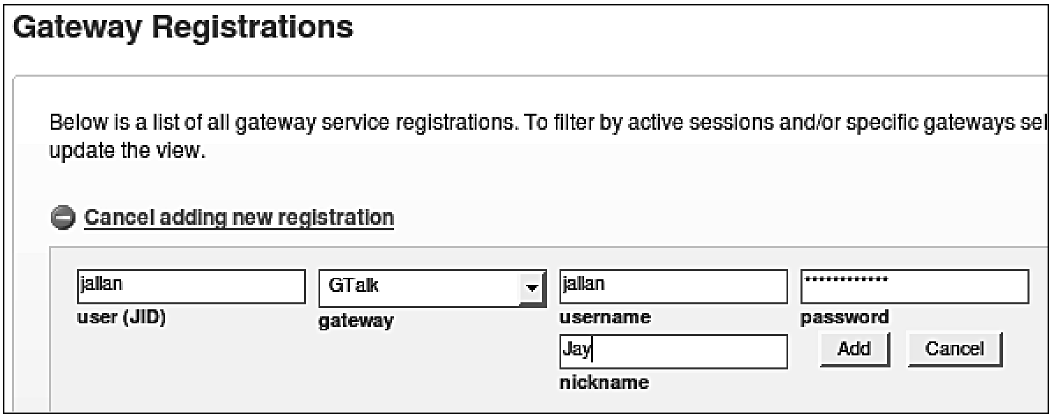
Scenario #3: Enabling Experimental Google Talk for Some Users

Google Talk is slowly gaining momentum as a popular means of publicly-accessible IM. After all, it's based on Jabber/XMPP! But as it's experimental, we'll allow it only for a limited set of users, like your boss! Since there aren't many people using the gateway, you can also enable the mail notifications for this gateway. Google Talk also has the added option of displaying buddy icons (small thumbnail images, generally scaled user images). Once you've saved the options and tested the gateway, it's time to head on to limiting the users.

There are two ways in which you can limit the users who can access a gateway. One is by selecting the **These users and/or groups can register** option button, clicking the **Modify Users** link and then entering the list of users in the text box that slides open.

The other option is to add these users manually. This might not be a good idea if you want to limit access to a large number of users. But since it's your boss we're talking about, I'm sure he'll appreciate it a little if you can help him "get on with it".

What you need to do is select the **Manual registration only** option button.
Once selected, save the permissions and head on over to the top of the page. On the left-hand side, you'll notice that the **Gateways** tab is subdivided into two sections—**Settings** (on which you are on) and **Registrations**. As you've selected the manual registration option, you'll have to add users from the Registrations section.



Gateway Registrations

Below is a list of all gateway service registrations. To filter by active sessions and/or specific gateways sel
update the view.

Cancel adding new registration

user (JID): jallan
gateway: GTalk
username: jallan
password: *************
nickname: Jay

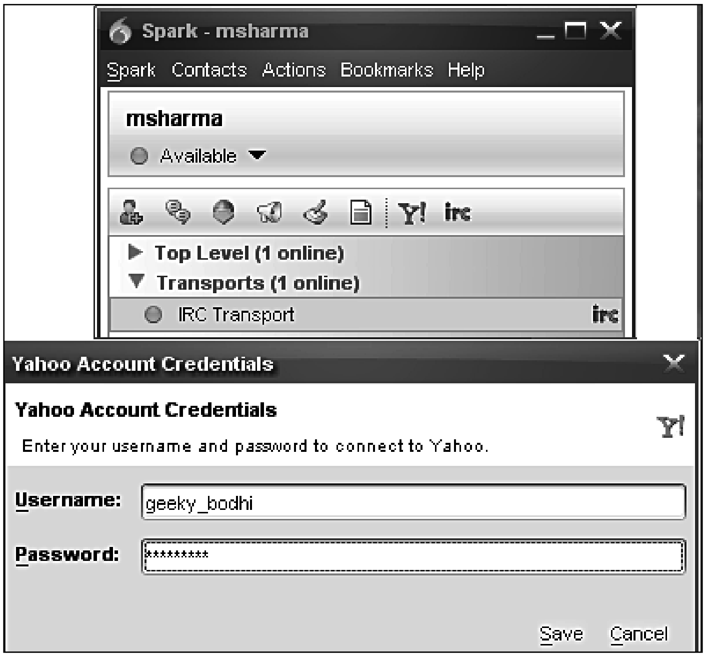
In the Gateway Registrations section, you'll notice a **Add a new registration** link at the top of the page. When clicked, this link slides open a small form (refer to the previous screenshot) that requires you to enter the following details:

- Jabber ID of the user
- **Gateway** to which access should be allowed
- **Username**, **password**, and **nickname** on the gateway

You'll have to add these details for all of the users you want to manually add.

Using A Gateway

When you have enabled a gateway, how do your users get to use it? Actually, it all depends on their client. On my network I use three clients: Spark (Windows), Kopete (Mandriva KDE), and Pidgin (Fedora GNOME).



Yahoo Account Credentials

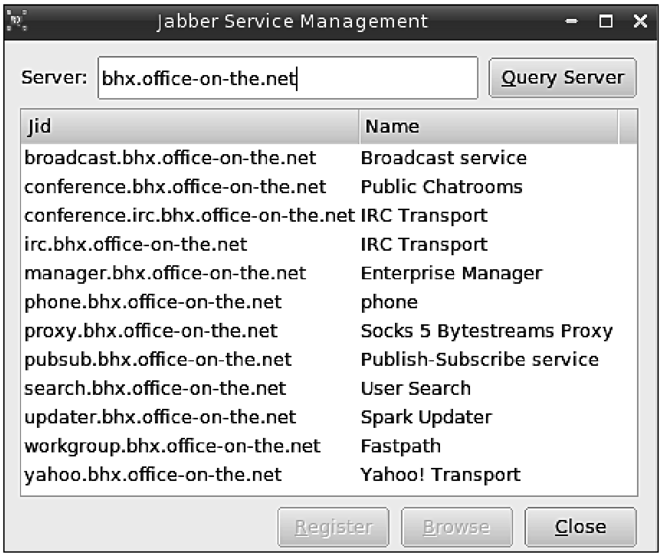
Yahoo Account Credentials

Enter your username and password to connect to Yahoo.

Username: geeky_bodhi

Password: *********

The easiest of the three is Spark. When a gateway is enabled, and the user is allowed to register himself or herself, a gateway icon appears in Spark (refer to the previous screenshot). The user can simply click on the icon and enter their username, password, and nickname, and are connected. Once connected, they can use this icon to disconnect, or to edit user information.

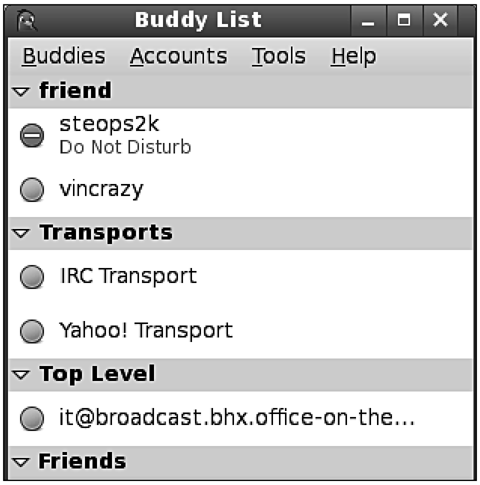


Jabber Service Management

Server: bhx.office-on-the.net

Jid	Name
broadcast.bhx.office-on-the.net	Broadcast service
conference.bhx.office-on-the.net	Public Chatrooms
conference.irc.bhx.office-on-the.net	IRC Transport
irc.bhx.office-on-the.net	IRC Transport
manager.bhx.office-on-the.net	Enterprise Manager
phone.bhx.office-on-the.net	phone
proxy.bhx.office-on-the.net	Socks 5 Bytestreams Proxy
pubsub.bhx.office-on-the.net	Publish-Subscribe service
search.bhx.office-on-the.net	User Search
updater.bhx.office-on-the.net	Spark Updater
workgroup.bhx.office-on-the.net	Fastpath
yahoo.bhx.office-on-the.net	Yahoo! Transport

In Kopete, your users will have to query the Openfire server for the list of available services (see the previous screenshot). In the list, they'll notice a dedicated service for each gateway that you have enabled. They'll have to select the service they wish to register, click the **Register** button, and add their gateway details. The first time they log on to the Openfire server after registering, they'll automatically be logged on to the gateway as well.



Both Spark and Kopete are relatively painless to configure when compared to Pidgin. There is no option to query the server in Pidgin, nor does it show the list of services or gateways available. You have no choice but to manually add those users who wish to use Pidgin. Once you've added the users (from the Registration section under the Gateways tab, which we'll cover in forthcoming sections) and allowed them access to a particular gateway, their contacts from the gateway will automatically popup in Pidgin (refer to the screenshot above).

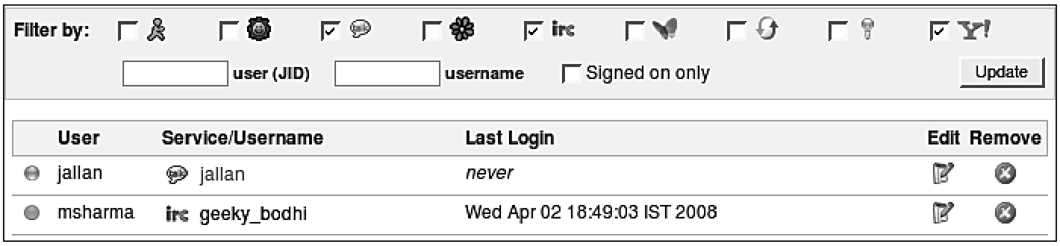
Keeping An Eye on The Gateway

In addition to being used for adding users, the Registration section also doubles up as a gateway monitor. It keeps track of the users who have signed up for one of the enabled gateway services. From here (refer to the following screenshot) you can:

- Filter users by gateway
- Narrow down a selection to only signed in users
- Find a user by their JID on the Openfire server
- Find a user by their username on the Gateway service

Once you've found a user, you can:

- Immediately find out what gateway they have subscribed to
- See the last time they logged on to the gateway
- See the gateway(s) on which they are currently online
- Remove them from the gateway
- Edit their gateway settings (username, password, and nickname on the gateway service)



Filter by: ☐ ☐ ☑ ☐ ☑ irc ☐ ☐ ☐ ☑ Y!

user (JID) username ☐ Signed on only Update

	User	Service/Username	Last Login	Edit	Remove
●	jallan	jallan	*never*		
●	msharma	irc geeky_bodhi	Wed Apr 02 18:49:03 IST 2008		

Connecting to VoIP

In addition to connecting to external conversation gateways, Openfire can also interact with a Voice over Internet Protocol (VoIP) service, if you have one in your network. In this section, we'll see how you can hook up your Openfire setup with Asterisk.

I'm assuming that you have an Asterisk VoIP server setup. If you don't, there is lots of online documentation that'll help. In addition, there's also Packt's Asterisk title—Building Telephony Systems With Asterisk. If you are new to Asterisk, I'd suggest you grab the TrixBox VMware image [`http://www.trixbox.org/downloads`] that provides a ready-to-use Asterisk environment.

To connect Openfire to your Asterisk setup, you need the Asterisk-IM plugin. For Asterisk-IM to work, you'll have to enable the Asterisk Manager and add a user for Openfire. It's up to the Asterisk Manager to allow a client program such as Openfire to connect to an Asterisk instance and issue commands or read phone events over the network.

The Asterisk Manager is controlled via the `/etc/asterisk/manager.conf` file. To add an Openfire user, you'll have to add the following lines:

```
[ofadmin]
 secret = mysecret
 deny=0.0.0.0/0.0.0.0
 permit=192.168.2.7/255.255.255.0
 permit=127.0.0.1/255.255.255.0
 read = system,call,log,verbose,command,agent,user
 write = system,call,log,verbose,command,agent,user
```

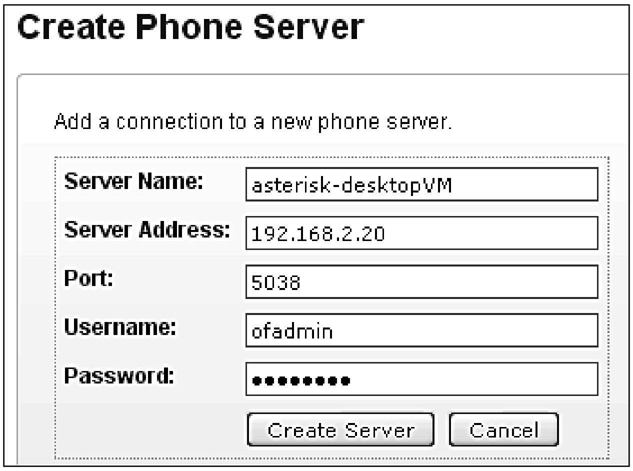
For more details, refer to the Asterisk Manager page at Voip-Info [`http://www.voip-info.org/wiki/view/Asterisk+manager+API`].

Next, head over to the Openfire admin console and install the Asterisk-IM plugin. Once you've installed the plugin, you'll notice that it creates an Asterisk-IM tab. The first thing you need to do is enable the plugin by selecting the **Enabled** option button and clicking the **Save** button at the bottom of the page. This enables the plugin.

Once the plugin is enabled, you need to add the details of your Asterisk VoIP server (refer to the following screenshot). You'll need the following details:

- The name of Asterisk Server. This can be arbitrarily chosen.
- The IP address of the Asterisk Server.
- The port number at which the Asterisk Server is accepting connections. The plugin defaults to 5038.
- The username and password to use to connect to the Asterisk Server. These are the ones specified in Asterisk's `manager.conf` file. In the example above, the username and password would be `ofadmin` and `mysecret` respectively.

When you've entered and saved the details, you're returned to the general settings page of the Asterisk-IM server.

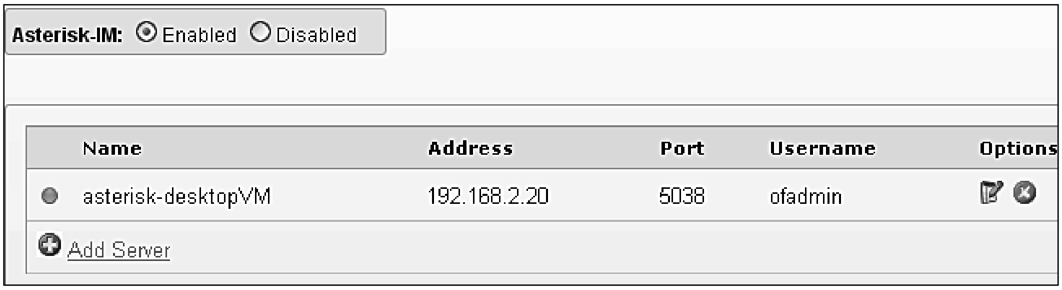


Create Phone Server

Add a connection to a new phone server.

Server Name:	asterisk-desktopVM
Server Address:	192.168.2.20
Port:	5038
Username:	ofadmin
Password:	●●●●●●●●●

Here you'll see the server you've just added. If the details you entered were correct and the server is able to communicate with the Asterisk server, there will be a green icon adjacent to this server (refer to the following screenshot).



Asterisk-IM: [✓ Enabled] Disabled

	Name	Address	Port	Username	Options
●	asterisk-desktopVM	192.168.2.20	5038	ofadmin	

On the other hand, if your server fails to show up, or has a gray light, make sure that you have used the correct Asterisk Manager settings. If you are using the built-in database, you'll need to update its database schema as mentioned on the Phone-71 bug page [`http://www.igniterealtime.org/issues/browse/PHONE-71`].

Summary

Using Openfire, we can not only enable members of an organization to communicate with each other but also go across company borders and communicate with users using publicly-accessible IM services. The best thing about doing this via Openfire is that all non-network conversations also follow the guidelines and rules we've set in earlier chapters.

In this chapter, we've covered the Gateway plugin which allows access to a variety of IM services including Yahoo! Messenger, IRC, ICQ, MSN Messenger, AOL Instant Messenger, and so on. Instead of just enabling access to these gateways, the plugin lets you configure several aspects of the service, such as whether or not you'd like users to receive email notifications or be able to view buddy icons. We also saw how you can restrict access to a gateway to all users, all users of a particular group, or to individual users. You can allow them to register themselves on the gateways or you can manually add them.

Because many organizations are now switching over to VoIP-based services for telephony, we've also covered how you can hook up Openfire to your Asterisk VoIP server.

8
Playing Big Brother

Increasing productivity and reducing communication costs via instant messaging introduces the problem of liability and responsible use. So playing big brother is an important function of running an instant messaging server. But snooping around the network looking at other folk's conversations on instant messages (IMs) will not win you any friends. However, it is your duty to ensure that your organization is in compliance with the federal laws that require corporate communications to be monitored and archived.

Electronic communications such as IMs, are being subject to an increasing number of industry and government regulations, especially as the regulatory bodies are of the opinion that IMs are no different from emails and therefore demand the same requirements for record keeping. The rules and degrees of compliance differ across sectors, and for those sectors involving money, such as banks and insurance companies, the rules tend to be stricter than for the others.

Irrespective of how sensitive your organization is to the issue, logging all conversations passing through the network helps, especially when a dispute arises, involving information given via an IM conversation. The logs help the organization by telling them what has been said, by whom, and when.

Open-Source (previously part of Openfire Enterprise edition) bundles tools (now available as plugins) that'll help you with compliance, and logs all conversations. To help the company prepare reports, the tools will help you retrieve logged information, based on various criteria. In this chapter, we'll also look at some of the freely available, third-party, user contributed tools that'll help you play big brother—but bring your own big arm chair!

In this chapter, we will discuss:

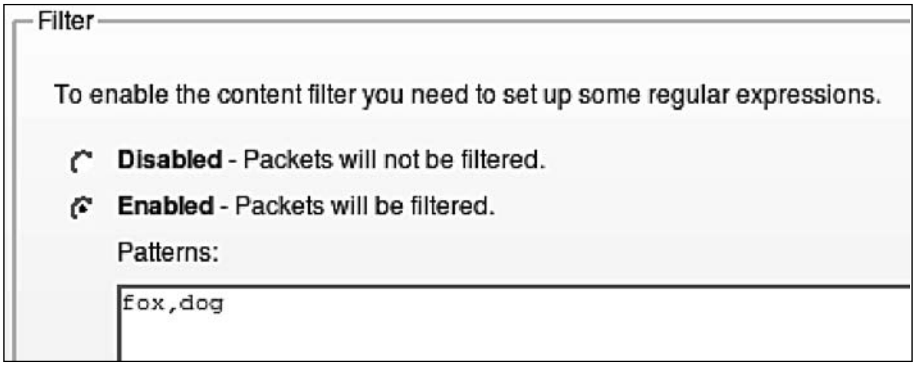
- How to monitor and filter content
- How to handle and deal with filtered content
- How to set up a message audit policy
- Tools to read audit logs and to search through message archives

Monitoring and Filtering Content

Monitoring conversations involves keeping an eye on what's being said over the network, and reporting policy abuse. Now, imagine doing it yourself—not only do you not win any friends but you will also kill whatever little social life you have. But a smart administrator lets the messenger do the hard work—Openfire can check messages and deliver those that meet specified criteria, and reject and report the ones that don't.

Of course, filtering objectionable IM content is important not only because it keeps the conversation clean, but also because it can be used to prevent sensitive information trespassing on your network via innocent IM conversations. But there's more to filtering conversation content than merely masking profanity. Maybe you don't want people sharing web links or non-company telephone numbers over IMs. Or maybe the company policy forbids discussing figures, budgets, and so on. Whatever the case may be, if you need a tool that'll keep an eye on all IM conversations and act when someone falls out of line, you need the **Content Filter** plugin.

Head over to the **Plugins** tab and grab the **Content Filter** plugin from the list of **Available Plugins**. To set up the filter, head back over to **Server | Server Settings**, where you'll notice a **Content Filter** option in the sidebar. This leads you to a page where you can specify the filter settings and how to react to a message when a positive match is detected.



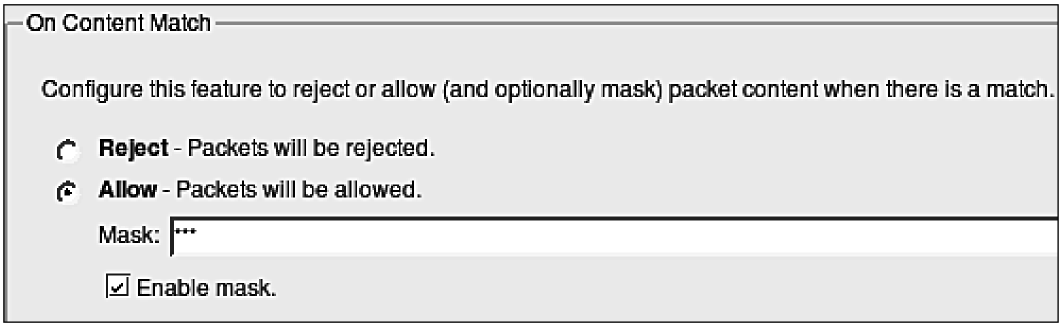
To enable the content filter you need to set up some regular expressions.

Disabled - Packets will not be filtered. [✓ Enabled - Packets will be filtered.]

Patterns:

```
fox,dog
```

In the first section of the filter page, you'll first have to enable the plugin by selecting the **Enabled** option button. Then, in the **Patterns** text box, specify the patterns that Openfire should look for in the packets being exchanged through it. By default, this box has two words, **fox,dog**. To test the plugin, let's leave this as it is while we explore the other options on this page.


On Content Match

Configure this feature to reject or allow (and optionally mask) packet content when there is a match.

Reject [✓ Allow]

Reject - Packets will be rejected.

Allow - Packets will be allowed.

Mask: ***

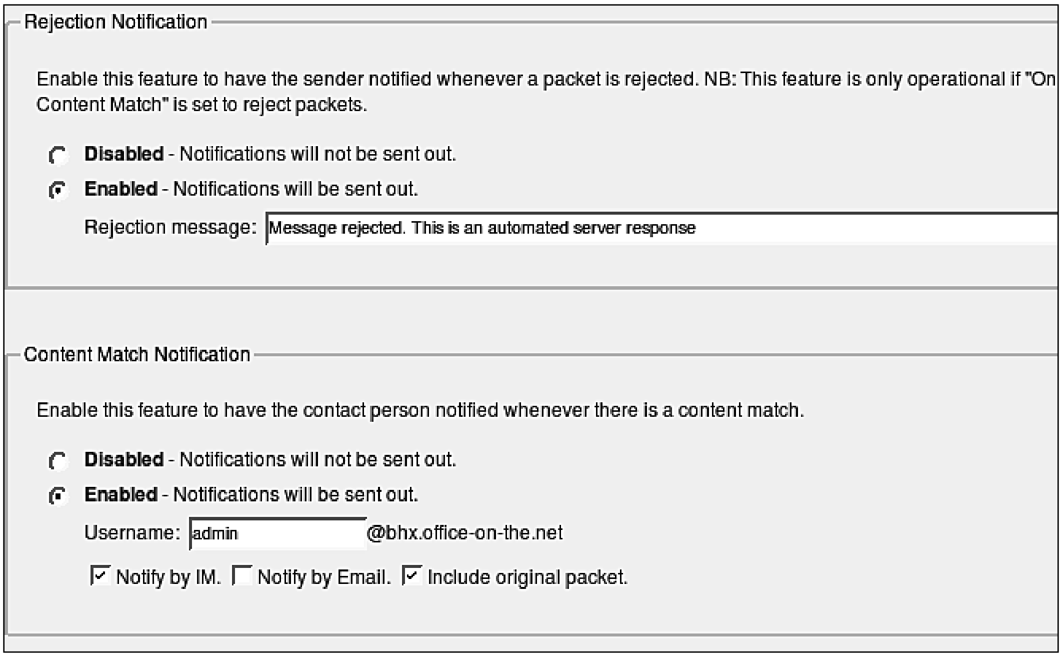
[✓] Enable mask.

The next section is called **On Content Match**, and as you might have guessed, here you choose the course of action when a positive match to the patterns specified by the administrator is found. The default action is to **Reject** the packets, which means packets that match the pattern are dropped and not delivered.

On the other hand, instead of rejecting the message you might want to censor the offending phrase. You can do so by first allowing message from the **On Content Match** section, and then selecting the **Enable Mask** option to mask the message. By default when you mask a message, Openfire replaces the offending keywords with a series of asterisks (***).

If you decide to reject the message, scroll down to the **Rejection Notification** section, and select whether or not you want to notify your users if their message has been rejected. By default, when a message is rejected, no notification is sent. But you can **Enable** the notification and edit the default **Rejection Message** as well, which, by default, reads **Message rejected. This is an automated server response**.

You can also configure the plugin to notify a designated person when there is a content match against the preset pattern. The **Content Match Notification** section lets you send out notifications via email, and/or IMs to the person whose username you specify. By default, it's sent out to you—the Openfire administrator. If you select the **Include original packet** checkbox, the plugin will also append the offending packet along with the notification.


Rejection Notification

Enable this feature to have the sender notified whenever a packet is rejected. NB: This feature is only operational if "On Content Match" is set to reject packets.

Disabled - Notifications will not be sent out. [✓ Enabled - Notifications will be sent out.]

Rejection message: Message rejected. This is an automated server response

Content Match Notification

Enable this feature to have the contact person notified whenever there is a content match.

Disabled - Notifications will not be sent out. [✓ Enabled - Notifications will be sent out.]

Username: admin @bhx.office-on-the.net

[✓ Notify by IM.] Notify by Email. [✓ Include original packet.]

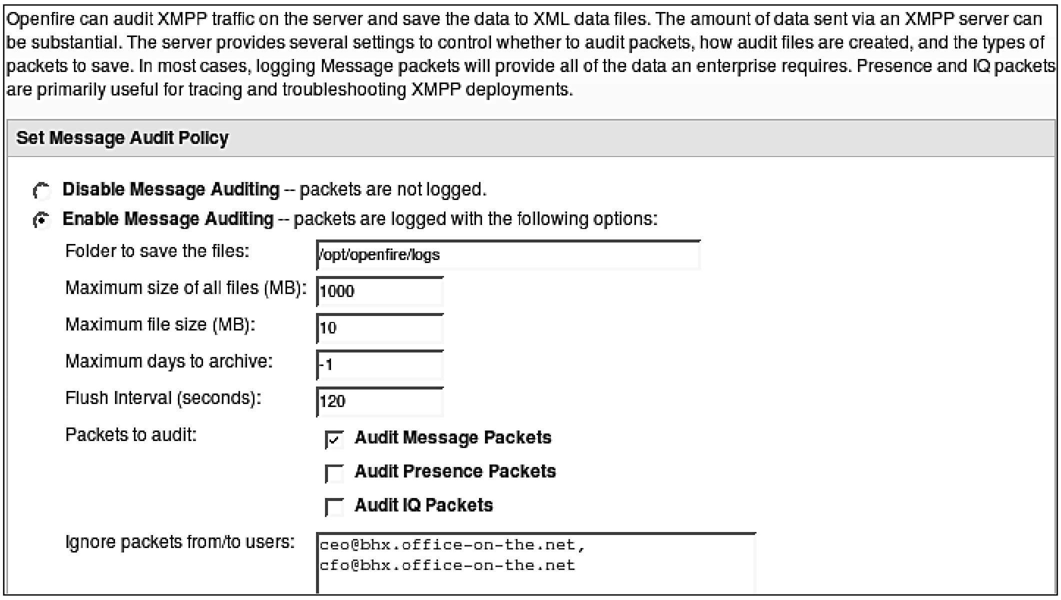
Once you've made your choices, save and apply them by clicking the **Save settings** button. The plugin also comes with a reset switch—the **Restore factory settings** button reverts all settings to their original states.

Auditing Messages

Filtering content is just part of the compliance procedure. Openfire is still not logging messages. In this section, we'll set up Openfire to first start logging messages, and then we'll deal with the tools that'll help you audit these logged conversations.

Setting up a Message Audit Policy

The first step in logging messages is to enable logging! This is easily done by heading over to **Server | Server Settings | Message Audit Policy**. From this section, you can configure Openfire to log all of the XMPP traffic in XML format.



Openfire can audit XMPP traffic on the server and save the data to XML data files. The amount of data sent via an XMPP server can be substantial. The server provides several settings to control whether to audit packets, how audit files are created, and the types of packets to save. In most cases, logging Message packets will provide all of the data an enterprise requires. Presence and IQ packets are primarily useful for tracing and troubleshooting XMPP deployments.

Set Message Audit Policy

- ○ **Disable Message Auditing** -- packets are not logged.
- ● **Enable Message Auditing** -- packets are logged with the following options:

Folder to save the files:	/opt/openfire/logs
Maximum size of all files (MB):	1000
Maximum file size (MB):	10
Maximum days to archive:	-1
Flush Interval (seconds):	120
Packets to audit:	✓ **Audit Message Packets** ☐ **Audit Presence Packets** ☐ **Audit IQ Packets**
Ignore packets from/to users:	ceo@bhx.office-on-the.net, cfo@bhx.office-on-the.net

In addition to clicking the **Enable Message Auditing** switch, you'll have to specify several settings before Openfire can begin logging messages. Some of the things you'll have to decide are:

- **Location of the logs:** The location on disk where you want to keep the logs. By default, they are inside the `logs` directory, which also houses the `error`, `warn`, and `info` logs, under Openfire installation directory. Depending on the number of users, the logs can become very huge in size.
- **Size of the logs:** You need to decide how much disk space you want to earmark for the conversation logs, as well as for individual conversations. The default size for all the logs is a whopping 1000 MB, with 10 MB for each individual log. These numbers might work for most setups, but if you have thousands of users on the network, and several hundreds using Openfire at any given moment, you'll have to cut out a larger piece of disk space for the logs.

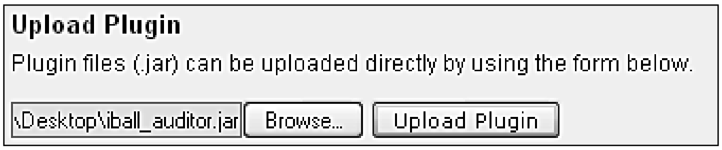
- **How long to keep the logs:** Some companies like to keep the logs forever, some for a year, while some others refresh their cache every week. By default, Openfire will keep the logs indefinitely (-1), but you are free to reduce this duration by specifying the logging period as a number of days.
- **What to log:** There's a lot of XMPP traffic flowing from Openfire to the clients. Openfire keeps track of information such as when a user came online, when he changed his presence status, and various other bits of information, thanks to the info/query (IQ) packets. You might not want to log all of this information, instead opting to log just the conversation packets.
- **Ignore some users:** Depending on your corporate policy, some users, such as the CEO, or members of the board of directors, might be exempted from having their messages logged. You can instruct Openfire to ignore messages sent from or to certain users and not to log them.

You can specify these settings under the **Set Message Audit Policy** form, after enabling auditing. Click on the **Save Settings** button to activate the new audit settings.

Reading Raw Audit Logs

The raw audit files are hard to read and filter, more so if you have multiple users and multiple conversations all messaging at the same time. But there are a few tools out there that'll help you visualize the logs and make them searchable. One such tool is the iBall plugin `(see http://www.suchtfaktorx.de/projects/iball).`

You can grab the latest version of the plugin (v2.0.1 at the time of writing this chapter) from the above website. This version is and known to work with Openfire v3.5.2. To install it from within the **Plugins** section of the Openfire admin interface, scroll down to the bottom of the plugins page where you'll notice the **Upload Plugin** section. Click on the **Browse** button to locate the plugin, and click on the **Upload Plugin** button to let Openfire upload and install it. Once it's uploaded, it'll appear in the list of installed plugins.

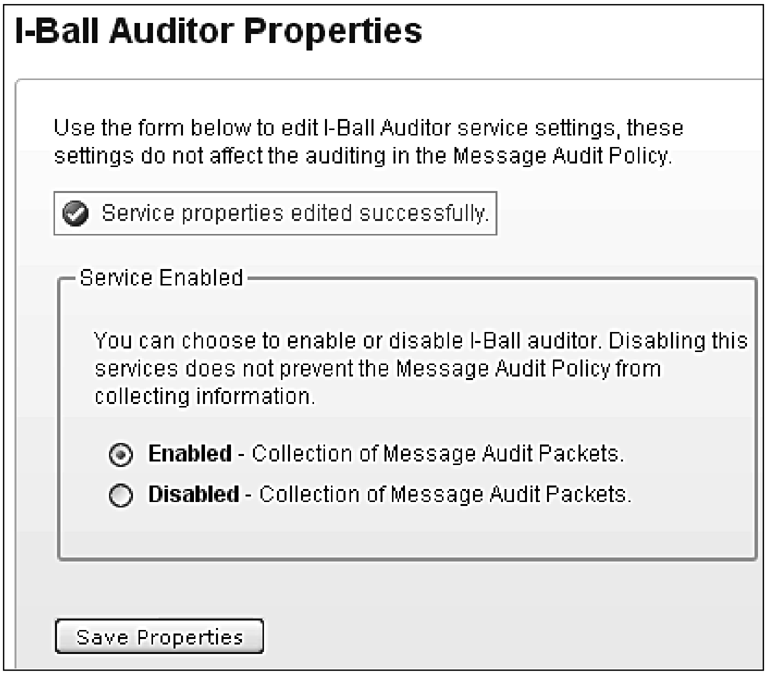


Upload Plugin

Plugin files (.jar) can be uploaded directly by using the form below.

\Desktop\iball_auditor.jar

To enable the plugin, head over to **Server** | **Server Settings** | **I-Ball Properties**, click on the **Enabled** option button and click on the **Save Properties** button to enable the plugin. Once the plugin has been successfully enabled, head over to **Users/Groups** | **Chat Log Search**.


I-Ball Auditor Properties

Use the form below to edit I-Ball Auditor service settings, these settings do not affect the auditing in the Message Audit Policy.

✓ Service properties edited successfully.

Service Enabled

You can choose to enable or disable I-Ball auditor. Disabling this services does not prevent the Message Audit Policy from collecting information.

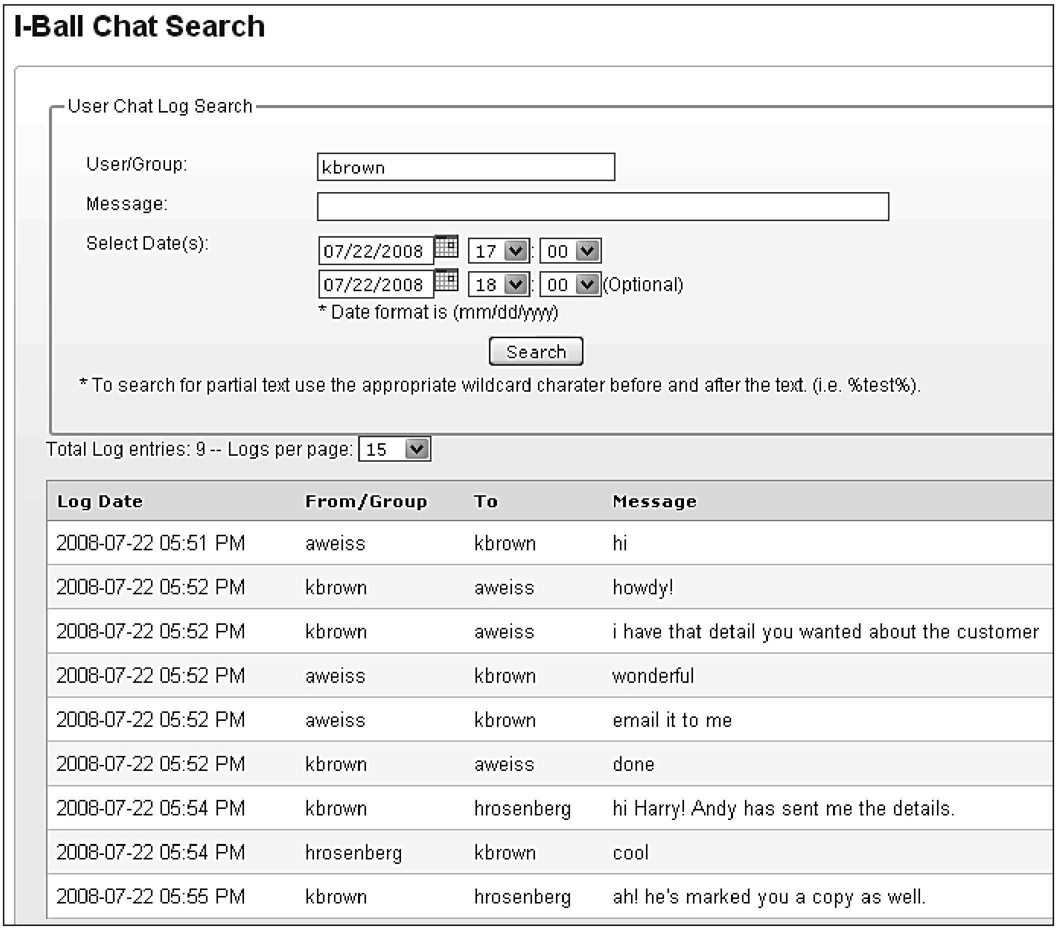
[✓ **Enabled** - Collection of Message Audit Packets.]

Disabled - Collection of Message Audit Packets.

From this interface, you can audit the logs by querying them in a variety of ways. You can track:

- All conversations involving a particular user
- All conversations that mention a particular word or string of words
- All conversations on or between particular dates, which can be narrowed down to particular hours and minutes as well

To further filter the logs, you can use a combination of any of the above.



I-Ball Chat Search

User Chat Log Search

User/Group: kbrown

Message:

Select Date(s): 07/22/2008 17 : 00
07/22/2008 18 : 00 (Optional)

* Date format is (mm/dd/yyyy)

* To search for partial text use the appropriate wildcard charater before and after the text. (i.e. %test%).

Total Log entries: 9 -- Logs per page: 15

Log Date	From/Group	To	Message
2008-07-22 05:51 PM	aweiss	kbrown	hi
2008-07-22 05:52 PM	kbrown	aweiss	howdy!
2008-07-22 05:52 PM	kbrown	aweiss	i have that detail you wanted about the customer
2008-07-22 05:52 PM	aweiss	kbrown	wonderful
2008-07-22 05:52 PM	aweiss	kbrown	email it to me
2008-07-22 05:52 PM	kbrown	aweiss	done
2008-07-22 05:54 PM	kbrown	hrosenberg	hi Harry! Andy has sent me the details.
2008-07-22 05:54 PM	hrosenberg	kbrown	cool
2008-07-22 05:55 PM	kbrown	hrosenberg	ah! he's marked you a copy as well.

For example, as shown in the previous screenshot, you can audit the logs for all conversations user **kbrown** had with others on July 22nd between 5 PM and 6 PM.

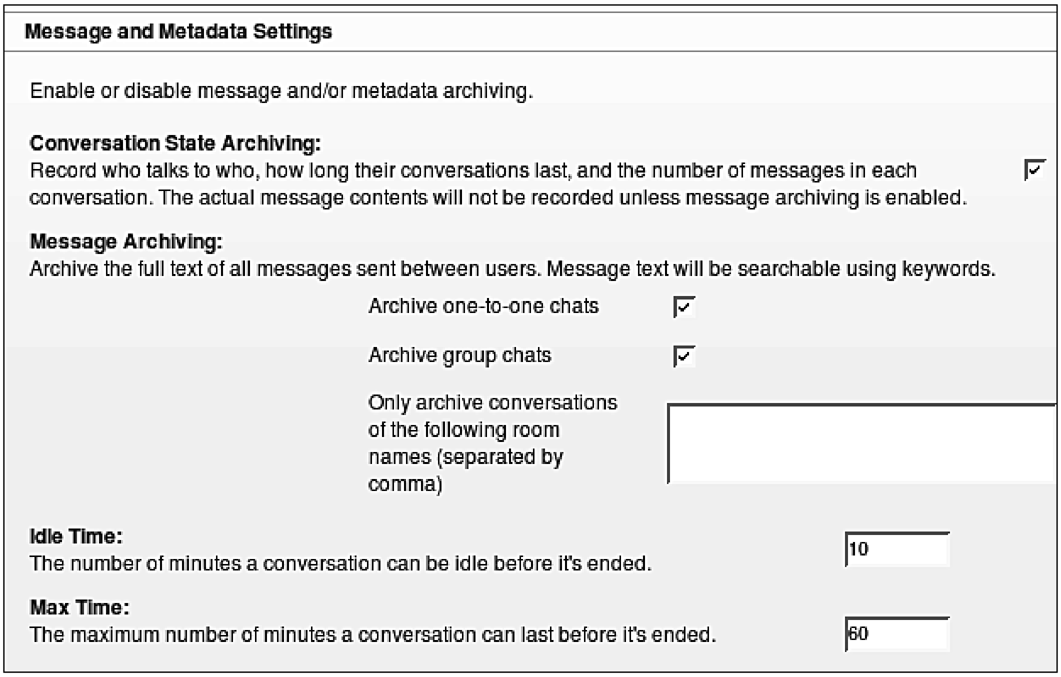
Advanced Archiving Settings

In addition to enabling Openfire to keep logs of all conversations, the **Monitoring Service** plugin (previously bundled as part of Openfire Enterprise edition, now installable from within the **Plugins** tab) offers some additional archiving options as well. The settings we just looked at creates raw XML logs of the conversation. However, if you are using the **Monitoring Service plugin**, you can archive the conversation with context, which makes it easier to trace, retrieve, and present, if the need arises.

Using the Monitoring Service plugin's archiving settings, you can:

- Log and retrieve individual one-to-one conversations
- Know at a glance how long each conversation lasted
- Know the number of messages exchanged in the conversation
- Archive particular rooms or all group chats
- Limit the duration of conversations

To set these parameters, head over to **Server** | **Archiving** | **Archive Settings**. Use the **Message and Metadata Settings** form and enable **Conversation State Archiving** if you want to add the relevant context (who talks to whom, the duration of the conversation, and so on) to all conversations.



Message and Metadata Settings

Enable or disable message and/or metadata archiving.

Conversation State Archiving:
Record who talks to who, how long their conversations last, and the number of messages in each conversation. The actual message contents will not be recorded unless message archiving is enabled. ✓

Message Archiving:
Archive the full text of all messages sent between users. Message text will be searchable using keywords.

Archive one-to-one chats ✓

Archive group chats ✓

Only archive conversations of the following room names (separated by comma)

Idle Time:
The number of minutes a conversation can be idle before it's ended. 10

Max Time:
The maximum number of minutes a conversation can last before it's ended. 60

Using the checkbox next to the options, you can also decide to **Archive one-to-one chats**, or **Archive group chats**, or both. If instead of logging all group chats, you only want to archive specific rooms, you can specify the rooms to be archived in the text box provided on this page.

You can also fix two time variables—one is the **Idle Time** setting which specifies the number of minutes after which an idle conversation is considered closed. The other setting is **Max Time, which** specifies the number of minutes after which a conversation, active or idle, is automatically ended. Both of these options keep the conversation logs neat, concise, and easier to browse.

When you've tweaked the options to your liking, apply them by clicking on the **Update Settings** button. As soon as the Monitoring Service plugin starts to log messages, this page expands to show you brief statistics of the search index. The **Index Settings** box informs you about:

- The current size of the search index
- The total number of archived conversations
- The total number of archived messages

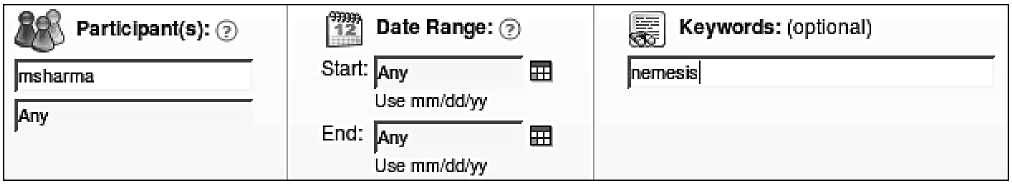
If you have active conversations while you are looking at this page, you can use the **Rebuild Index** button to update these settings to reflect the state of the current conversations.

Searching Archives

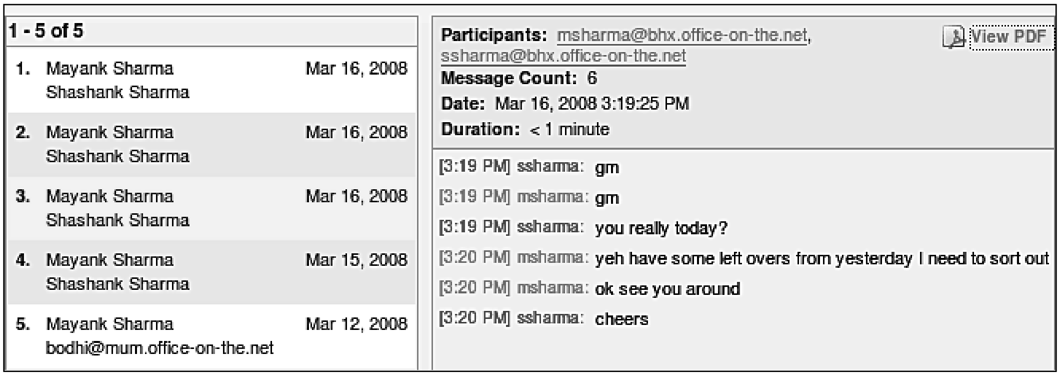
Alright, so you have enabled logging and have been archiving conversations. Now roll out the big arm chair and call in the participants—let us start snooping.

In their raw forms, the XML IM logs are difficult to read. Luckily, the Monitoring Service plugin offers a very slick interface that'll help you search archived conversations.Using this interface, you can skip the details and narrow down to the relevant portion of the log. You can filter the logs based on these criteria:

- **By Participants:** Generally, you'd want to look at the conversation history of someone in particular. Using the interface you can look at the conversation history of an individual. Better still, you can go deeper and look at the conversation between two specific people.
- **By Date:** Want to go over conversations that happened on this day last year? No problem. From the interface, you can search the conversation logs on a particular date or between a range of dates.
- **By Keyword:** Depending on why you are looking at the conversation archives, searching by keywords can be another important filtering technique.
- **By a combination of all three:** The best aspect about the search interface is that you can use all three search narrowing parameters together. In this way you can look at conversation records between two people, during a particular time period, when they discussed something in particular, say a particular project.


Participant(s): msharma Any
Date Range: Start: Any Use mm/dd/yy End: Any Use mm/dd/yy
Keywords: (optional) nemesis

Enough talk! Let's put this to the test. Head over to the interface at **Server | Archiving | Search Archive**. Enter your criteria and click the **Search** button, and let Openfire sniff around the logs. If something matching your search string turns up, Openfire expands the page and displays the archived conversations according to the names of the participants and the date on which the conversation took place.

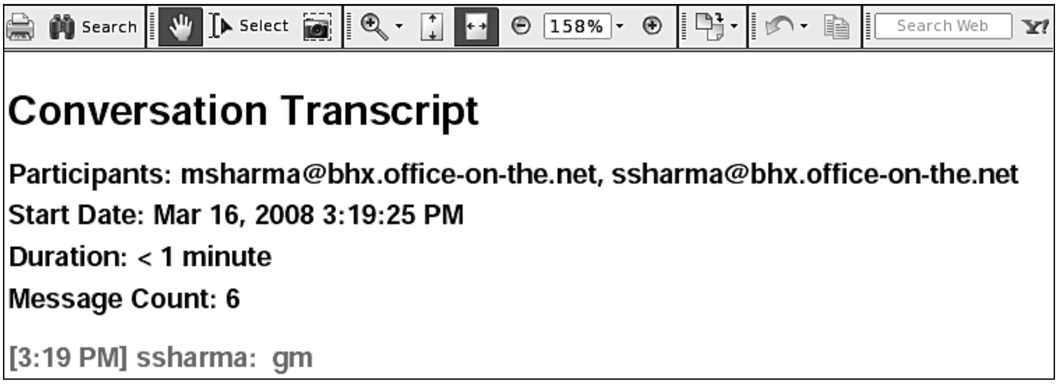

1 - 5 of 5

1.	Mayank Sharma Shashank Sharma	Mar 16, 2008
2.	Mayank Sharma Shashank Sharma	Mar 16, 2008
3.	Mayank Sharma Shashank Sharma	Mar 16, 2008
4.	Mayank Sharma Shashank Sharma	Mar 15, 2008
5.	Mayank Sharma bodhi@mum.office-on-the.net	Mar 12, 2008

Participants: msharma@bhx.office-on-the.net, ssharma@bhx.office-on-the.net
View PDF
Message Count: 6
Date: Mar 16, 2008 3:19:25 PM
Duration: < 1 minute

```
[3:19 PM] ssharma: gm
[3:19 PM] msharma: gm
[3:19 PM] ssharma: you really today?
[3:20 PM] msharma: yeh have some left overs from yesterday I need to sort out
[3:20 PM] msharma: ok see you around
[3:20 PM] ssharma: cheers
```

To look at the exact conversation, click on any of the search results. This opens the conversation in the pane on the rightmost-side of the page. In addition to the text of the conversation, which, by the way, is time stamped, you also get a list of participants in the conversation, the total number of messages exchanged, the date and time at which the conversation took place, and the duration of the conversation.



Conversation Transcript

Participants: msharma@bhx.office-on-the.net, ssharma@bhx.office-on-the.net
Start Date: Mar 16, 2008 3:19:25 PM
Duration: < 1 minute
Message Count: 6

[3:19 PM] ssharma: gm

And if you thought this was amazing, you haven't noticed the **View PDF** link at the top-right corner of the pane. When clicked, Openfire creates a PDF file of this particular conversation with all the relevant information. You can then save the PDF, for passing on as an email attachment, or print it out for filing.

Light-Weight Third-Party Archiving Apps

Impressed by the archiving facility of the Monitoring Service plugin? If you are in a sector that takes compliance with the IM regulations seriously, there's no better way to do it than through the archiving interface of the Monitoring Service plugin. But what happens if your archiving demands are more modest? What if you are a small NGO or charity that doesn't really need the powerful features of the Monitoring Service plugin, but wouldn't mind a simple interface to archive and access old conversations? Well, thanks to the Openfire community, you can have your cake and eat it too—albeit minus a few cherries.

Using Open Archive

Open Archive is an open-source-community plugin that tries to duplicate the look and feel of the Monitoring Service plugin's archive function. And it does a pretty good job of it. But there's a big difference between Open Archive and the Monitoring Service plugin. To quote the Open Archive website:

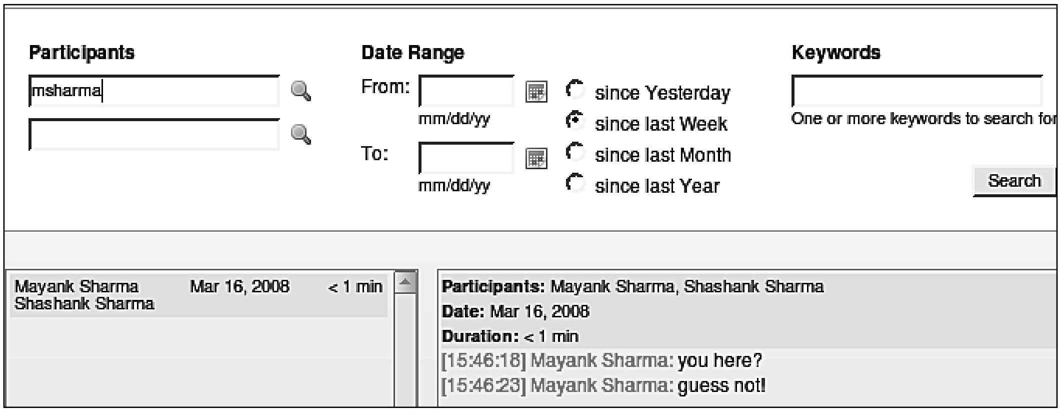
> *While Open Archive is basically a server side storage for users' messages, the enterprise (now open source as the Monitoring Service plugin) archive offered by Jive is a tool to satisfy compliance requirements by reliably saving messages for a long time. Open Archive gives users full control over their messages so they are able to remove them at any time or disable archiving completely.*

What this means is that users on the network can exclude themselves from being logged and can also "recall" their messages from the server. This is possible due to the experimental XMPP mechanism known as XEP-0136. Needless to say, clients will only be able to recall their messages using a XEP-0136 compliant client.

To install the plugin, download the latest version (v1.0.4 at the time of writing) from its website (see `http://maven.reucon.com/projects/public/archive/`). Openfire plugins are written in Java and distributed as ready-to-use `.jar` files. Once you have the Open Archive `.jar` file, copy it into the `plugins` directory under the Openfire installation directory. Once it's there, it'll automatically be listed under the installed plugins list in the Openfire admin interface.

Once it's installed, you can activate it by heading over to **Server | Server Settings | Archive Settings** in the Openfire admin interface. Activating the **Archive Settings** plugin simply involves selecting the **Enabled** option button to let the plugin log messages. Unlike the Monitoring Service plugin archive options, you only get to configure one parameter—the duration (in minutes) before an idle conversation is considered closed. Once you've enabled logging and entered an idle-conversation-disconnect duration, save the settings by clicking on the **Save Settings** button.

Now comes the important bit—searching archives. Head over to **Server | Server Settings | Search Archive** in the Openfire admin interface. The search archive interface of the Open Archive plugin looks similar to the Monitoring Service archive search interface. You can search either by one or both participants in the conversation, a single date or a date range, one or more keywords, or a combination of all three.



Mayank Sharma Shashank Sharma Mar 16, 2008 < 1 min

Participants: Mayank Sharma, Shashank Sharma
Date: Mar 16, 2008
Duration: < 1 min
[15:46:18] Mayank Sharma: you here?
[15:46:23] Mayank Sharma: guess not!

Additionally, you have four interesting quick-access options. You can choose to view any of the following conversations:

- **since Yesterday**
- **since last Week**
- **since last Month**
- **since last Year**

Although you can select only one of these four options, you can combine them with two other search options (participants and keyword, but not the date range).

The four date quick access options are only a convenience addition. From the Monitoring Service plugin, you can replicate and search within these parameters by using the appropriate dates in the date range option. Having the buttons, though, makes the task simpler and quicker.

Once you've found a conversation that you would like to view in detail, click on it. As with the Monitoring Service plugin, Open Archive displays the time-stamped content of the conversation, along with the name of the participants, the date, and the duration of the conversation. Unlike the Monitoring Service plugin, though, which is specifically designed for compliance purposes, you don't get the option to view and save the conversation archive in PDF format.

Summary

Archiving conversations is an important part of running an instant messaging server. This is particularly true in environments that need to adhere to strict industry regulations governing the usage of IMs. With the regulations treating IM as email, archived conversations can also be used later to resolve internal or even external disputes. However, conversation archives are difficult to read in their raw format because of the sheer volume of data they log. To help us sift through the logs we've used the iBall plugin.

Furthermore, the Monitoring Service plugin (previously part of the Openfire Enterprise edition) comes bundled with a very comprehensive archiving and searching mechanism. In this chapter, we've seen how to archive one-on-one, group, and room conversations. The plugin further shows off its flexibility by allowing us to exclude some users from the archiving process. But archiving conversations is just half the job. This chapter also explained the Monitoring Service plugin's archived conversation searching interface. This interface lets you retrieve conversations by cleverly using certain parameters such as dates and keywords.

For users in less demanding areas, we've also looked at some of the freely-available archiving options. Leading the pack here is Open Archive. This plugin is easy to configure and has a search interface similar to the Monitoring Service plugin's archived conversation search interface.

9 Large-scale Openfire

Do you administer a huge corporation? Wonder if Openfire will be able to service all of your users, from the geeks in the fourth floor to the bosses in the penthouse? Fear not. Throughout the book I've been praising Openfire's ability to scale and accommodate thousands of users with the same ease and effortlessness with which it handles a handful. In this chapter, we put that claim to test. We'll learn to setup Openfire to handle thousands of users concurrently.

First and foremost, if you want so many users concurrently interacting through Openfire, you'll have to employ an external database. Openfire's in-built HSQLDB works, but several people on the Openfire forum suggest that HSQL should not be used in a production environment even with a small number of users. As we are talking about several thousand users exchanging data at the same time, it's best to use a database equipped to handle such extreme loads with ease. Openfire supports some of the best known and popular heavy-weights of the database world—MySQL, PostgreSQL, Oracle, IBM DB2, and Microsoft SQLServer.

In this chapter, we will cover:

- Using an external database with Openfire
- Setting up an Openfire cluster
- Scaling Openfire with Connection Managers

Enabling Database Support in Openfire

We are given the option to use an external database when setting up Openfire, right after installation. Because we can't go back in time, we'll do the next best thing—re-run the Openfire setup again. We've done this before, in Chapter 5, when we hooked our Openfire installation upto a directory server. No need to flip back to that chapter; I'll retrace the steps here.

To re-run the "initial" setup again, first stop the OpenFire server if it's running. Then open a terminal and navigate to the OpenFire server installation directory. Change to the `conf/` directory of the Openfire installation and open the `openfire.xml` file in your favorite text editor. Look for a line that reads "`<setup>true</setup>`" and change it to `<setup>false</setup>`. Save the file, restart the server, and head back to the administration console. Presto! You're back at the setup interface.

But before you run through the setup wizard, you'll have to prepare your database and keep the database server settings handy. Briefly, the steps involved in setting up any database to work with Openfire are:

1. Create a database.
2. Run the appropriate database scheme from under the `resources/database` directory of the Openfire installation.
3. Specify the driver and server information in the Openfire setup tool.

Once your database is ready, you can head over to the Openfire setup tool. In the database setup step, you'll be asked to provide the following information:

1. JDBC Driver Class: This is the class name of the JDBC driver that talks to the database. The drivers for the databases are bundled with Openfire.
2. Database URL: This is the URL to the database server.

Let's look at these steps in detail for the two most popular Open Source databases, MySQL, and PostgreSQL. If you want to use one of the other supported databases with Openfire, follow the instructions in the Openfire Database Installation Guide available on the Openfire website (see `http://www.igniterealtime.org/builds/openfire/docs/latest/documentation/database.html#hsql`). On the other hand, if you have user data in a custom database, then follow the instructions in the Openfire Database Integration Guide (see `http://www.igniterealtime.org/builds/openfire/docs/latest/documentation/db-integration-guide.html`).

Setting up MySQL

Assuming that you have the MySQL database up and running, first create a database for Openfire:

```
# mysqladmin -u mysqlroot -p create openfire
```

The above command assumes that the `mysqlroot` user can create MySQL databases. After prompting for the password for this account, the mysqladmin tool creates a database called "openfire".

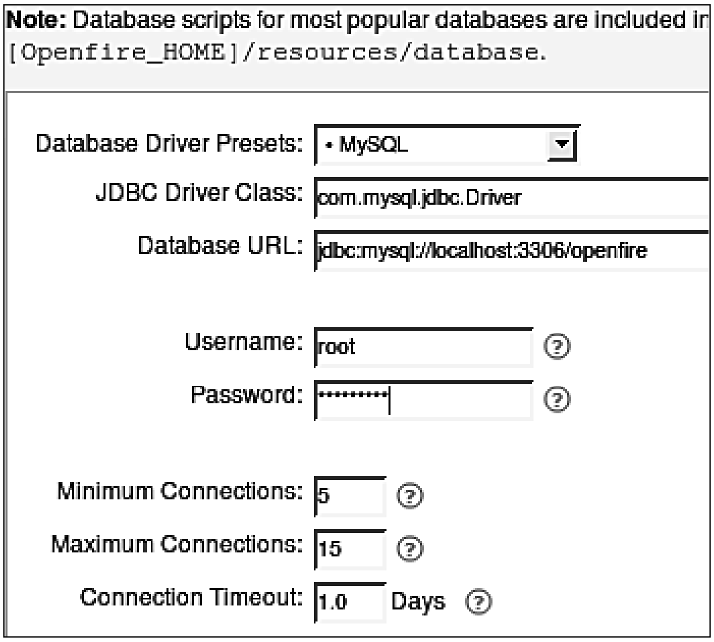
Then change to the directory that houses the database schemas and copy them over to MySQL:

```
# cd <openfire-installation>/resources/database
# mysql -u mysqlroot -p openfire < openfire_mysql.sql
```

Again, this command prompts you for the password for the `mysqlroot` user, and then creates the appropriate tables from the schema. If you are running the MySQL server on Windows, you'll need to use the following command to create the tables from the schema:

```
type openfire_mysql.sql | mysql openfire;
```

Then, in the Openfire setup tool, select the MySQL JDBC driver (**com.mysql.jdbc.Driver**) and enter the database URL in the format `jdbc:mysql://[host-name]:3306/[database-name]` (refer to the following screenshot). Substitute host-name and database-name with the appropriate values.



Note: Database scripts for most popular databases are included ir
[Openfire_HOME]/resources/database.

Database Driver Presets:	• MySQL
JDBC Driver Class:	com.mysql.jdbc.Driver
Database URL:	jdbc:mysql://localhost:3306/openfire
Username:	root
Password:	••••••••••
Minimum Connections:	5
Maximum Connections:	15
Connection Timeout:	1.0 Days

You're done! Continue with the setup until it lets you log in to the administration console.

Setting up PostgreSQL

Once you have PostgreSQL running in your network, start by creating a database:

```
# su - postgres
-bash-3.1$ createuser ofadmin
Shall the new role be a superuser? (y/n) y
CREATE USER
-bash-3.1b$ createdb -E UNICODE openfire
CREATE DATABASE
-bash-3.1b$ exit
# exit
```

In the above set of commands, we've created a new PostgreSQL administrator and then followed this up by creating a new database called "openfire" that supports Unicode character encoding. This allows it to store and archive multilingual text.

Follow this up by importing the schema from the `resources/database` directory of the Openfire installation using the following command:

```
# psql -d openfire -f <openfire-installation/resources/database/
openfire_postgresql.sql
```

This sets up the database. Now head over to the Openfire setup tool where you'll be prompted for a PostgreSQL JDBC driver (`org.postgresql.Driver`) and the database URL, which is to be specified in the format `jdbc:postgresql://[host-name]:5432/[database-name]`, substituting host-name and database-name with the appropriate values.

Troubleshooting Database Errors

Although there's hardly anything to setting up Openfire to delegate record-keeping to an external database, there are a couple of common errors that might prevent this from happening. When you try to connect to the database through Openfire, you may get a "java.net.ConnectException: Connection refused" error. There could be two reasons for this error. One, that you have provided the wrong database connection information and two, that your database server isn't listening to connection on the network.

If your connection is being refused, it's a good idea to first make sure that you supplied the correct information for the database name and the username and password details of the user that can connect to the database. If it still doesn't work, look at the error logs carefully, and you might find a line that reads something like "message from server: Host 'xxx.xxx.x.x' is not allowed to connect to this MySQL server". To avoid this error, you can add a user to your MySQL database list of users to allow access the database from any host on the network.

```
mysql> GRANT ALL PRIVILEGES ON *.* TO 'lore'@'%'  IDENTIFIED BY 'some_
pass' WITH GRANT OPTION;
```

Make sure the username and password of this user are the same as the username and password of the user that can access the database from the localhost.

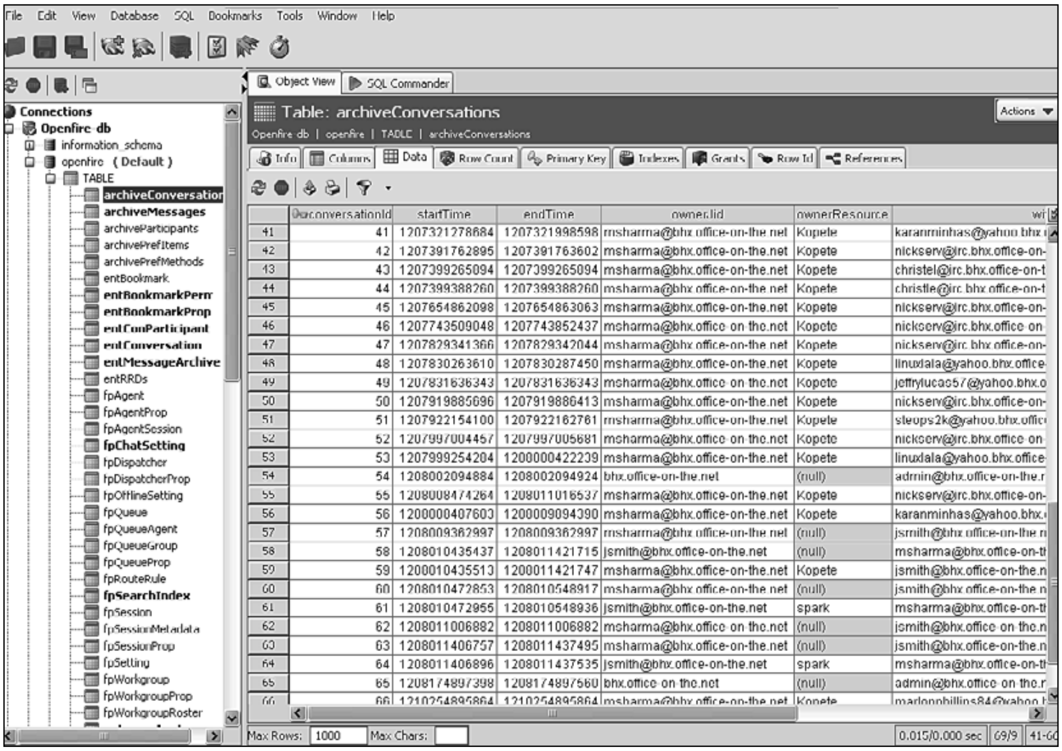
On the other hand, if the user information you've provided is correct, you'll have to tweak your database settings to make sure that it is listening on a TCP/IP network port. PostgreSQL can either start the postmaster with the `-i` flag or specify the list of addresses to listen to using the `-h` option or in the `postgresql.conf` configuration file.

Similarly, MySQL, by default, also only allows connections through a socket. MySQL users need to edit their MySQL configuration file (`my.cnf`) and comment out the "skip-networking" option, which then forces MySQL to listen to a TCP/IP port for connections. You'll have to restart the MySQL server after making changes to the `my.cnf` file.

Monitoring the Database

Once your database is setup, it's a good idea to keep a tab on its health. Monitoring the database also helps you track the load on the database. Furthermore, if something goes wrong, for example, deleted users aren't actually deleted, you need to make sure that the database is carrying out your instructions.

There are various tools that'll help you do this for all sorts of databases on both Linux and Windows platforms. Although phpMyAdmin (see `http://www.phpmyadmin.net/`) seems to be the favorite graphical administration tool for managing MySQL, PostgreSQL users can also use pgAdmin (see `http://www.pgadmin.org/`), if they prefer a graphical tool. You could also use a third-party tool, such as DbVisualizer (see `http://www.minq.se/products/dbvis/`), that supports most major databases (see the following screenshot).

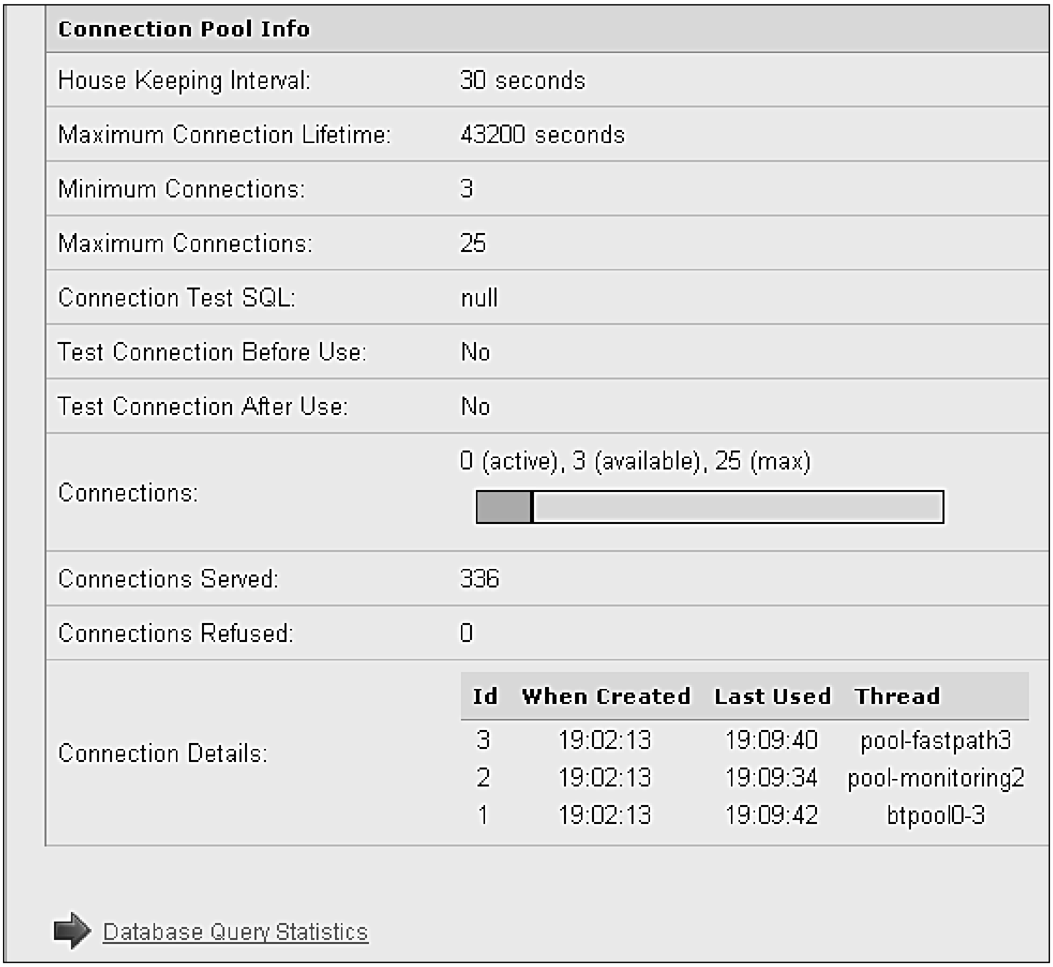


	conversationId	startTime	endTime	ownerJid	ownerResource	wi
41	41	1207321278684	1207321998598	msharma@bhx.office-on-the.net	Kopete	karanminhas@yahoo.bhx.
42	42	1207391762895	1207391763602	msharma@bhx.office-on-the.net	Kopete	nickserv@irc.bhx.office-on-
43	43	1207399265094	1207399265094	msharma@bhx.office-on-the.net	Kopete	christel@irc.bhx.office-on-t
44	44	1207399388260	1207399388260	msharma@bhx.office-on-the.net	Kopete	christle@irc.bhx.office-on-t
45	45	1207654862098	1207654863063	msharma@bhx.office-on-the.net	Kopete	nickserv@irc.bhx.office-on-
46	46	1207743509048	1207743852437	msharma@bhx.office-on-the.net	Kopete	nickserv@irc.bhx.office-on-
47	47	1207829341366	1207829342044	msharma@bhx.office-on-the.net	Kopete	nickserv@irc.bhx.office-on-
48	48	1207830263610	1207830287450	msharma@bhx.office-on-the.net	Kopete	linuxlala@yahoo.bhx.office-
49	49	1207831636343	1207831636343	msharma@bhx.office-on-the.net	Kopete	jeffrylucas57@yahoo.bhx.o
50	50	1207919885696	1207919886413	msharma@bhx.office-on-the.net	Kopete	nickserv@irc.bhx.office-on-
51	51	1207922154100	1207922162761	msharma@bhx.office-on-the.net	Kopete	steops2k@yahoo.bhx.offic
52	52	1207997004457	1207997005681	msharma@bhx.office-on-the.net	Kopete	nickserv@irc.bhx.office-on-
53	53	1207999254204	1208000422239	msharma@bhx.office-on-the.net	Kopete	linuxlala@yahoo.bhx.office-
54	54	1208002094884	1208002094924	bhx.office-on-the.net	(null)	admin@bhx.office-on-the.r
55	55	1208008474264	1208011016537	msharma@bhx.office-on-the.net	Kopete	nickserv@irc.bhx.office-on-
56	56	1208008407603	1208009094390	msharma@bhx.office-on-the.net	Kopete	karanminhas@yahoo.bhx.
57	57	1208009362997	1208009362997	msharma@bhx.office-on-the.net	(null)	jsmith@bhx.office-on-the.n
58	58	1208010435437	1208011421715	jsmith@bhx.office-on-the.net	(null)	msharma@bhx.office-on-th
59	59	1208010435513	1208011421747	msharma@bhx.office-on-the.net	Kopete	jsmith@bhx.office-on-the.n
60	60	1208010472853	1208010548917	msharma@bhx.office-on-the.net	(null)	jsmith@bhx.office-on-the.n
61	61	1208010472955	1208010548936	jsmith@bhx.office-on-the.net	spark	msharma@bhx.office-on-th
62	62	1208011006882	1208011006882	msharma@bhx.office-on-the.net	(null)	jsmith@bhx.office-on-the.n
63	63	1208011406757	1208011437495	msharma@bhx.office-on-the.net	(null)	jsmith@bhx.office-on-the.n
64	64	1208011406896	1208011437535	jsmith@bhx.office-on-the.net	spark	msharma@bhx.office-on-th
65	65	1208174897398	1208174897560	bhx.office-on-the.net	(null)	admin@bhx.office-on-the.r
66	66	1210254895864	1210254895864	msharma@bhx.office-on-the.net	Kopete	marlonphillips84@yahoo.b

Max Rows: 1000 Max Chars: 0.015/0.000 sec 69/9 41-6

Openfire, too, bundles a simple interface for checking up on the database. When you head over to the Database Properties section in the admin interface via Server **Manager | Database**, you'll see a detailed list of properties for your database including the database connection details (refer to the following screenshot).

At the bottom of that page, you'll notice the **Database Query Statistics** link.



Connection Pool Info

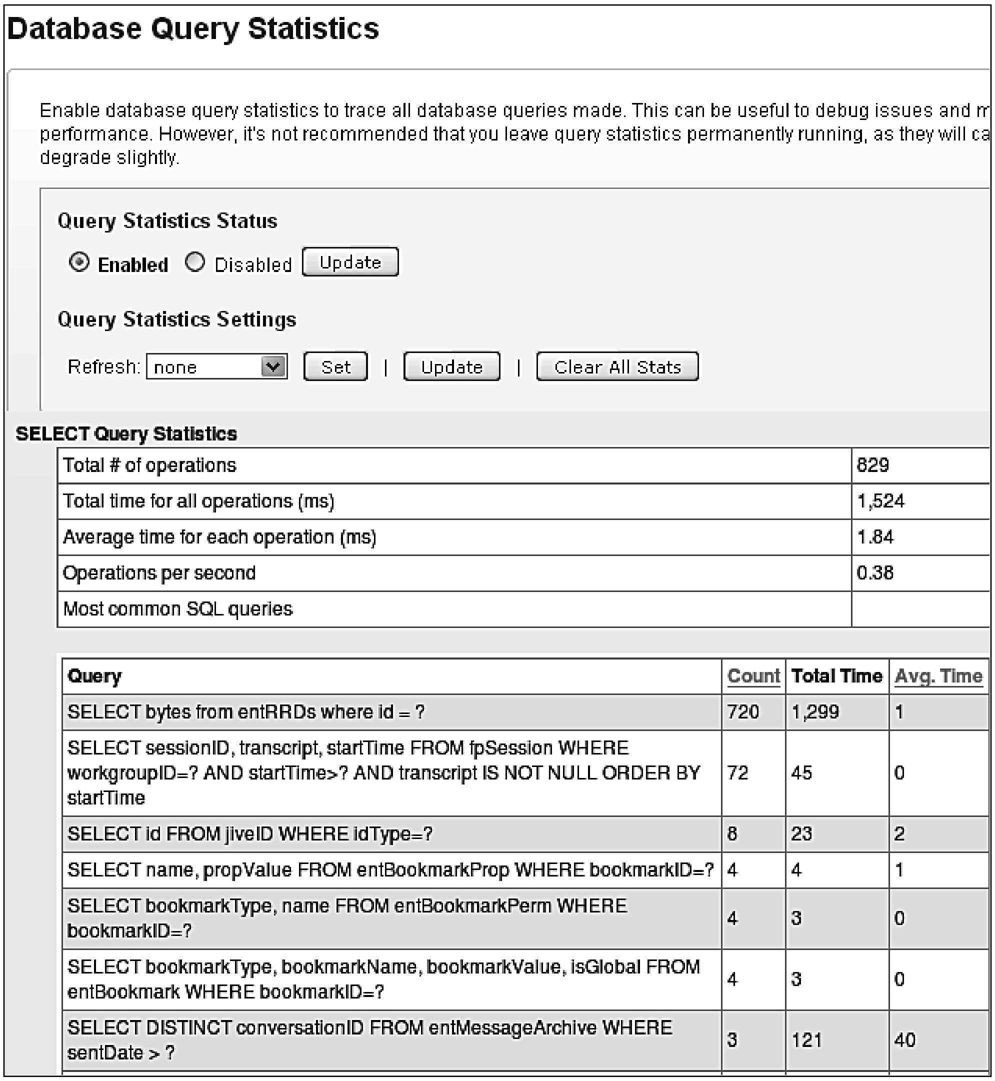
House Keeping Interval:	30 seconds
Maximum Connection Lifetime:	43200 seconds
Minimum Connections:	3
Maximum Connections:	25
Connection Test SQL:	null
Test Connection Before Use:	No
Test Connection After Use:	No
Connections:	0 (active), 3 (available), 25 (max)
Connections Served:	336
Connections Refused:	0

Connection Details:

Id	When Created	Last Used	Thread
3	19:02:13	19:09:40	pool-fastpath3
2	19:02:13	19:09:34	pool-monitoring2
1	19:02:13	19:09:42	btpool0-3

Database Query Statistics

By default, the query statistics function is disabled. You can select the **Enabled** option button to turn it on (refer to the following screenshot). Once enabled, you can set a refresh interval after which the statistics are refreshed.



Database Query Statistics

Enable database query statistics to trace all database queries made. This can be useful to debug issues and m
performance. However, it's not recommended that you leave query statistics permanently running, as they will ca
degrade slightly.

Query Statistics Status

[✓ Enabled] Disabled [Update]

Query Statistics Settings

Refresh: none [Set] | [Update] | [Clear All Stats]

SELECT Query Statistics

Total # of operations	829
Total time for all operations (ms)	1,524
Average time for each operation (ms)	1.84
Operations per second	0.36
Most common SQL queries	

Query	Count	Total Time	Avg. Time
SELECT bytes from entRRDs where id = ?	720	1,299	1
SELECT sessionID, transcript, startTime FROM fpSession WHERE workgroupID=? AND startTime>? AND transcript IS NOT NULL ORDER BY startTime	72	45	0
SELECT id FROM jiveID WHERE idType=?	8	23	2
SELECT name, propValue FROM entBookmarkProp WHERE bookmarkID=?	4	4	1
SELECT bookmarkType, name FROM entBookmarkPerm WHERE bookmarkID=?	4	3	0
SELECT bookmarkType, bookmarkName, bookmarkValue, isGlobal FROM entBookmark WHERE bookmarkID=?	4	3	0
SELECT DISTINCT conversationID FROM entMessageArchive WHERE sentDate > ?	3	121	40

This page offers various types of statistics about the database, grouped into four categories based on the SQL query—SELECT, INSERT, UPDATE, DELETE. For each type of SQL query, you can see the total number of queries, the time taken to complete all operations, the average time for each operation, the average number of operations completed per second, and the most common SQL queries, which are listed with their respective counts, total, and average time.

Scaling Openfire

Once you have the database in place, it's time to enable Openfire to scale. There are two methods that you can use to enable Openfire to scale and handle a large number of users. One of the methods involves forming a cluster of Openfire servers and nodes. Due to the way the cluster is implemented in Openfire Enterprise, which we'll see and discuss later, this solution requires an external database. The other scaling method involves deploying a bunch of connection managers. This solution was borne out of an old Openfire limitation, but lives on as a cost-effective solution to equip Openfire to handle thousands of users.

In the following sections we'll look at setting up both of these implementations as well as optimizing them to meet our needs via other network services.

Setting up Clustering

There are two benefits to setting up clustering. In addition to setting up scalability, the other benefit is availability. By making a cluster, we are in essence setting up multiple points of failure. In the event that one node goes down, our users won't lose connectivity because they can simply jump to another available node.

Setting up clustering isn't a complex task, but it requires you to have an Enterprise license for Openfire.

> The Enterprise plugin has been discontinued, but till the final stages of this book, the Openfire developers had not yet released an Open Source version of the clustering component. If you wish to use Openfire in a cluster, please contact the Openfire developers.

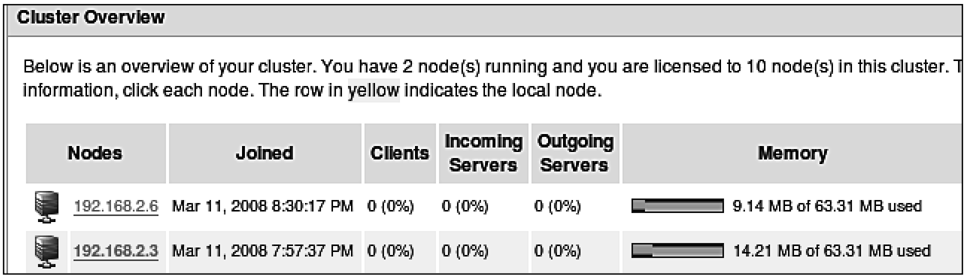
Assuming you have the Enterprise license, follow these instructions:

1. Install Openfire server on multiple machines. Identify the machines that you want to turn into a cluster of Openfire servers and install Openfire on each one of them.
2. When setting up the various servers, make sure that they all use the same XMPP domain and all connect to the same external database.
3. Download the Openfire Enterprise plugin (distributed as a `.jar` file) and upload it from the plugins sections. Activate the plugin using your Enterprise license.

At this point, you have several machines running Openfire Enterprise. The last step is to enable clustering and see how the machines meet each other. Follow these steps to enable clustering:

1. Go to the clustering page in the admin console (**Server | Server Manager | Clustering**).
2. Select the Enabled option button to enable clustering on the machine and click the **Save Settings** button. This will activate the cluster. There might be a brief delay when you enable clustering on the first machine.
3. Once the cluster is enabled on one machine, repeat the same step for the other machines on which you've installed Openfire.

Your Openfire cluster is now up and running! Refer to the screenshot below:

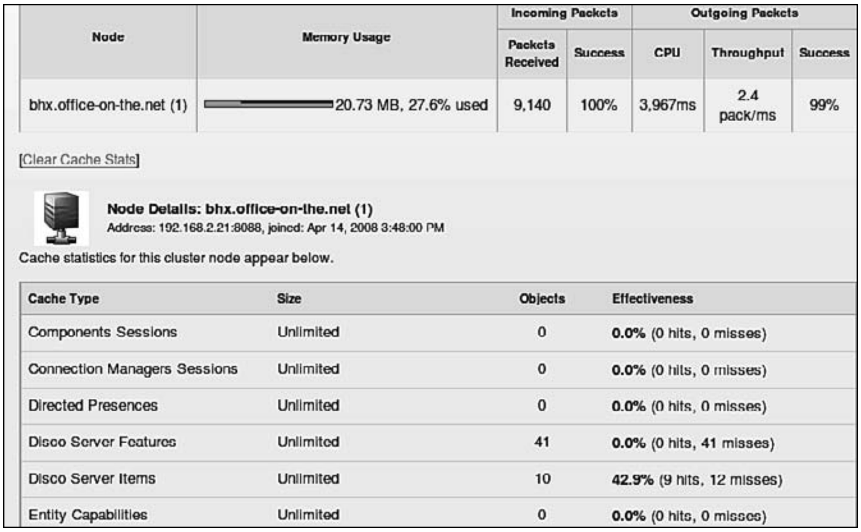


Cluster Overview

Below is an overview of your cluster. You have 2 node(s) running and you are licensed to 10 node(s) in this cluster. T
information, click each node. The row in yellow indicates the local node.

Nodes	Joined	Clients	Incoming Servers	Outgoing Servers	Memory
192.168.2.6	Mar 11, 2008 8:30:17 PM	0 (0%)	0 (0%)	0 (0%)	9.14 MB of 63.31 MB used
192.168.2.3	Mar 11, 2008 7:57:37 PM	0 (0%)	0 (0%)	0 (0%)	14.21 MB of 63.31 MB used

Once you've enabled the cluster, the Clustering page expands to give you an overview of the available nodes in the cluster. You can glance at all of the nodes in the cluster: when each one joined in, the number of users each is serving, how much resource (memory) each is consuming, and so on.



Node	Memory Usage	Incoming Packets		Outgoing Packets		
		Packets Received	Success	CPU	Throughput	Success
bhx.office-on-the.net (1)	20.73 MB, 27.6% used	9,140	100%	3,967ms	2.4 pack/ms	99%

[Clear Cache Stats]

Node Details: bhx.office-on-the.net (1)
Address: 192.168.2.21:8088, joined: Apr 14, 2008 3:48:00 PM

Cache statistics for this cluster node appear below.

Cache Type	Size	Objects	Effectiveness
Components Sessions	Unlimited	0	**0.0%** (0 hits, 0 misses)
Connection Managers Sessions	Unlimited	0	**0.0%** (0 hits, 0 misses)
Directed Presences	Unlimited	0	**0.0%** (0 hits, 0 misses)
Disco Server Features	Unlimited	41	**0.0%** (0 hits, 41 misses)
Disco Server Items	Unlimited	10	**42.9%** (9 hits, 12 misses)
Entity Capabilities	Unlimited	0	**0.0%** (0 hits, 0 misses)

You can also get more detailed information about each node by clicking on its name in the summary table (refer to the previous screenshot). This takes you to a detailed Cluster Node Information page which is constantly refreshed to give you the latest information for the selected node. From this interface, you can keep track of the packets flowing into a node and the time taken to process the outgoing packets, with the success rate at both ends. To further help you keep tabs on a remote cluster node, you are also presented with the statistics for various Openfire caches on every node, including the File Transfer Cache, the Gateway Registration Cache, the Last Activity Cache, the Offline Message Size, and so on.

Of Caches and Clustering

Wondering how Openfire seamlessly handles and manages all user data and other data caches, despite the fact that it's now practically running in a distributed environment as a cluster instead of a single Openfire server? When the Openfire developers designed the clustering part of the server, they separated Openfire caches into two kinds—distributed and replicated.

The replicated caches are configured to be replicated over the cluster, out-of-the-box. The contents of such a cache are distributed across nodes because they are accessed frequently. Having them available locally reduces access time. The other type of cache is a distributed cache which is cordoned off in the node. These are the caches that can grow in size, and replicating them over the cluster would defeat our scalability purpose.

On the topic of clusters, now that you have one running, you might have noticed it isn't quite like the common HTTP clustering that you're used to. In traditional HTTP clustering, connections sent to a failed domain go to the next available node in the cluster. This happens transparently without the user being aware of the fact that his request is being serviced by another host. In the Openfire world, which is ruled by the XMPP protocol, unlike HTTP the connection to the server is a stateful connection. So when the server goes down, the client has to open a new socket connection and connect to a new node in the cluster.

So if you have a cluster running as we do now, if one node goes down, users connected to that node will be disconnected. They'll have to manually reconnect to another node on the cluster to resume their IM service. Meanwhile, you can diagnose and resurrect the node that has gone down. This is far better than just relying on one server, which, if it goes down, knocks the entire service offline.

Tips for Optimizing the Cluster

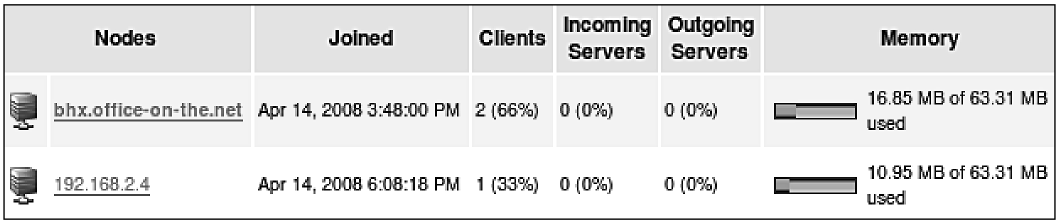
While a cluster with multiple points of failure works wonders if you have a large number of users relying on the service, you could further enhance the setup by automatically balancing the load between nodes (refer to the following screenshot).

What's even better is that you can balance loads between the cluster nodes, without having to spend too much time doing something fancy, and without an expensive hardware load balancer. If you run your own DNS server, which most organizations these days do, you could rely on your DNS server to balance loads between your nodes using the Round Robin technique. This technique works by responding to a DNS query with a bunch of IP addresses (one for each cluster node) instead of just the one for the Openfire server.

But while a DNS Round Robin is easy to setup, there are two rather limiting disadvantages. This technique doesn't involve calculating the load on a node before directing traffic to it. Also, if a node fails, its address will still be in the list of addresses returned by the DNS server.

Managing a cluster of nodes is a major task in itself. Depending on your cluster, you should prepare a Clustering Best Practices guide for your network. Many people suggests that at the minimum, you should consider lowering the Time-to-Live (TTL) value, so that when you take out failed nodes from the cluster, requests are no longer sent to the unavailable node.

If you use a client that can transparently reconnect to the server (as is the case with Spark), upon disconnection, your users will automatically be reconnected (possibly after a few attempts) to a new node in the cluster.



Nodes	Joined	Clients	Incoming Servers	Outgoing Servers	Memory
bhx.office-on-the.net	Apr 14, 2008 3:48:00 PM	2 (66%)	0 (0%)	0 (0%)	16.85 MB of 63.31 MB used
192.168.2.4	Apr 14, 2008 6:08:18 PM	1 (33%)	0 (0%)	0 (0%)	10.95 MB of 63.31 MB used

Balancing loads is critical in some Openfire deployments because of the sheer number of users employing the service. DNS Round Robin may do a good job of balancing loads evenly, but it doesn't take into account the load statistics. Depending on your needs, you can also use a more sophisticated load balancer such as the Linux Virtual Server (see `http://www.linuxvirtualserver.org/`) or even a hardware load balancer if you have a very large number of users simultaneously IMing via Openfire.

Scaling with Connection Managers

In addition to running a cluster of Openfire servers, you can also scale your Openfire setup by using Connection Managers. Connection Managers were first developed back in the days when Openfire couldn't handle more than 5,000 concurrent users. With the help of a Connection Manager, you could scale an Openfire server to handle an additional 5,000 users.

With the development of the clustering module, most enterprise users would prefer to run a cluster of Openfire servers. But the Connection Manager module, which is constantly developed with every Openfire release, is a great way to scale and distribute load among several Connection Managers.

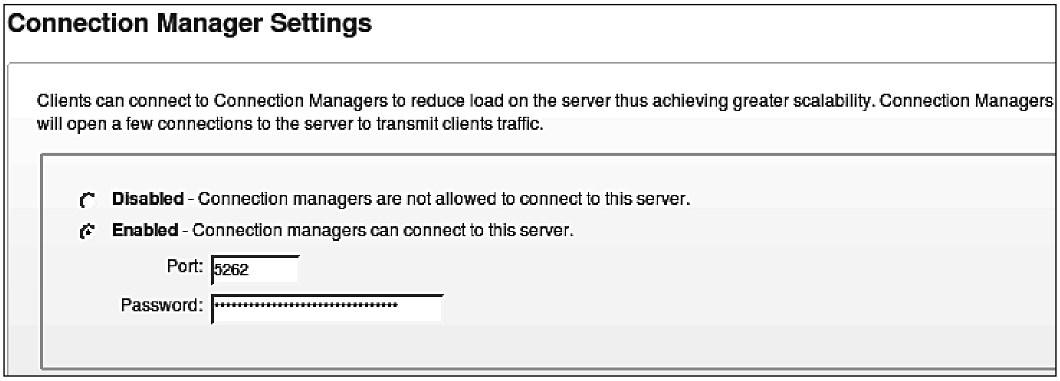
Deploying Connection Managers

Connection Manager can be deployed on both Linux and Windows platforms. Each Connection Manager is capable of handling 5,000 users, and you can deploy as many Connection Managers as you want. If you have users spread over multiple floors or across large rooms, you may want to deploy a Connection Manager in every room or on every floor. You could also assign users to different Connection Managers using the DNS Round Robin technique we've discussed above.

Setting up a Connection Manager involves little more than downloading and activating a plugin. The Connection Manager module is distributed as an archive for both platforms. However, as it doesn't bundle a Java Runtime Environment (JRE), make sure you have one installed on your machine. JREs for Windows and Linux are both available as a free download from Sun's Java website (see `http://www.java.com/en/download/manual.jsp`).

Step 1: Configure the Openfire Server

To setup a Connection Manager, you'll first have to configure your Openfire server to let it delegate connections to a Connection Manager. To do this, log into the administration console and head over to the **Connection Manager Settings** page (**Server** | **Server Settings** | **Connection Managers**).


Connection Manager Settings

Clients can connect to Connection Managers to reduce load on the server thus achieving greater scalability. Connection Managers will open a few connections to the server to transmit clients traffic.

Disabled - Connection managers are not allowed to connect to this server. [✓ **Enabled** - Connection managers can connect to this server.]

Port: 5262

Password: ********************************

By default, an Openfire server doesn't allow Connection Managers to connect to it. Selecting the **Enabled** option button (refer to the previous screenshot) should do the trick, though. When this option is enabled, make sure you note down the two important values:

- **Port**: The port number over which Connection Managers communicate with the Openfire server. The default port number is **5262**.
- **Password:** All Connection Managers must authenticate themselves to the server by using a password.

After making the appropriate changes, click the **Save Settings** button to apply the settings. Our Openfire server is now prepared for receiving connections from Connection Managers. All that remains is to set up the Connection Manager and bringing it online.

Step 2: Configure the Connection Manager

Once you have the JRE setup, download and extract the contents of the Connection Manager module for your preferred platform. Put the extracted `connection_manager` folder in an appropriate location, such as under `Program Files` in Windows and `/opt` under Linux. That's about all it takes to install a Connection Manager.

Next you'll have to edit the configuration file for the Connection Manager to specify the settings (port and password) that it'll need to connect to the Openfire server. Open the `connection_manager/conf/manager.xml` file in your favorite text editor and locate the following properties in the `<xmpp>` section:

- <domain>: This is the domain name of the Openfire server. The Connection Manager will forward all connections to this domain.
- <port>: This is the port on which the Connection Manager connects with the Openfire server. This defaults to 5262.
- <password>**:** This is the password your Openfire server expects from all Connection Managers.

Step 3: Run The Connection Manager

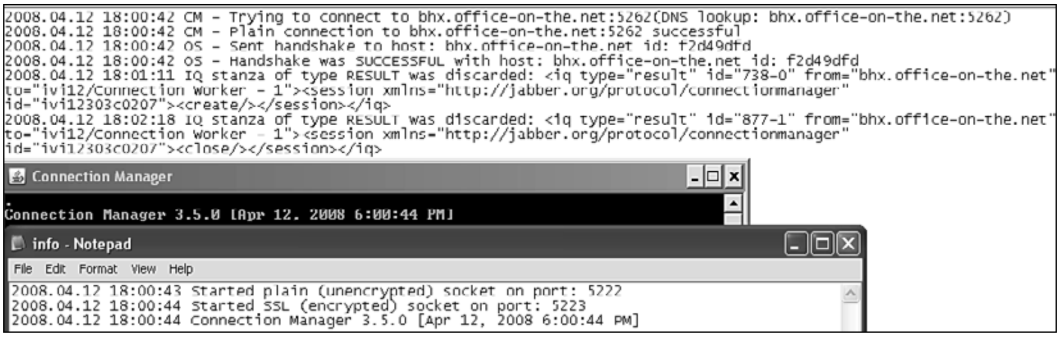
Once the Connection Manager is configured, you can start it using one of the following commands, depending upon the platform.

Windows users:

```
cd <path-to-connection_manager>/bin
cmanager.bat
```

Linux users:

```
cd <path-to-connection_manager>/bin
chmod u+x cmanager.sh
./cmanager.sh
```

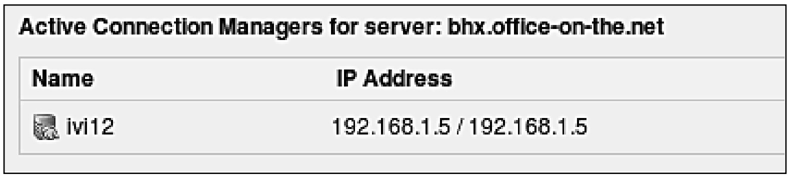


```
2008.04.12 18:00:42 CM - Trying to connect to bhx.office-on-the.net:5262(DNS lookup: bhx.office-on-the.net:5262)
2008.04.12 18:00:42 CM - Plain connection to bhx.office-on-the.net:5262 successful
2008.04.12 18:00:42 OS - Sent handshake to host: bhx.office-on-the.net id: f2d49dfd
2008.04.12 18:00:42 OS - Handshake was SUCCESSFUL with host: bhx.office-on-the.net id: f2d49dfd
2008.04.12 18:01:11 IQ stanza of type RESULT was discarded: <iq type="result" id="738-0" from="bhx.office-on-the.net"
to="ivi12/Connection Worker - 1"><session xmlns="http://jabber.org/protocol/connectionmanager"
id="ivi12303c0207"><create/></session></iq>
2008.04.12 18:02:18 IQ stanza of type RESULT was discarded: <iq type="result" id="877-1" from="bhx.office-on-the.net"
to="ivi12/Connection Worker - 1"><session xmlns="http://jabber.org/protocol/connectionmanager"
id="ivi12303c0207"><close/></session></iq>
```

```
.
Connection Manager 3.5.0 [Apr 12. 2008 6:00:44 PM]
```

```
2008.04.12 18:00:43 Started plain (unencrypted) socket on port: 5222
2008.04.12 18:00:44 Started SSL (encrypted) socket on port: 5223
2008.04.12 18:00:44 Connection Manager 3.5.0 [Apr 12, 2008 6:00:44 PM]
```

If the Connection Manager fails to start, make sure that you have JRE installed (see `http://www.igniterealtime.org/community/message/127912`) and that your JAVA_HOME environment variable path is in order (see `http://www.igniterealtime.org/builds/openfire/docs/3.5.0_rc1/documentation/source-build.html#setup`).


Active Connection Managers for server: bhx.office-on-the.net

Name	IP Address
ivi12	192.168.1.5 / 192.168.1.5

When the Connection Manager is up and running and communicating with the Openfire server (refer to the previous screenshot), it'll be listed on the Connection Manager Settings page, along with its IP address and the number of clients connected to it. Users who want to connect through the Connection Manager will have to specify the IP address of the Connection Manager in the server field when logging on to Openfire.

Summary

Openfire is very capable of handling several thousands of users. In this chapter, the rubber hit the road. We've put to test Openfire's scalability claims, and in doing so have introduced redundancy into our setup to make sure that a server failing doesn't affect the service.

There are two ways in which a server can be configured to distribute loads—by forming a cluster of Openfire nodes and by delegating Connection Managers to make connections to clients. The clustering feature is bundled with Openfire Enterprise edition, while the Connection Manager module can be downloaded for free. Because clustering in HTTP is different from XMPP clustering, we've also discussed how one can use simple load balancers and clients that reconnect automatically to provide an uninterrupted IM'ing experience.

Not many people run Connection Managers, since they were primarily introduced to counter a now-extinct Openfire limitation. But they are still a viable cost-effective option to scale Openfire and introduce redundancy. This chapter also covered installing and deploying Connection Managers in both Windows and Linux environments.

10 Communication Across Multiple Locations

Is your organization spread across multiple offices in geographically dispersed locations? Does a particular team have members in multiple offices? While the last chapter helps you balance load in a single location, in this chapter, we'll look at the other major communication bottleneck—connecting users across multiple locations. The complexity here of course is that users in different locations are connected to their own regional intranets. But if you run Openfire servers on your various intranets, you can easily configure them to talk to each other.

This seemingly difficult task is managed by the Openfire feature known as server-to-server communication. True to its name, the server-to-server communication feature helps establish communication between two different Openfire servers. It doesn't matter if the servers are in different buildings or in different continents, as long as they can be reached via either the Internet or a dedicated communication channel such as a Virtual Private Network (VPN).

With Openfire, the more complex the task, the easier it is to set up, and server-to-server communication is no different. But there's more to setting up communication between two remote servers than just clicking a button or two in the administration console. In a snap, all you need are a couple of Openfire servers with names that can be resolved by the DNS. When the servers are up, users from either server can add users on the other to their roster as if they were on the same network.

This set up also helps to eliminate the problem of the *road warrior*. Road warrior is a fancy name given to a user who moves between networks. With server-to-server communication, our road warrior would be able to log on to the internal Instant Messages (IM) server from another network or from anywhere else in the world.

In this chapter, we will look at:

- Linking users on two Openfire servers
- Adding users from the remote network
- Monitoring remote connections
- Enabling secure communication

Linking Two Branches

To better understand this type of remote interaction, let's assume we have two offices, one based in Mumbai, and the other based in Birmingham. Both offices run Openfire servers. Now, it would be a great idea to connect users from both of the offices with each other seamlessly and securely:

- From their own office
- When they are in the other office
- From anywhere else in the world

This set up isn't very difficult to implement. But you need to make sure the Openfire servers have a DNS-resolvable name. Openfire servers make a DNS lookup to work out the actual IP address and port to be used to connect to the remote server. If you don't have one, you can use one of the free DNS services on the Web such as DynDNS (see `http://www.dyndns.com/`).

Why Use DynDNS

If you are a multinational corporation, you would probably have more than a handful of DNS names. But if you are a nonprofit or charitable organization and don't have much load, you can make your own Openfire server publically accessible using DynDNS over your broadband connection.

In very broad terms, a DNS is like a database that converts a computer's fully qualified domain name (FQDN) into an IP address. We need the DNS to translate the difficult-to-recall IP address (such as 61.123.154.97) into a human-friendly name (such as `chat.wethepeople.org`).

Now the complexity with most consumer Internet connections (broadband or otherwise) is that the ISPs assign different IPs to a user every time he or she logs in. For example, while 61.123.154.97 points to `chat.wethepeople.org` one day, the next day it could point to `meanbadmonopoly.com`. Two different hosts cannot have the same IP address, so when users try to get to `chat.wethepeople.org`, they get a **Host not found** error because your DNS record doesn't have an IP address listed. On rare occasions, your users could also be redirected to `meanbadmonopoly.com`, that's where your last known IP address points to currently.

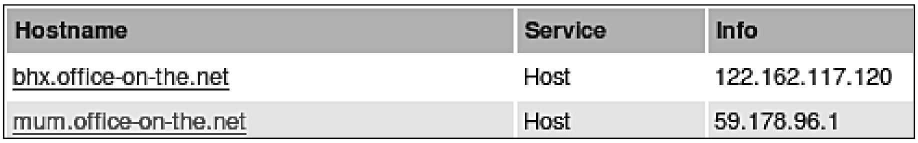
This is where Dynamic DNS services such as DynDNS come into the picture. They keep track of your dynamically changing IP address to make sure your DNS record is kept updated with your changing IP address. This allows your users to always reach you despite your dynamic IP address. Getting a DynDNS account isn't a complex task. Just follow the simple instructions on their website and set up your free account. Make sure to get as many accounts as you have branches.

Setting up Server-to-Server Communications

Using DynDNS, I've set up the following two accounts based on the example given in this chapter:

- **bhx.office-on-the.net**—It points to the Openfire server in Birmingham, UK
- **mum.office-on-the.net**—It points to the Openfire server in Mumbai, India

You can choose different names and if you run your own DNS, your names will be different.



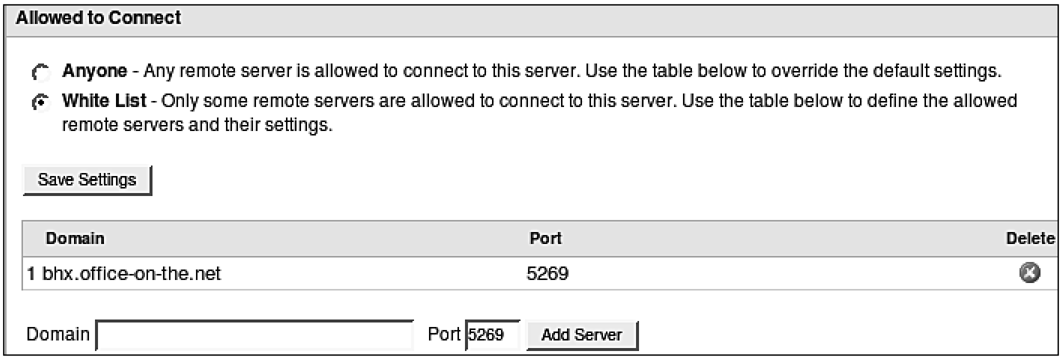
Hostname	Service	Info
bhx.office-on-the.net	Host	122.162.117.120
mum.office-on-the.net	Host	59.178.96.1

Once you have set up the hostnames, make sure to use them as "server name" when setting up Openfire at both the locations. Once Openfire is set up, head over to the Openfire administration console. The server-to-server settings can be accessed from **Server** | **Server Settings** | **Server to Server**. From this page, you can do the following:

1. Enable/Disable the service and choose a port number: All server-to-server communication flows through a particular port. By default, this is port number 5269, but you can choose any other free port.
2. Disconnect on idle: Server-to-server connections are established on an as-needed basis, and you can choose to close idle connections. The disconnect time interval is specified in minutes and the default is 10 minutes.

3. Restrict access: You can choose to allow users from all remote servers to connect, or you can restrict access to particular trusted servers via a white list, which is advisable.
4. Optionally, instead of using a white list to allow particular servers to connect, you can have a black list which lists servers that you don't want to connect to (while allowing all others to connect).

Using the example we are following in this chapter, we'll "Enable the service" to communicate through the default port 5269 on both the servers. Furthermore, we'll also restrict communication and allow users only from these two servers by adding the details of the remote server in the white list of each server. So in the Openfire at Birmingham, UK, we'll add the Mumbai server in the white list and vice-versa (refer to the following screenshot).


Allowed to Connect

Anyone - Any remote server is allowed to connect to this server. Use the table below to override the default settings. [✓ White List - Only some remote servers are allowed to connect to this server. Use the table below to define the allowed remote servers and their settings.]

Domain	Port	Delete
1 bhx.office-on-the.net	5269	⊗

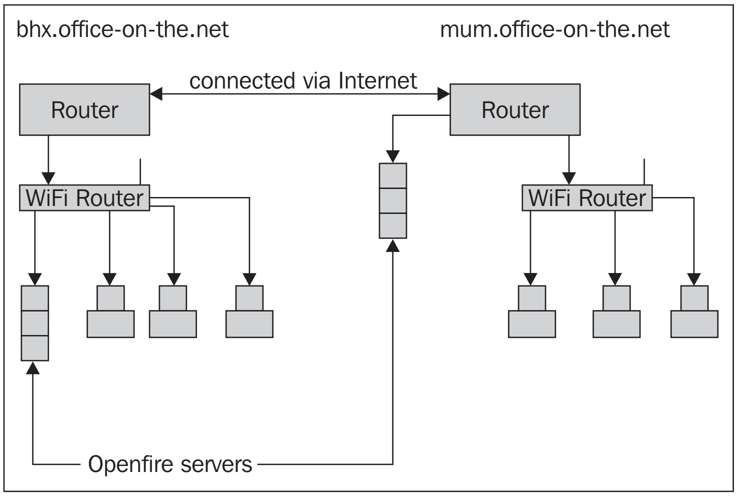
Domain: (empty) Port: 5269

Now, we're all set. Theoretically, users from one server will now be able to search and add users from the other server. But as it is with communication across networks, there are various complexities involved before communication can take place. Let's examine them in detail.

About Remote Networks, Firewalls, and Port Forwarding

For server-to-server communication to succeed, there are a couple of things that must happen. First of all, you will have to make sure that all of your Openfire servers (both local and remote) have names that can be resolved by the DNS. We have already taken care of this in the preceding section.

The second thing we have to take care of is making sure that there is no firewall blocking the port over which server-to-server communication will take place (port 5269 by default). This is more complex that just turning off or adding appropriate rules in the firewall on the machine running the Openfire server.



The diagram above is a graphical representation of two typical Openfire networks. On the left, we have a network that is connected to the Internet via a DSL modem. The DSL modem in turn is connected to a wireless router. The Openfire server and other users on the network are all connected to the wireless router. In such a network, you'll have to make sure that the DSL router forwards traffic bound for the Openfire server via the 5269 port to the wireless router, which in turn forwards it to the Openfire server.

Similarly, the network on the right is also connected to the Internet via a DSL router. That DSL router is, in turn, connected to a wired router, which is connected to the Openfire server. The users on this network are connected to the wireless router, which is connected with the wired router, which in turn serves the Openfire server.

Both of these networks can find each other since they have DNS resolvable names. But to get users on both of these networks to talk to each other, you'll have to make sure that the traffic is forwarded to the appropriate computer running the Openfire server on the correct port.

Checking Connectivity on the Port

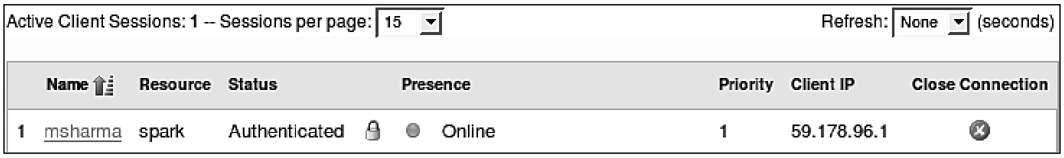
So you have edited your firewall and have made sure that it passed traffic through the relevant port. But how do you test whether you are able to connect to the remote Openfire server? For that, you'll need a telnet client. All mainstream Linux distributions are equipped with a telnet client. There are several free telnet clients for Windows users as well, such as PuTTY (see `http://www.chiark.greenend.org.uk/~sgtatham/putty/`) and dtelnet (see `http://dtelnet.sourceforge.net`).

Once you have the telnet client installed, use it to connect to the remote host over the server-to-server port. That's it! Congratulations! You have now successfully established communication between the servers. You're now ready to go.

Adding Users from Remote Network

Connections to and from remote servers are established when users from both networks want to communicate with each other. If you have successfully been able to telnet over port 5269 to the remote server, it's time to add users from the remote network.

There's nothing special about adding users from a remote server. You can use the **Search** plugin to look for users and then add them in the same way as you'd add local users. Because under XMPP, the server name is part of the Jabber ID of the user, the server knows where to look for local users. This also avoids the problem of having two users with the same username on different servers. In addition to adding them, for all other activities involving users, such as logging conversations, group chat, and so on, the remote users will be treated as if they were local users on the local server.

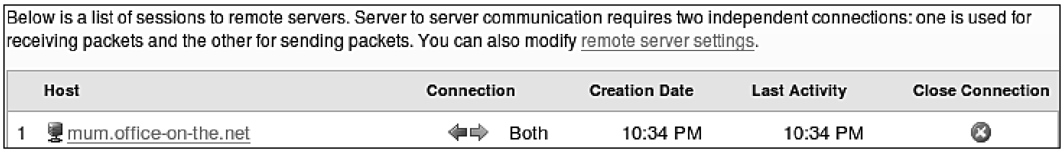

Active Client Sessions: 1 -- Sessions per page: 15 Refresh: None (seconds)

	Name	Resource	Status	Presence	Priority	Client IP	Close Connection
1	msharma	spark	Authenticated	Online	1	59.178.96.1	

You'll also be able to administer remote users as if they were your own. They'll be listed under the **Client Sessions** section accessible from **Sessions** | **Active Sessions** | **Client Sessions**, from where you have the power to terminate their session. Furthermore, you'll also be able to add remote users to predefined rosters and groups. This would be really helpful if members of a team are in remote offices.

Monitoring Remote Connections

A server-to-server connection is established only when a user from a remote server starts a conversation with a user on the local server. Actually, it doesn't matter who initiates the conversation; a server-to-server connection is only established when two users who are not on the same server start communicating with each other.

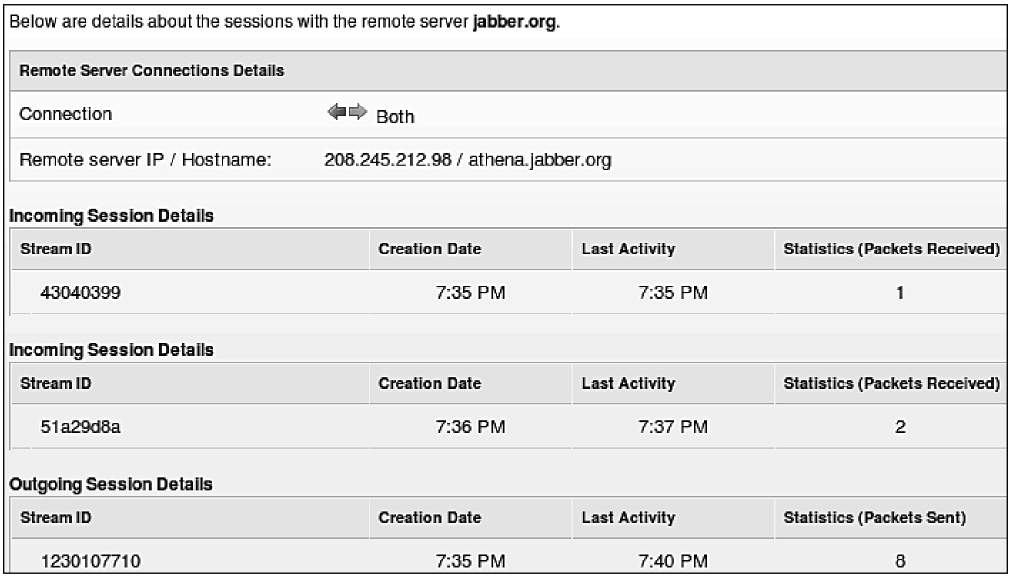

Below is a list of sessions to remote servers. Server to server communication requires two independent connections: one is used for receiving packets and the other for sending packets. You can also modify remote server settings.

	Host	Connection	Creation Date	Last Activity	Close Connection
1	mum.office-on-the.net	Both	10:34 PM	10:34 PM	

When a server-to-server session has been established, it's listed under the server session view accessible from **Sessions | Server Sessions**. From here, you will get the following details for each session:

- The name of the remote server
- The type of connection – One-way or two-way
- The date/time when the connection was created
- The date/time of the most recent activity through the remote connection

You can also click on the name of the server to get more details about the session with that particular remote server.



Below are details about the sessions with the remote server **jabber.org**.

Remote Server Connections Details	
Connection	⇦⇨ Both
Remote server IP / Hostname:	208.245.212.98 / athena.jabber.org

Incoming Session Details

Stream ID	Creation Date	Last Activity	Statistics (Packets Received)
43040399	7:35 PM	7:35 PM	1

Incoming Session Details

Stream ID	Creation Date	Last Activity	Statistics (Packets Received)
51a29d8a	7:36 PM	7:37 PM	2

Outgoing Session Details

Stream ID	Creation Date	Last Activity	Statistics (Packets Sent)
1230107710	7:35 PM	7:40 PM	8

From the **Remote Server Connections Details** page, you can find the IP address and hostname of the remote server as well as get details about the incoming and outgoing sessions.

Establishing Secure Server-to-Server Communication

There's one drawback to server-to-server communication. Unless you are using a secure channel such as a VPN, communication between the networks flies across an unsecured channel—the Internet. Because you can't really secure the Internet, the only other way to safeguard against information leaks is to secure the communication itself.

This can be done by establishing a secure communication channel between the local and the remote servers. To do so, head over to the **Server | Server Settings | Security Settings** section. From here, you can define both how clients connect to the server and how servers connect to each other. Because a client-to-server communication happens within the confines of a network, you can leave **Client Connection Security** as **Optional**, which means clients can connect to the server using secured connections.

But because servers may connect with each other over the unsecured Internet, it's advisable to switch **Server Connection Security** to **Required**, which forces connections between servers to always establish secured connections. If you are using a self-signed certificate, you'll have to add a system property to stop the server from verifying the authenticity of the certificate. Head over to **Server | Server Manager | System Properties**, scroll down to the **Add new property** section, enter **xmpp.server.certificate.verify** as the name of the property, and set its value to **False**. You now have a working encrypted server-to-server session!.

> Creating secure connections via SSL certificates is a fairly simple but involved process. The Openfire developers have written an SSL Guide (see `http://www.igniterealtime.org/builds/openfire/docs/latest/documentation/ssl-guide.html`) to help you add SSL certificates to Openfire. Also check out the Certificate Survival Kit (see `http://www.igniterealtime.org/community/blogs/ignite/2007/11/30/your-certificate-survival-kit`) blog post by Openfire's lead-developer.

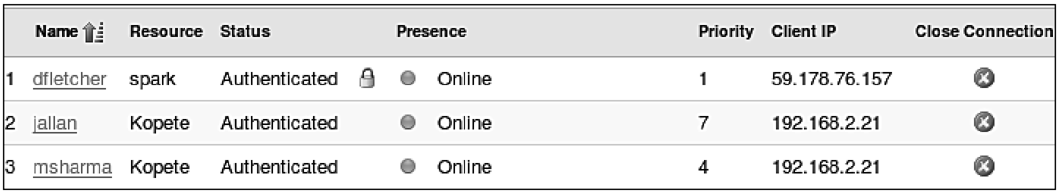
When users from either servers are connected, the server-to-server session is shown with a padlock in the server session view.

The Road Warrior

So what happens if a user from one office has gone to the other office for a brief visit? Do you need to tweak his or her server settings? Does the administrator of the remote server create a temporary user for him or her? As strange as it may sound, our road warrior needs to do nothing special when on the road. Yes, that's true. Absolutely nothing!

> As a general policy, it's a good idea to use secure client connections when connecting from outside the network. Also, double check those firewall ports if you run into trouble.

Suppose a user from the Birmingham office is in the Mumbai office. In this case, he or she will still be able to connect to his or her server in Birmingham over the Internet because the server has a DNS-resolvable name. Once the user is connected to the server, he or she will be able to access his or her roster and chat with users on the network he or she is roaming in, that is, users in the Mumbai office.



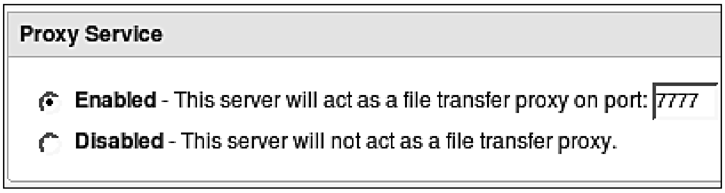
	Name	Resource	Status	Presence	Priority	Client IP	Close Connection
1	dfletcher	spark	Authenticated	● Online	1	59.178.76.157	✕
2	jallan	Kopete	Authenticated	● Online	7	192.168.2.21	✕
3	msharma	Kopete	Authenticated	● Online	4	192.168.2.21	✕

But here's the real killer. Because both servers in our example use DNS-resolvable names, users from either server can log into the server from anywhere on the planet. In the preceding screenshot, users **jallan** and **msharma** are on the local network (notice their 192.168.x.x series IP) while user **dfletcher** is on a beach in Florida!

File Transfer Across Servers

After you've configured your network to allow server-to-server communication, there's one little bit of tweaking that you'll have to do if you want users to be able to share files across the network. File transfers between users not on the same network have to go through a file transfer proxy.

Like all things in Openfire, setting this up involves a couple of clicks. Head over to the **Server | Server Settings | File Transfer Settings** section. To bring the file transfer proxy to life, simply select the **Enabled** option button. You can also optionally change the port that acts as the proxy, which defaults to 7777.



Proxy Service

[✓ **Enabled** - This server will act as a file transfer proxy on port: 7777]

Disabled - This server will not act as a file transfer proxy.

That takes care of the Openfire bit of the configuration. Again, as with the server-to-server settings, you'll have to make sure that port 7777 is not blocked by a firewall, and then add appropriate traffic-forwarding rules to the appropriate network devices leading to and from the Openfire server.

[Once the Openfire server is set up, you'll have to configure your respective clients to make them use the proxy for file transfers.]

Summary

Server-to-server communication is a very powerful feature of Openfire. You can certainly use it to take the load off one server, but you'll truly appreciate it when you have people spread over multiple locations, all wanting to collaborate.

In this chapter, we've looked at how users can connect and interact with users on remote Openfire servers. The real beauty of this form of communication is that users don't have to learn new tricks to add or communicate with offsite users. Due to the way server-to-server communication is designed and handled, users who move between multiple Openfire servers can stay connected with each other.

11
Running an Online Helpdesk

Openfire is a fantastic instant messaging server, as you'll probably agree, having read through the chapters. But there's another trick it can do which is pretty cool if you need to interact with people outside your organization—an online helpdesk.

An online helpdesk is a web-based interface to your internal instant messaging server which lets users outside the network connect and interact with members of the organization. Traditionally, you'd need such a service if you provide product or software-based service, but as Openfire is free, it is accessible to a wide spectrum of companies and organizations who can use it as an online receptionist. Once you're through with this chapter you'll see that Openfire fits snuggly on your website, next to your contact details.

In this chapter, we will learn how to:

- Create and design an online helpdesk
- Add users as agents
- Route online calls to agents
- Set up canned responses
- Monitor sessions

Installing Fastpath and Webchat

For deploying an online help desk, you'll need a couple of plugins. These plugins started out as commercial plugins but were released as Open Source ones just before I started working on this chapter. Because the transition is still underway, the plugins might still be listed as an Enterprise plugin, depending on the Openfire version you are using, but you'll be able to install it nonetheless.

The first plugin you need for an online helpdesk is called Fastpath. This plugin does all the work in the background, such as managing queues of callers as they call in. This plugin is also responsible for routing online chat requests to users inside the Openfire network. You can install the plugin from the **Plugins** tab under Openfire's web-based admin console. Or, you can download the plugin from the Openfire website (see `http://www.igniterealtime.org/projects/openfire/plugins.jsp`) and upload it from the plugins interface in the admin console, by browsing to the downloaded plugin (`fastpath.jar`) and hitting the **Upload Plugin** button.

> The process for deploying the plugin is a little involved if you are upgrading the plugin from its pre-Open Source days. The steps are detailed in the plugins readme file (see `http://www.igniterealtime.org/projects/openfire/plugins/fastpath/readme.htm`).

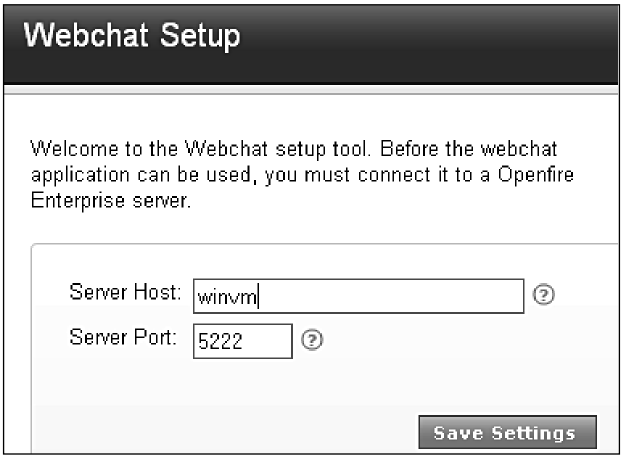
The other plugin you need is the Spark Fastpath Webchat plugin. Webchat is responsible for the interface that your users/visitors to the website will use to exchange messages with people inside the network. You'll need to install this plugin on your web server. It works with any Application Server with Java Servlet 2.3 or greater, such as Apache Tomcat (see `http://tomcat.apache.org/`), Caucho's Resin (see `http://www.caucho.com/`), RedHat's JBoss (see `http://www.jboss.org/`), BEA's WebLogic (see `http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/weblogic/server`), and so on. You'll need to grab the plugin that is distributed as a .WAR java archive package (see `http://www.igniterealtime.org/projects/openfire/plugins/webchat.war`) and copy it to your application server's `webapps/` directory.

You can also install the plugin from within the Openfire interface for evaluation purposes. This installation of the Webchat client is powered by the Jetty WebServer (see `http://www.mortbay.org/jetty-6/`). To keep things synchronized, irrespective of your setup, let's assume this is what you've done. The steps vary minutely in their execution depending on which application server you are running.

After the plugin is installed, you'll have to restart your Openfire server and then head over to `http://<hostname>:<port_number>/webchat` where hostname is your application server and port number is the port used for admin access. This launches a simple configuration form where you have to enter a couple of details:

- **Server Host**: This is the name or IP of the server running the Openfire Instant Messenger (IM).
- **Server Port**: This is the port over which the server listens for IM communications. By default, Openfire listens on port 5222. If you haven't changed that, you should leave this box as it is.

And that's it.



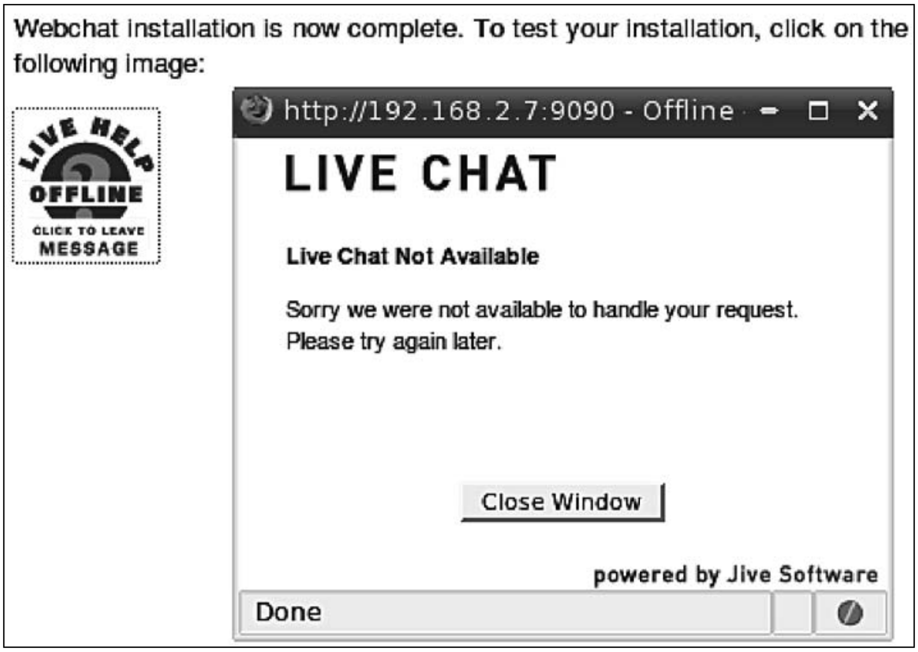
Webchat Setup

Welcome to the Webchat setup tool. Before the webchat application can be used, you must connect it to a Openfire Enterprise server.

Server Host: winvm

Server Port: 5222

Using these settings, the plugin will try contacting the Openfire server. When it's successful, it will display an icon and ask you to click on it to test your installation. As we haven't configured any users / support agents, you'll get a **Live Chat Not Available** message (refer to the following screenshot).



Webchat installation is now complete. To test your installation, click on the following image:

LIVE CHAT

Live Chat Not Available

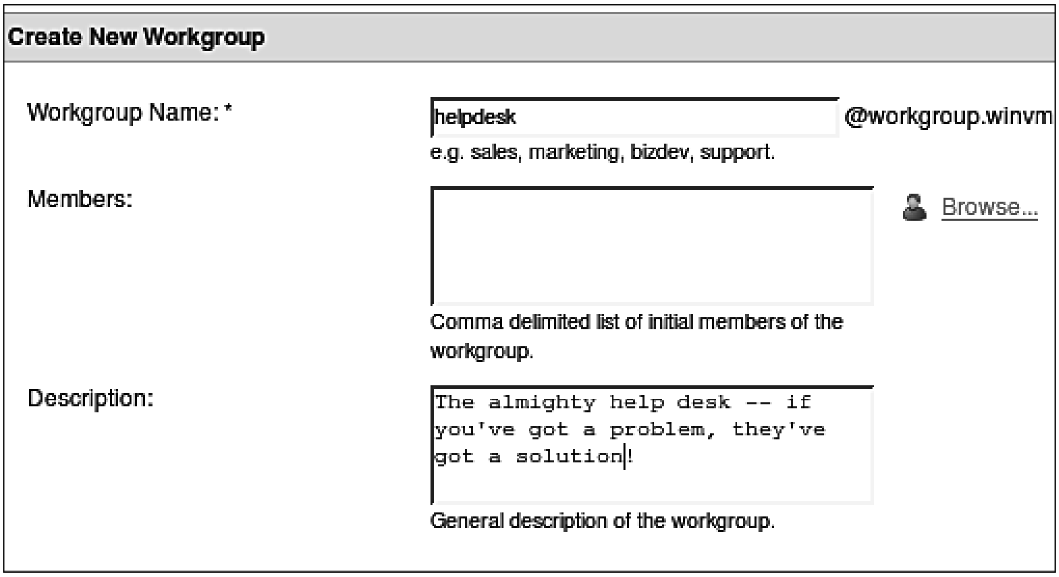
Sorry we were not available to handle your request. Please try again later.

Now that our plugins are all set, let's head on to Openfire and set up our helpdesk.

Creating a Helpdesk

In Openfire terminology, a helpdesk is referred to as a workgroup. Each workgroup has several members who are Openfire users inside the network. You can have multiple workgroups, and workgroups can share members. When a visitor on your website wants to talk to someone on the inside, he or she will have to select the **chat** icon for the corresponding workgroup. This request is then intercepted by the Fastpath plugin which looks up the settings for that particular workgroup and acts accordingly. In this section, we'll create a **helpdesk** workgroup and go over its settings.

Once the Fastpath plugin is installed, you'll notice an additional **Fastpath** tab in the top navigation bar in the Openfire administration console. When you click on this tab, it takes you to the Fastpath **Workgroup Manager** from where you can add, edit, and delete workgroups, as well as configure them.


Create New Workgroup

Workgroup Name: * helpdesk @workgroup.winvm
e.g. sales, marketing, bizdev, support.

Members: Browse...
Comma delimited list of initial members of the workgroup.

Description: The almighty help desk -- if you've got a problem, they've got a solution!
General description of the workgroup.

By default, the plugin already lists a **demo** workgroup. To add a workgroup, click on **Create Workgroup** (**Fastpath** | **Workgroups Manager** | **Create Workgroup**) and fill in the following details:

- **Workgroup Name**: The name of the workgroup (in our case **helpdesk**)
- **Description**: A brief synopsis of the workgroup
- **Members**: A comma-separated list of members of the workgroup

Except for the name of the workgroup, the other two details are not required for creating a workgroup. You could skip adding a description, although it's not a good idea to do so. However, adding members later would be a good idea, as we'll see soon.

Global Settings

Once you've created a group, head over to the **Manage Settings** section (**Fastpath | Workgroups Manager | Manage Settings**). This displays a form that you can use to set some global parameters which are applied to all groups. On offer are:

- **Maximum chat sessions per agent**: The maximum number of sessions a particular agent can handle. Once the number is reached, Openfire will not route any more calls to that particular user. By default, this is set to **4**.
- **Minimum chat sessions per agent**: The least number of chat sessions an agent can handle. By default, this is set to **1**.
- **Request timeout**: The amount of time (specified in seconds) agents have to pick up a request from the user. The call is timed out if no agents picks it up with in the time specified.
- **Agent timeout to accept an offer**: The amount of time (in seconds) each agent has to accept a request before it's passed on to another agent. This period cannot be greater than the request timeout.
- **Expire agent rejection**: The amount of time after which a rejected request can be requested again.
- **Times to overflow before canceling request**: The number of times a request can be routed to other queues (we'll talk about queues in the next section) before the request is cancelled.
- **Agents are allowed to change their names**: If selected, this option will let the agents/users on the Openfire server change their identity when interacting with external users.

None of these options can be left unspecified, and all of them have preset values. As these settings are applied to all workgroups globally, you might want to review them. Instead, leave them as they are, and tailor the settings of the individual groups to your liking.

Workgroup Settings

Workgroup settings can be customized for individual workgroups. To change a workgroup's settings, click on a **workgroup** in the **View Workgroups** section, and scroll down to the **Settings** link at the bottom. These settings are based on the global settings and offer only those configuration options that affect a group the most. For a particular workgroup, you can:

- Specify the maximum and minimum number of sessions per agent
- Set a request timeout
- Set an offer timeout which is the amount of time an agent has, to answer a request
- Specify whether a user logging on needs to have a valid Openfire account

Because you can customize of all the parameters for every group, there are a couple of settings that all groups share, such as the amount of time after which a rejected request can be requested again, the number of times a request can be routed to other queues before the request is cancelled, and whether or not the agents are allowed to change their names. If you need to alter these parameters, you'll have to change them globally, and then they will apply to all workgroups.

Diversifying Workgroups with Queues

So there wasn't much to creating, configuring, and setting up the helpdesk, now was there? And now we get to the smart bit of the Fastpath plugin—queues.

Queues are routing patterns based on which Fastpath sends the request to a particular user/agent. Each queue can have its own members. For example, if your organization offers multiple levels of support, you can use queues to route a request to the correct one. Or, if you offer support for multiple products, you can have queues for every product, with their own dedicated agents, and have support requests routed to the correct agent. Let's set this up.

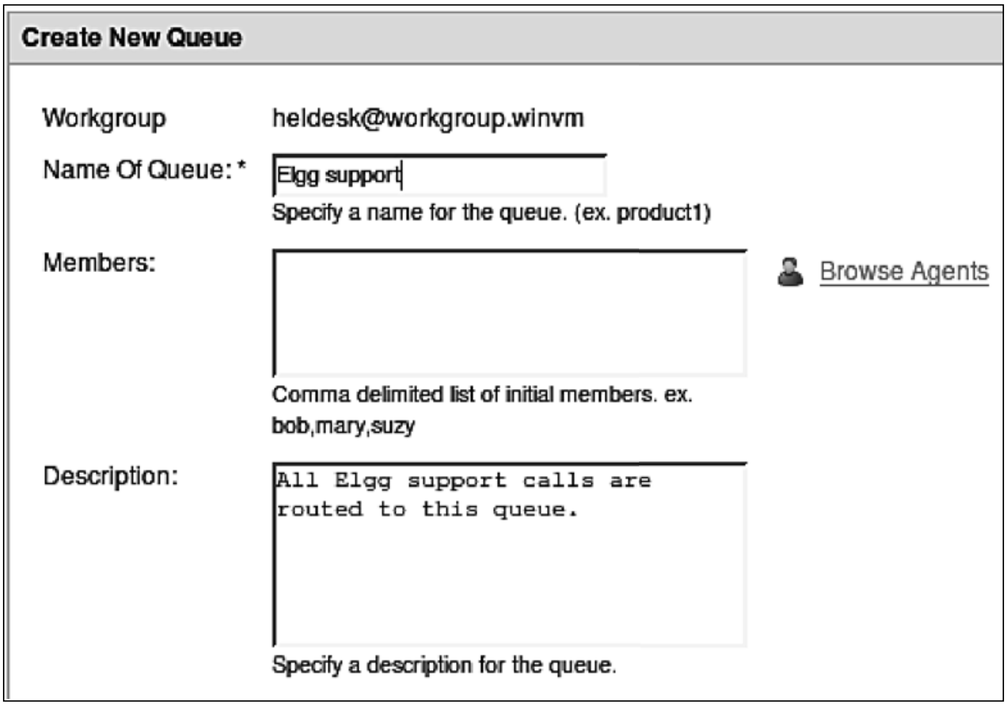
Creating Queues

Let's assume you provide support on Joomla! and Elgg. You'd obviously need a couple of queues for each of these. To create a queue, click on the workgroup you want the queues to be under. In our case, both of these queues are under the Helpdesk workgroup. From the expanded workgroup menu on the left, select the "Manage Queues" section, which is where you probably are already.

Here again, as with the Demo workgroup, you'll notice Fastpath already lists a queue called **Default Queue**. We'll leave it undisturbed. To add your queue, click on the **Add Queue** button at the top of the page. This displays a form that's quite similar to the form used for creating a workgroup, and asks for the following bits of information:

- Name of the queue
- Members that'll be part of the queue
- A brief description of the queue

As with the create workgroup form, the only bit of information necessary to enter here is the name of the queue. It's advisable to enter a brief description, but skip adding members.



Create New Queue

Workgroup	heldesk@workgroup.winvm
Name Of Queue: *	Elgg support Specify a name for the queue. (ex. product1)
Members:	 Comma delimited list of initial members. ex. bob,mary,suzy
Description:	All Elgg support calls are routed to this queue. Specify a description for the queue.

Repeat the process and create the queue for the Joomla! helpdesk.

Adding Routing Rules

Now that our queues are set, let's add rules to route requests to them. In the **Manage Queues** section, below the list of queues, you'll notice the **Routing Rules** section. Currently there aren't any queues listed. Scroll down a bit more until you get to the **Create New Routing Rule** section. Let's create a rule first and then we'll break it down and analyze what we just did.

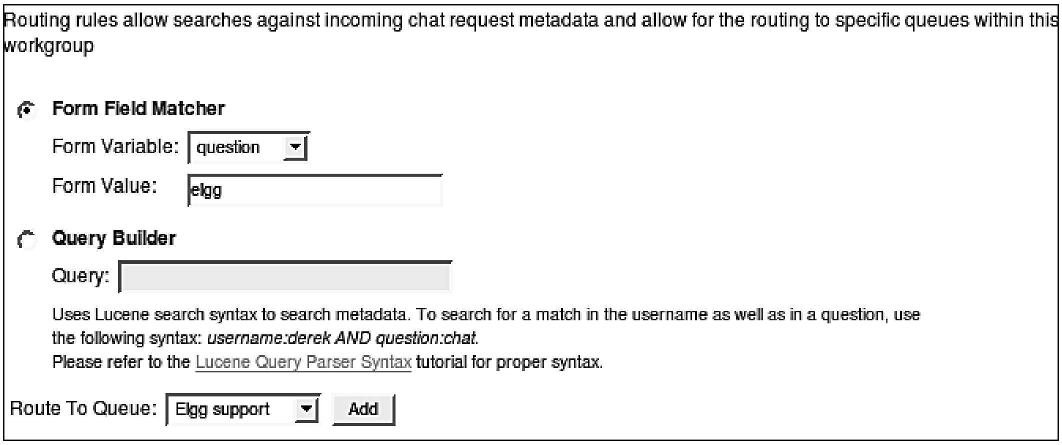
1. Make sure the option button next to **Form Field Matcher** is selected.
2. From the drop-down menu next to **Form Variable** select **question**, and in the **Form value** text box, enter the word **elgg**.
3. Now scroll down to the **Route To Queue** pull-down menu, select the **Elgg support** queue, and click on the **Add** button.

Presto! You'll notice the **Routing Rules** section isn't empty anymore. It now lists our newly-created rule.

Rule Book for Routing Rules

So let's analyze what we just did, although it's really quite obvious. We've created a rule that scans the form a user fills when he or she wishes to interact with a support agent. Our rule asks Fastpath to check if the question part of that form contains the word **elgg**. If it does, Fastpath is to route that support call to the queue specified—in this case, the **Elgg support** queue.

When we have to forward a request based on some part of the form the user has to fill, we have to select the **Form Field Matcher** option.


Routing rules allow searches against incoming chat request metadata and allow for the routing to specific queues within this workgroup

[✓ Form Field Matcher]

Form Variable: question

Form Value: elgg

Query Builder

Query:

Uses Lucene search syntax to search metadata. To search for a match in the username as well as in a question, use the following syntax: *username:derek AND question:chat*.
Please refer to the Lucene Query Parser Syntax tutorial for proper syntax.

Route To Queue: Elgg support

In addition to analyzing the "question" part of the form, we can also create a form to check for specific:

- Usernames
- Email addresses
- Agent names

We can also filter requests by using the **Query Builder**. The query builder uses the Lucene search syntax to identify patterns. Using the query builder, you can also look for patterns in more than just the form elements. Refer to the **Lucene Query Parser Syntax** tutorial `(see http://lucene.apache.org/java/docs/queryparsersyntax.html)` for more options and an explanation of the query syntax.

Before we move ahead, make sure that you create a query for identifying Joomla!-related questions and forwarding them to the Joomla! queue.

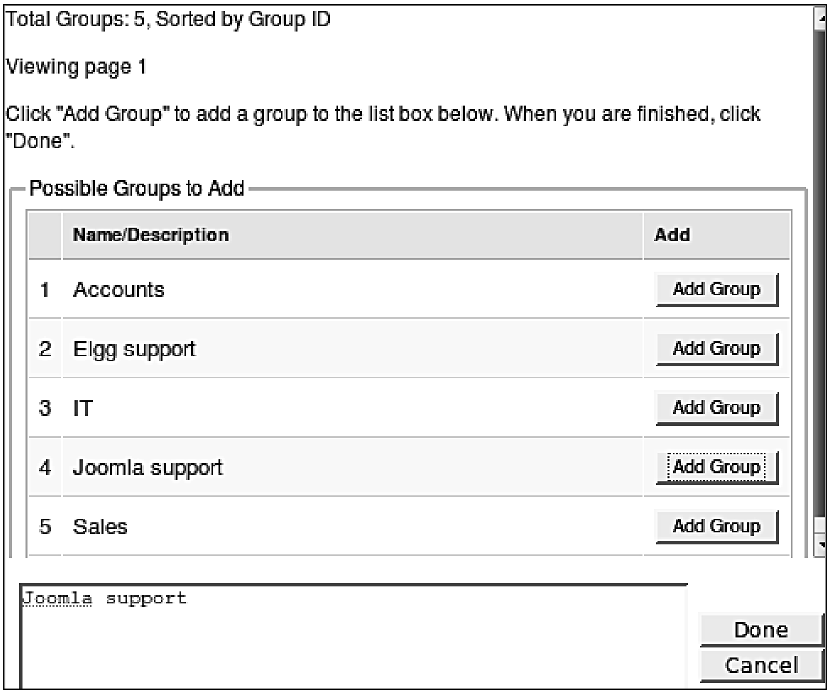
Adding Members to Queues

We've been putting off adding members to queues since the start of the chapter. Now that we have created a workgroup, created a couple of queues, and matching rules for forwarding requests, it's time we added some folks to receive those requests.

To add members to a queue, simply click on the queue under the **Manage Queues** section. The page that opens displays the queue settings at the top. This is followed by a text field to add individual agents or complete groups. I've been postponing adding users because it's easier to add them as a group.

For example, in our case, let's assume you have support agents in groups called **Joomla support** and **Elgg support**. If you don't have them in groups, it wouldn't be a bad idea to create them now, via the **Users/Groups** tab.

Once you have the groups in place, to add them to their respective queues, head back to **Fastpath | Workgroups Manager | View Workgroup | Helpdesk** and click on a queue, say, "Joomla!". Scroll down to the **Add Group(s)** section and click on the **Browse** link, which shows all of the groups under Openfire. To add a group, click on the **Add Group** button adjacent to that group.



Total Groups: 5, Sorted by Group ID

Viewing page 1

Click "Add Group" to add a group to the list box below. When you are finished, click "Done".

Possible Groups to Add

	Name/Description	Add
1	Accounts	Add Group
2	Elgg support	Add Group
3	IT	Add Group
4	Joomla support	Add Group
5	Sales	Add Group

```
Joomla support
```

When you're finished, click on the **Done** button at the bottom of the page, and you are returned to the **Manage Queues** page with your selected groups listed in the text box adjacent to **Add Group(s)**. Click on the **Add** button to finally add them to your queue.

You can follow the same procedure to add individual agents.

Make sure that you repeat these steps to add groups of agents and/or individual agents to the other queue as well, before moving to the next section.

Configuring The Helpdesk

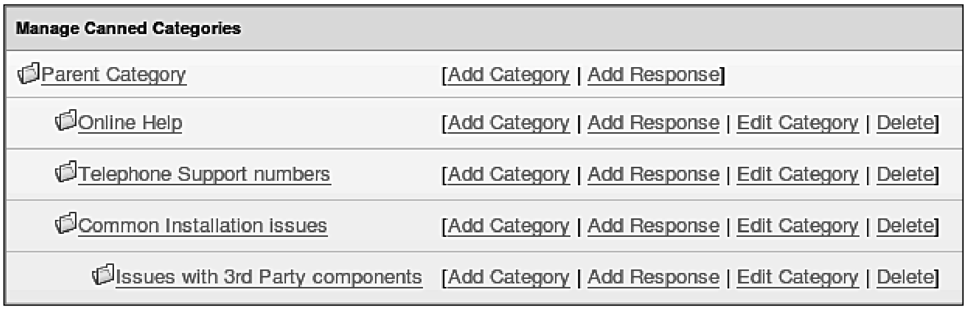
Before we put our helpdesk online, let's tweak it a little to add some users. We'll also add some canned responses to help agents quickly respond to certain questions.

Canned Responses

Let's begin with canned responses. Canned responses are predefined responses defined in anticipation of typical queries. For example, for a support company, URLs for a product's online help documentation, FAQs, forum boards, mailing lists, and so on are best handled as canned responses.

To add a canned response, head over to the **Workgroups Summary** (**Fastpath | Workgroups Manager | View Workgroups**) and click on the workgroup for which you want to define canned responses. In our case, this would be the **helpdesk** workgroup. Click on **Canned Responses** in the expanded workgroup menu in the left column.

This opens up the page that'll help you add and categorize canned responses. Categorizing responses helps agents find a response quickly. To create a category, click on the **Add Category** link and enter the name of the category in the page that opens. You can further add a sub-category within a category.



Manage Canned Categories	
Parent Category	[Add Category | Add Response]
Online Help	[Add Category | Add Response | Edit Category | Delete]
Telephone Support numbers	[Add Category | Add Response | Edit Category | Delete]
Common Installation issues	[Add Category | Add Response | Edit Category | Delete]
Issues with 3rd Party components	[Add Category | Add Response | Edit Category | Delete]

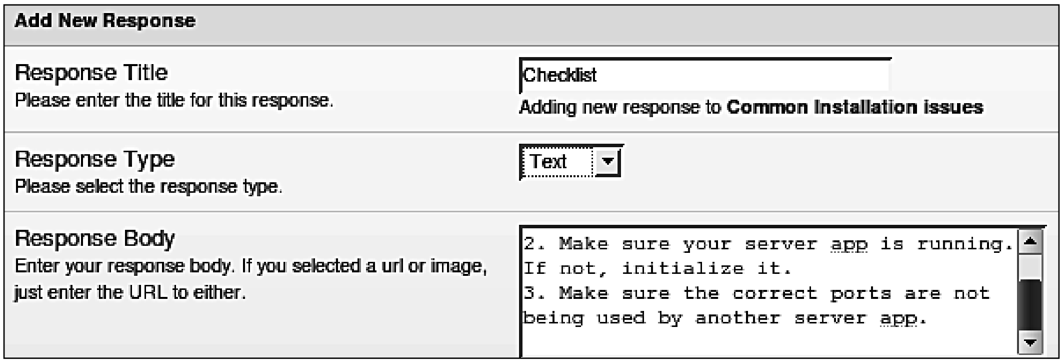
To enter a canned response, click on the **Add Response** link adjacent to the name of the category under which you want it. You can add three types of responses:

1. Text response: This is similar to a checklist or a greeting.
2. URL: This is useful for pointing to resources on the Internet.
3. Image: This is ideal for showing pictorial representations such as the flow of data, or typical network setups. The image is pointed to via a URL.

To add a canned response you need to enter:

1. **Response Title**: This can be descriptive to help the agent identify it.
2. **Response Type**: This can be one of the three options listed above.
3. **Response Body**: This is the actual response. Depending on the response type, the body can be a URL or several lines of text.

All canned responses and categories listed here are available to all agents of that particular workgroup.



Add New Response	
Response Title Please enter the title for this response.	Checklist Adding new response to **Common Installation issues**
Response Type Please select the response type.	Text
Response Body Enter your response body. If you selected a url or image, just enter the URL to either.	2. Make sure your server app is running. If not, initialize it. 3. Make sure the correct ports are not being used by another server app.

So in our case, all canned responses under the helpdesk will be available to agents of both the Elgg and Joomla! support groups. Individual agents can also set their own canned messages, which we'll discuss towards the end of the chapter.

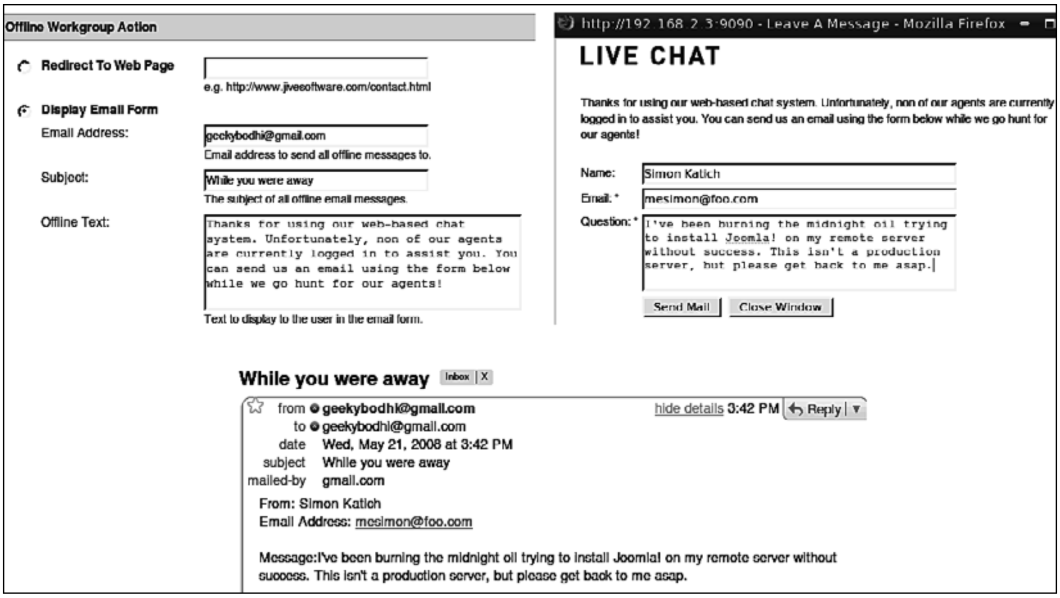
Offline Settings

What happens when a user wishes to contact you when none of your agents are online, say after business hours, or on holidays? Just turning them away wouldn't be acceptable. Fastpath offers two user features that can be employed when your agents are offline.

- **Redirect to another page**: If this is set, users are redirected to the website specified here. Typically, if online contact options are not set, you'd want to redirect users to a traditional "Contact Us" page which lists other offline means of contacting you.
- **Email form**: The other option you can use is to let Fastpath display an email form to collect information from the user, which is then passed on to you. In the simple configuration form, you need to specify the following details:
    - **Email Address**: The email address to which you want the offline form to be sent. There will probably be someone from the helpdesk department who intercepts the messages, and then redirects them to the correct department.
    - **Subject**: Because the email address to which the offline form is being sent receives other messages as well, a crisp, clear, and unique subject line helps it stand out from the other messages in the mailbox. This is similar to an "Offline message from a website".

- **Offline Text**: This is the text that will be displayed to the user while asking him or her to complete the email form. It is a good idea to concentrate the text on explaining why they (the users) are typing in an email form instead of talking with an agent.

After the offline action has been set, Fastpath will take the appropriate steps if none of the selected agents is online. If you've set up an email form, the visitor on the website is given the opportunity to send you an email. Using your email settings from Openfire, Fastpath will deliver this email to the email address specified.


Offline Workgroup Action

Redirect To Web Page:
e.g. http://www.jiveoftware.com/contact.html

[✓ Display Email Form]
Email Address: geekybodhi@gmail.com
Email address to send all offline messages to.
Subject: While you were away
The subject of all offline email messages.
Offline Text: Thanks for using our web-based chat system. Unfortunately, non of our agents are currently logged in to assist you. You can send us an email using the form below while we go hunt for our agents!
Text to display to the user in the email form.

http://192.168.2.3:9090 - Leave A Message - Mozilla Firefox

LIVE CHAT

Thanks for using our web-based chat system. Unfortunately, non of our agents are currently logged in to assist you. You can send us an email using the form below while we go hunt for our agents!

Name: Simon Katich
Email: * mesimon@foo.com
Question: * I've been burning the midnight oil trying to install Joomla! on my remote server without success. This isn't a production server, but please get back to me asap.

While you were away

from geekybodhi@gmail.com
to geekybodhi@gmail.com
date Wed, May 21, 2008 at 3:42 PM
subject While you were away
mailed-by gmail.com

From: Simon Katich
Email Address: mesimon@foo.com

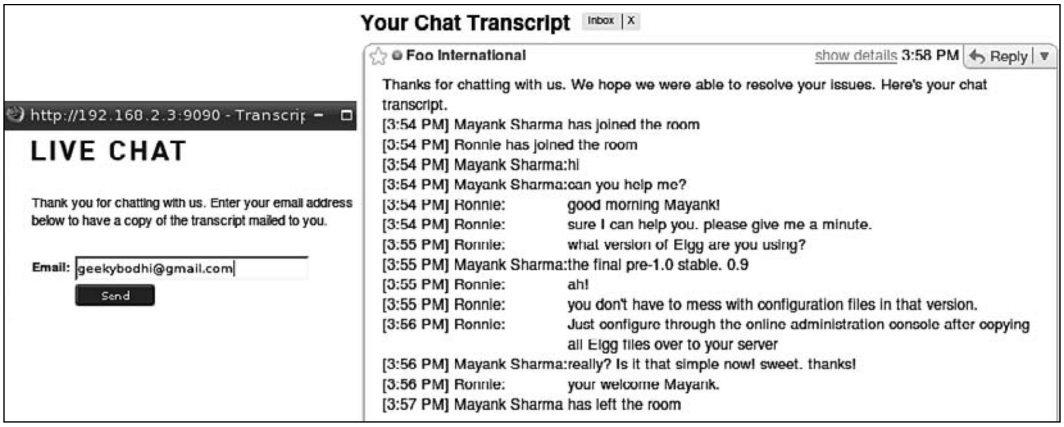
Message:I've been burning the midnight oil trying to install Joomla! on my remote server without success. This isn't a production server, but please get back to me asap.

While replying, make sure that you use the email address mentioned in the message body. If you just click reply, the message will land in your own mail server, so be careful.

Chat Transcript

If the online user hadn't been taking notes during the chat, it's quite possible that he or she might run into the same problem later and wouldn't recall the solution especially if the solution involved a series of steps. Even if the solutions and steps outlined by the agent were followed immediately, it's also quite possible that in a long discussion, the user might lose his or her way, or might miss a minor check.

This is where chat transcripts help. When set and configured, after a session has ended, Fastpath asks whether the user would like a transcript of the chat session to be emailed to the user. If he or she chooses to do so, Fastpath emails a time-stamped copy of the transcript to the user.



Thank you for chatting with us. Enter your email address below to have a copy of the transcript mailed to you.

Email: geekybodhi@gmail.com

```
Thanks for chatting with us. We hope we were able to resolve your issues. Here's your chat
transcript.
[3:54 PM] Mayank Sharma has joined the room
[3:54 PM] Ronnie has joined the room
[3:54 PM] Mayank Sharma:hi
[3:54 PM] Mayank Sharma:can you help me?
[3:54 PM] Ronnie:          good morning Mayank!
[3:54 PM] Ronnie:          sure I can help you. please give me a minute.
[3:55 PM] Ronnie:          what version of Elgg are you using?
[3:55 PM] Mayank Sharma:the final pre-1.0 stable. 0.9
[3:55 PM] Ronnie:          ah!
[3:55 PM] Ronnie:          you don't have to mess with configuration files in that version.
[3:56 PM] Ronnie:          Just configure through the online administration console after copying
                           all Elgg files over to your server
[3:56 PM] Mayank Sharma:really? Is it that simple now! sweet. thanks!
[3:56 PM] Ronnie:          your welcome Mayank.
[3:57 PM] Mayank Sharma has left the room
```

To set this up, head over to **Fastpath | Workgroups Manager | View Workgroups**, and select the workgroup for which you want to configure transcripts; in our case it would be the helpdesk workgroup. In the expanded workgroup menu, select the **Transcripts** section and enter the following details in the transcript setup form:

- **From**: The name that will be displayed in the **From** section of the email. Typically, you'd want to specify the name of your organization/company.
- **Email Address**: The email address from which the message will be sent typically, your support id.
- **Subject**: The subject of the email containing the transcript. Something like "Your Chat Transcript" will make it easily searchable.
- **Message**: The message that accompanies the transcript.

Once set, as soon as a chat session ends, the user is presented with the option to receive the transcript to the email address he or she used when entering the chat. The user has the option to have the transcript sent to another email address.

Putting the Helpdesk Online

So we have our helpdesk setup, with agents sorted into the two support queues, ready with their canned responses. Hmm...I am pretty sure we are missing something. Oh! There isn't any mechanism for our site visitors to get in touch with us. It is time to put our helpdesk online.

Design a Contact Form

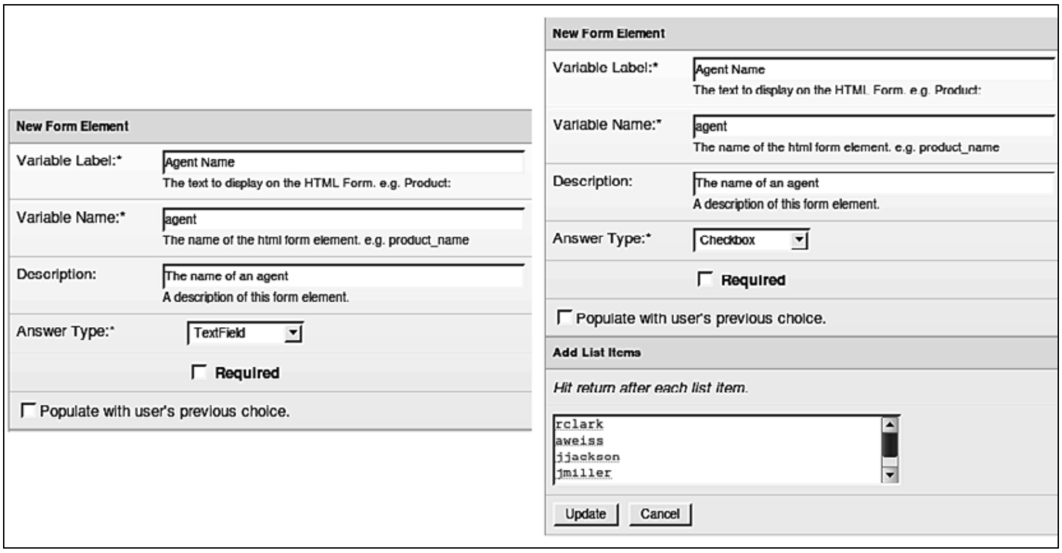
Before you put up a button for users to click on, you'll have to create a form to gather information from them. For this, head over to **Fastpath | Workgroups Manager | View Workgroups** and select your workgroup from the list. Now, in the extended workgroup menu, select the **Form UI** section. This is where you create an HTML form for your users. By default, the form lists three pieces of information which all users are required to provide:

- Their name.
- Their email addresse
- Their question

You can customize this form to add more fields. Let's add another field to this form where a user can specify the name of the agent he or she wants to talk to, in case he or she has a favorite one. Immediately below the three fields, you'll notice an **Add Field** button which, when clicked, requests the following bits of information:

- **Variable Label**: The label with which to identify this field in the Form UI. This label can contain spaces.
- **Variable Name**: The name of the variable that will be used by Fastpath. This name cannot contain spaces.
- **Description**: A brief description of the variable, to help recall its use.
- **Answer Type**: The type of answer you want from the user. In our case, we can either have a **TextField** and let the user enter the name of his or her favorite agent, or have a drop-down box of agent names that they can select from.
- **Required**: This is a checkbox that specifies whether this variable can be left blank by the user or not. Because we have a smart queue acting at the back, we don't want to force the users to pick a user, so we'll leave it optional.
- **Populate with user's previous choice**: If selected, the choice is stored in a user's cache and upon his or her return to the helpdesk, the form values from his or her last visit are pre-filled.
- **Add List items**: If we have selected any of the multiple choice options in the **Answer Type** field above such as a drop-down box, a checkbox, or a radio button, then we'll use this area to specify the choices.

Once you've specified all required information, click the **Update** button.



New Form Element

Variable Label:*	Agent Name The text to display on the HTML Form. e.g. Product:
Variable Name:*	agent The name of the html form element. e.g. product_name
Description:	The name of an agent A description of this form element.
Answer Type:*	TextField
	☐ Required
☐ Populate with user's previous choice.	

New Form Element

Variable Label:*	Agent Name The text to display on the HTML Form. e.g. Product:
Variable Name:*	agent The name of the html form element. e.g. product_name
Description:	The name of an agent A description of this form element.
Answer Type:*	Checkbox
	☐ Required
☐ Populate with user's previous choice.	

Add List Items

Hit return after each list item.

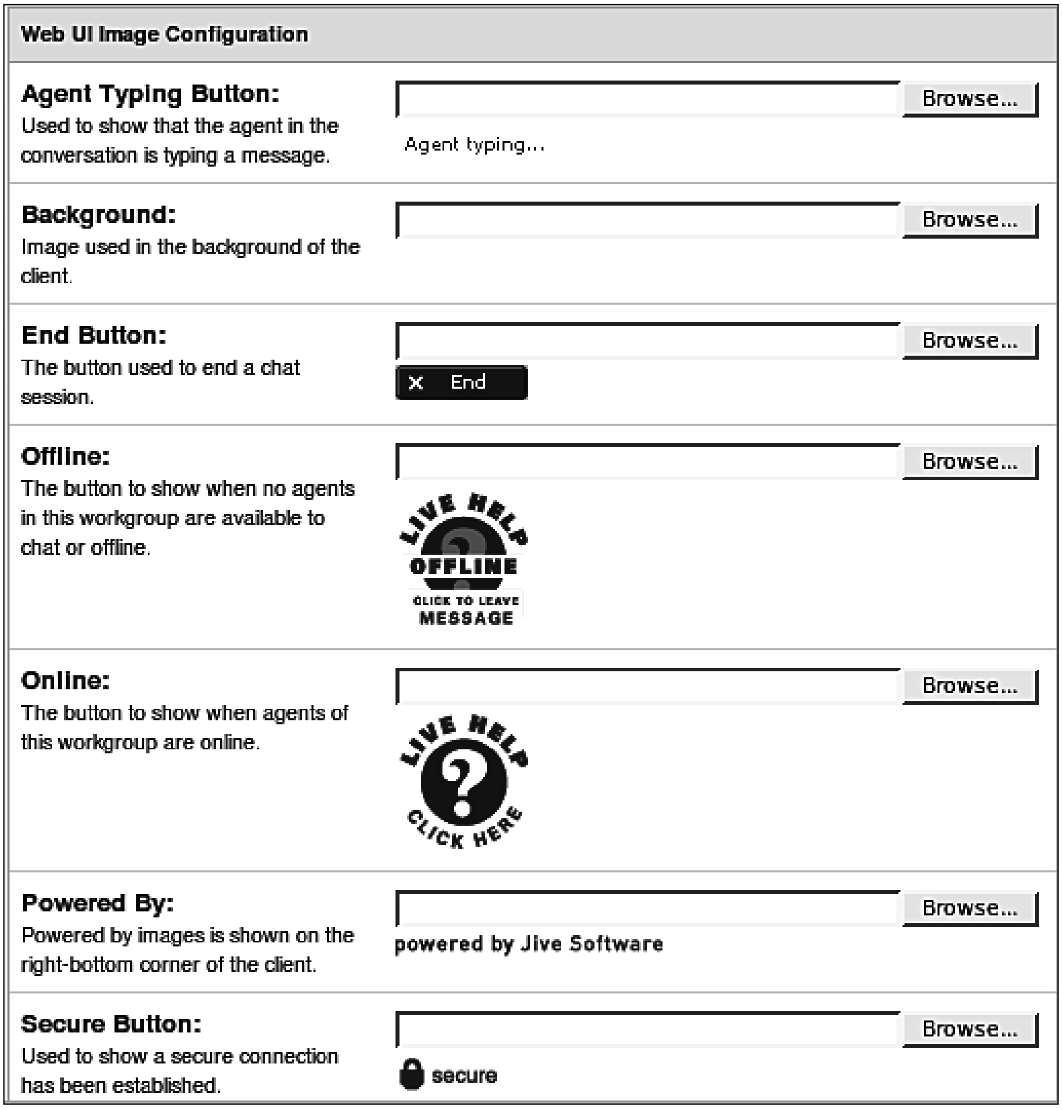
```
rclark
aweiss
jjackson
jmiller
```

When we've added a variable, we are returned to the **Form UI** page, with our new variable listed at the bottom of the form. You can use the up and down arrows adjacent to the form variable to change the order in which the variables are listed on the final form.

Once you are done, click on the **Save Changes** button to save the form.

Review Images and Text

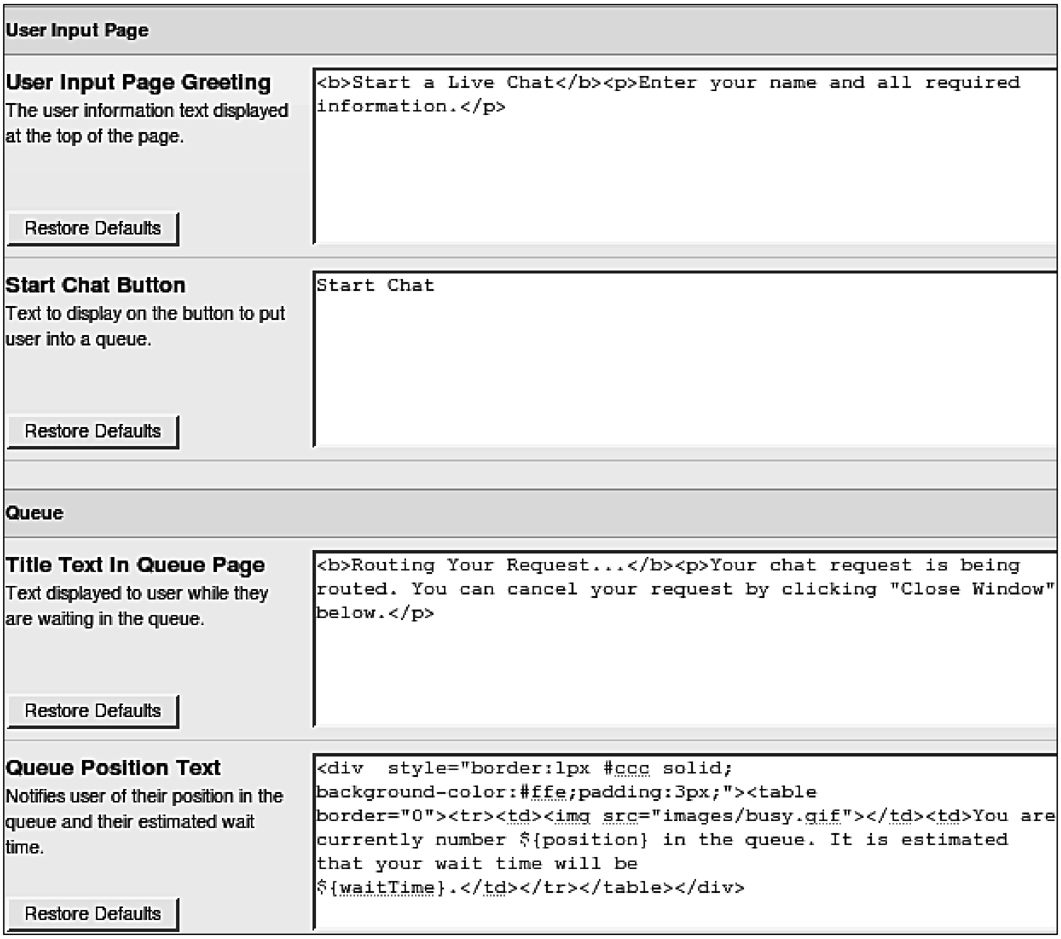
We are almost there. The Fastpath plugin bundles a set of images and animations you can use to design the user interface for the user. You can preview the images under the **Images** section of each workgroup. If you plan to upload your own images, make sure they are of the same size as the existing ones. Also make sure that you replace the existing images with yours in the same folder as the original images.



Web UI Image Configuration

Agent Typing Button: Used to show that the agent in the conversation is typing a message.	Browse... Agent typing...
Background: Image used in the background of the client.	Browse...
End Button: The button used to end a chat session.	Browse... ✕ End
Offline: The button to show when no agents in this workgroup are available to chat or offline.	Browse... LIVE HELP OFFLINE CLICK TO LEAVE MESSAGE
Online: The button to show when agents of this workgroup are online.	Browse... LIVE HELP ? CLICK HERE
Powered By: Powered by images is shown on the right-bottom corner of the client.	Browse... powered by Jive Software
Secure Button: Used to show a secure connection has been established.	Browse... secure

The plugin also has pre-defined text to do everything from greeting the user, to conveying various details such as his or her position in the queue, and pointing out errors in case of an operation failure. You can review all bits of text from the **Text** section under each workgroup.



User Input Page

User Input Page Greeting
The user information text displayed at the top of the page.

```
<b>Start a Live Chat</b><p>Enter your name and all required
information.</p>
```

Start Chat Button
Text to display on the button to put user into a queue.

```
Start Chat
```

Queue

Title Text In Queue Page
Text displayed to user while they are waiting in the queue.

```
<b>Routing Your Request...</b><p>Your chat request is being
routed. You can cancel your request by clicking "Close Window"
below.</p>
```

Queue Position Text
Notifies user of their position in the queue and their estimated wait time.

```
<div  style="border:1px #ccc solid;
background-color:#ffe;padding:3px;"><table
border="0"><tr><td><img src="images/busy.gif"></td><td>You are
currently number ${position} in the queue. It is estimated
that your wait time will be
${waitTime}.</td></tr></table></div>
```

If you make changes to any of the text boxes, make sure that you use HTML syntax where required.

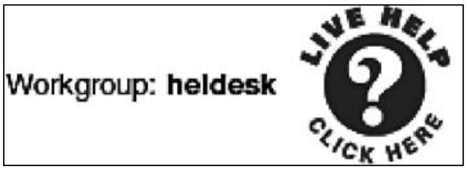
Upload the HTML

Once you've reviewed all images and pieces of text, identify the page on your website to which you want to add the **Chat** button. Open that page in your favorite HTML editor, and copy and paste the following HTML snippet into it , at the position at which you want to the chat button to appea:

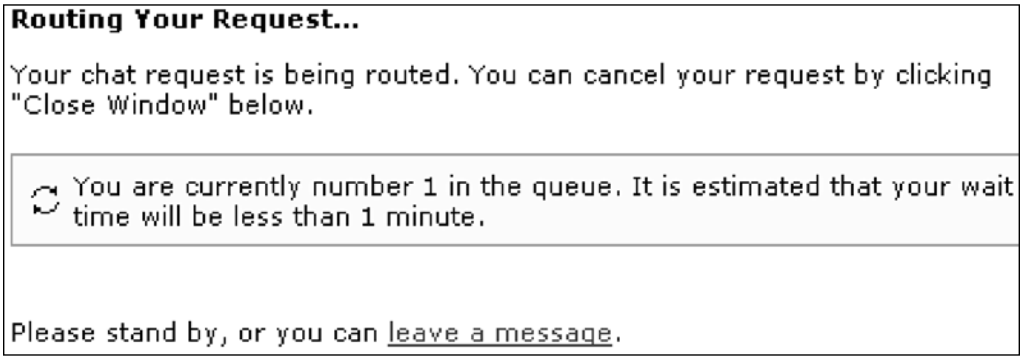
```
<!-- Insert this snippet where you would like the Chat button image to
show up -->
<script language="JavaScript" type="text/javascript"
         src="url to jivelive.jsp"></script>
<script>showChatButton('helpdesk@workgroup.winvm');</script>
<!-- End Of Spark Fastpath Snippet -->
```

Make sure you edit the `url to jivelive.jsp` bit with the actual URL of the file on your server. This file will be under the `webchat` folder under your application server, or in our case, under the Openfire's `plugin` folder because we have installed it as a plugin under Openfire itself.

This JavaScript snippet will display a **Live Help—Click Here** icon, if you haven't replaced it with your own image. As you can see, this button is for our helpdesk example. If you have multiple workgroups, you'll have to copy and paste JavaScript snippets for each. These snippets are also listed at the top of the **Text** section under each workgroup.



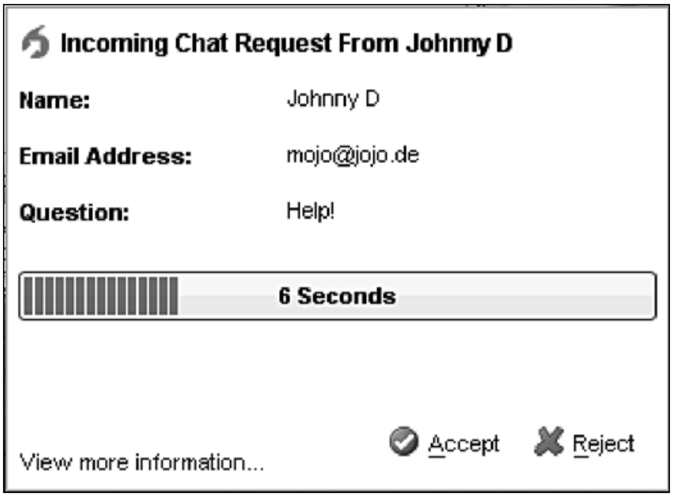
Congratulations, you're all set. When the icon is clicked, Fastpath presents users with a form. Depending on their question, Fastpath then routes the call to a free agent in the respective queue.


Routing Your Request...

Your chat request is being routed. You can cancel your request by clicking "Close Window" below.

You are currently number 1 in the queue. It is estimated that your wait time will be less than 1 minute.

Please stand by, or you can leave a message.


Incoming Chat Request From Johnny D

Name:	Johnny D
Email Address:	mojo@jojo.de
Question:	Help!

If the call isn't answered, it's forwarded to another agent until someone picks it up, or it is timed out.

After the session is closed, Fastpath offers to send the transcript of the chat to the user's email address.

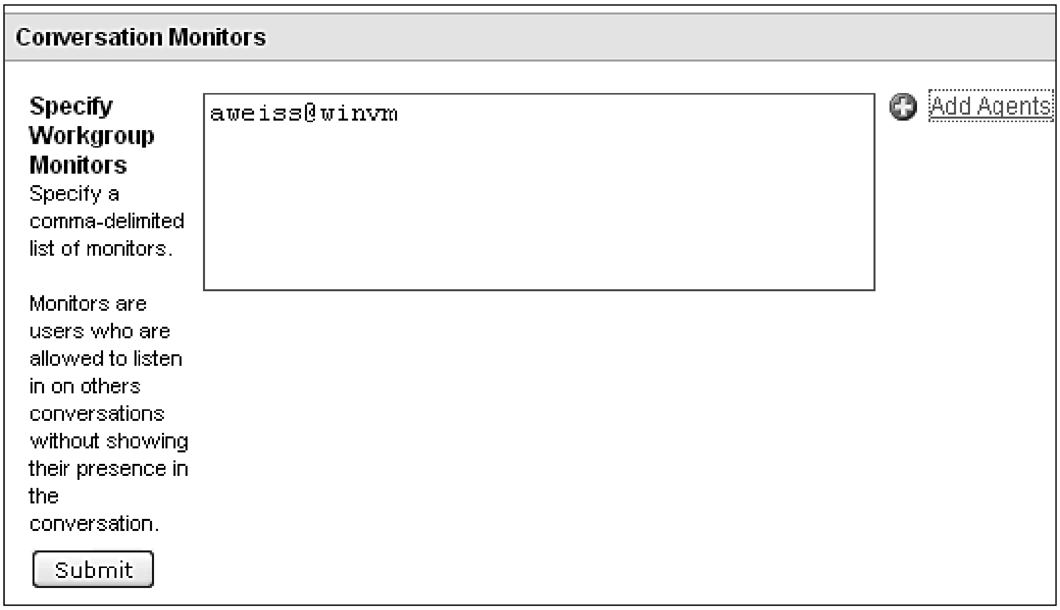
Sweet!

Monitoring Sessions, Reports, and Usage Statistics

Once you stop jumping in excitement, you'll notice that your role as administrator doesn't quite end just now. In a support service environment, it's very common for team leaders to listen in on conversations, to provide input for an agent's performance assessment and review. In this section, we'll give two users the power to secretly listen in on any conversation. We'll also look at Fastpath's reporting tools for browsing through chat history.

Snooping Super Users

Users who can secretly listen in ongoing conversations are known as room monitors, in Fastpath speak. To add one (or more) to a workgroup, head over to **Fastpath | Workgroups Manager | View Workgroups** and select your workgroup. In the extended workgroup settings, click on the **Room Monitors** link.



Conversation Monitors

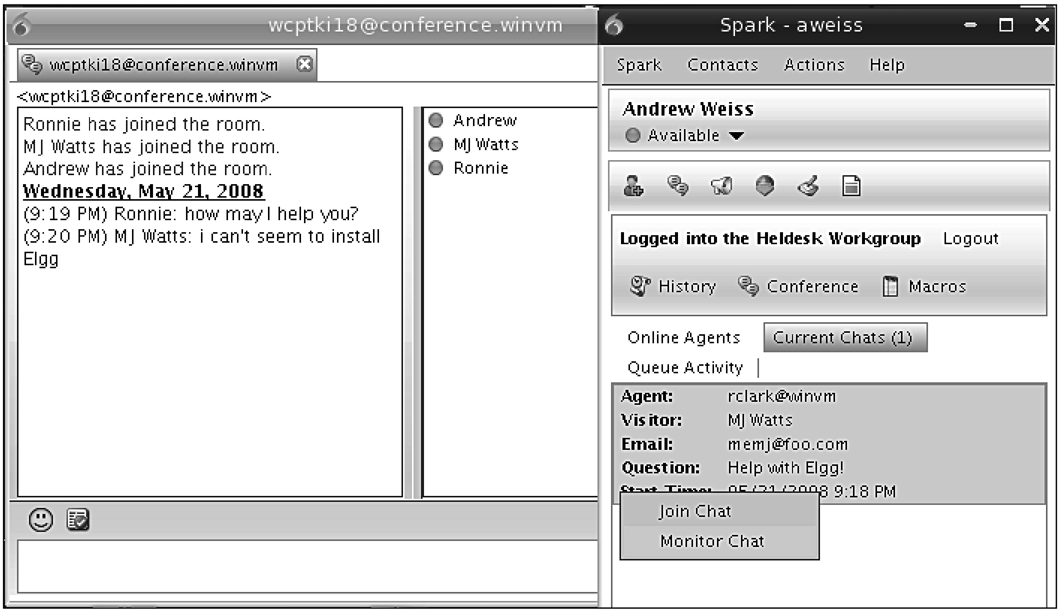
Specify Workgroup Monitors
Specify a comma-delimited list of monitors.

Monitors are users who are allowed to listen in on others conversations without showing their presence in the conversation.

aweiss@winvm

In the page that opens you see a huge textbox where you can add as many users as you wish. All of these users will have the power to listen in the conversation of other users, with their presence being hidden. To help you look for users, the page has an **Add Agents** button, which pops open a list of all of the users in the system. Select the users you want to designate as room monitors, save the settings, and there you go.

To monitor calls, a super user can right-click on any ongoing session and choose the **Monitor Chat** option, as shown in the following image.



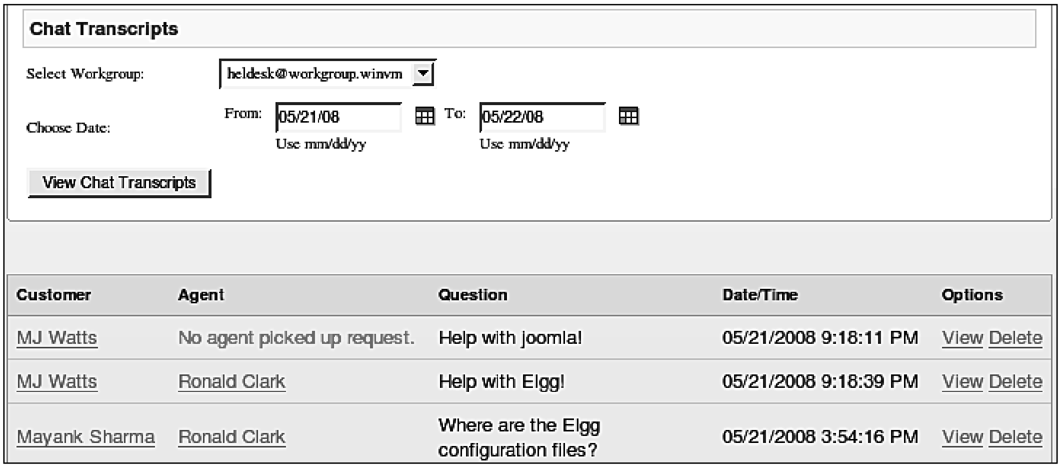
<wcptki18@conference.winvm>
Ronnie has joined the room.
MJ Watts has joined the room.
Andrew has joined the room.
Wednesday, May 21, 2008
(9:19 PM) Ronnie: how may I help you?
(9:20 PM) MJ Watts: i can't seem to install Elgg

Agent: rclark@winvm
Visitor: MJ Watts
Email: memj@foo.com
Question: Help with Elgg!
Start Time: 05/21/2008 9:18 PM

Chat Archives

Do you remember how we configured Fastpath to offer chat transcripts to users after their individual sessions? Fastpath also logs time-stamped archives of all the conversations agents have had with the users, and all unanswered calls.

To search the archives head over to **Fastpath | Reports**. In the **View Previous Chats** section, you need to select the workgroup for which you want to search the archives and then enter a date range. There's also a handy little JavaScript calendar that saves you the effort of typing in the dates yourself.


Chat Transcripts

Select Workgroup:	heldesk@workgroup.winvm
Choose Date:	From: 05/21/08 Use mm/dd/yy To: 05/22/08 Use mm/dd/yy

View Chat Transcripts

Customer	Agent	Question	Date/Time	Options
MJ Watts	No agent picked up request.	Help with joomla!	05/21/2008 9:18:11 PM	View Delete
MJ Watts	Ronald Clark	Help with Elgg!	05/21/2008 9:18:39 PM	View Delete
Mayank Sharma	Ronald Clark	Where are the Elgg configuration files?	05/21/2008 3:54:16 PM	View Delete

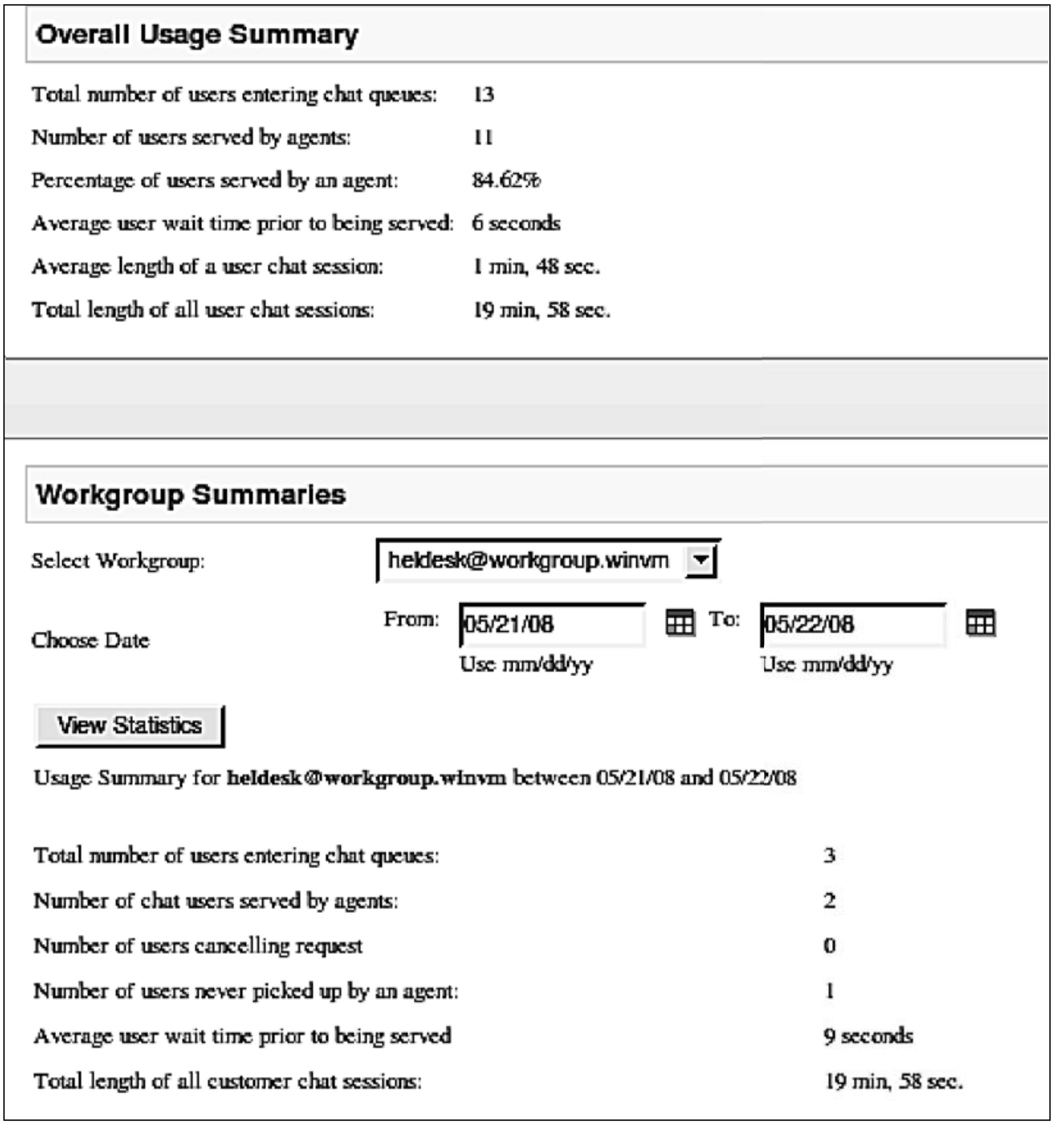
From the search results, you can find out the name and email address of the user, the questions asked, if and when the call was answered and by whom, and of course the actual chat transcript as well.

While this is a very handy way to look at transcripts, it has its shortcomings. For one, the search system requires that the two date parameters be different, and hence doesn't allow for same day searches. Secondly, because agents are allowed to change their names, the links to the agent names wouldn't necessarily lead to the agent's user properties page in Openfire. For example, the chat search might return an agent named Ronnie, whose real username in the system is Ronald. So if you let your agents change their names, make sure you keep a list of your agent's secret identities and their real names handy.

The other link under the **Fastpath | Reports** is the **View Chats Usage**. This page gives you statistics regarding the chat sessions including:

- **Total number of users entering chat queues**
- **Number of users served by agents**
- **Percentage of users served by an agent**
- **Average user wait time prior to being served**
- **Average length of a user chat session**
- **Total length of all user chat sessions**

You can also get these statistics for individual workgroups between particular dates, and then compare them with the total statistics for that group.



Overall Usage Summary

Total number of users entering chat queues:	13
Number of users served by agents:	11
Percentage of users served by an agent:	84.62%
Average user wait time prior to being served:	6 seconds
Average length of a user chat session:	1 min, 48 sec.
Total length of all user chat sessions:	19 min, 58 sec.

Workgroup Summaries

Select Workgroup: heldesk@workgroup.winvm

Choose Date From: 05/21/08 Use mm/dd/yy To: 05/22/08 Use mm/dd/yy

View Statistics

Usage Summary for **heldesk@workgroup.winvm** between 05/21/08 and 05/22/08

Total number of users entering chat queues:	3
Number of chat users served by agents:	2
Number of users cancelling request	0
Number of users never picked up by an agent:	1
Average user wait time prior to being served	9 seconds
Total length of all customer chat sessions:	19 min, 58 sec.

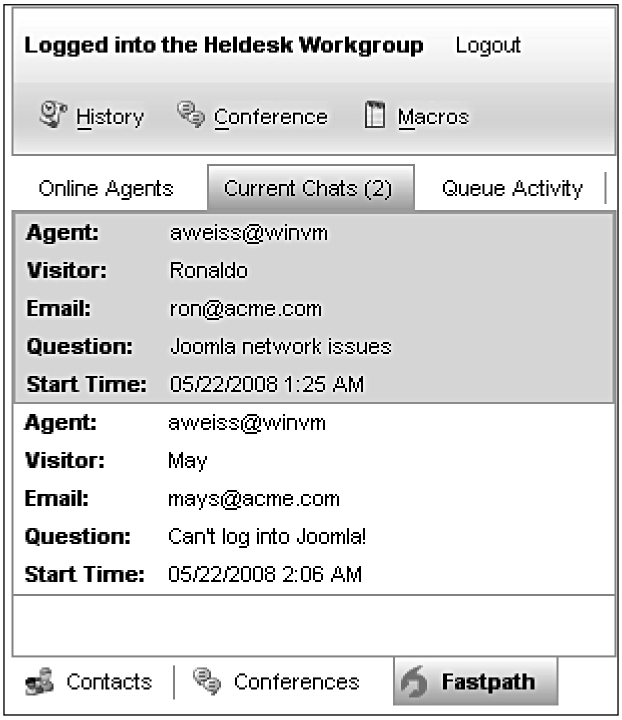
In the last section of this chapter, we'll look at the chat sessions from an agent's point of view to understand what he or she can do during a session.

The Agent's Perspective

All through this chapter we've been discussing the Fastpath service from the perspective of the administrator—how to set up a workgroup, how to add queues, routing rules, agents, and so on. So now you know what you as an administrator can do. But that's not the complete picture. There's the agent's side to the story as well, which plays a critical role in selling Fastpath to the management.

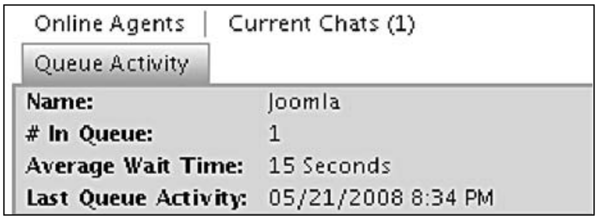
This section looks at the features and function available to an agent to help him or her do his or her job better.

- **Current chats**: When signed into Fastpath, all agents can look at the ongoing chats. They can also look at the question and the agent answering the chat. Room monitors are the only users who can right-click on the ongoing chats and either join them to assist the agent, or join in stealth mode for monitoring purposes.



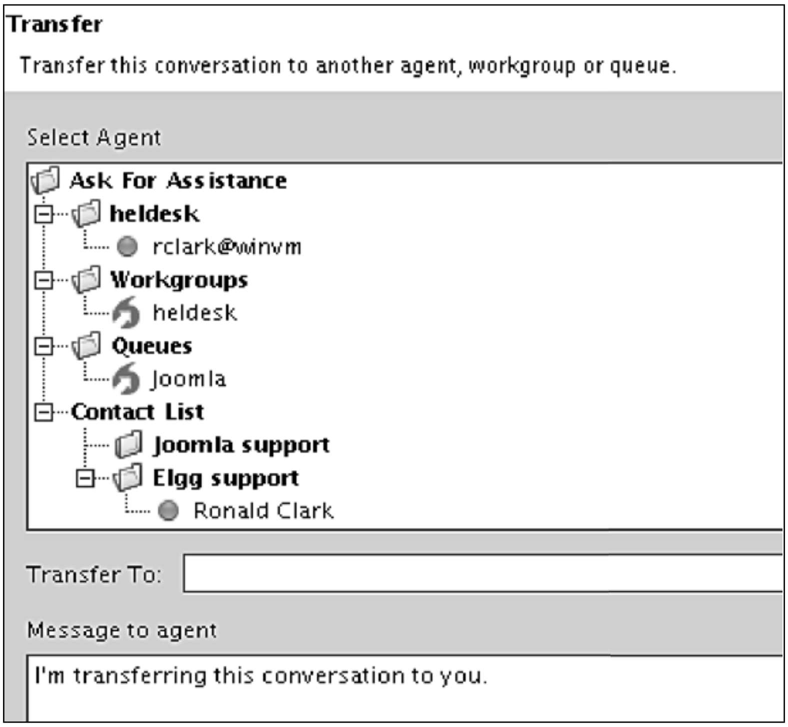
Agent:	aweiss@winvm
Visitor:	Ronaldo
Email:	ron@acme.com
Question:	Joomla network issues
Start Time:	05/22/2008 1:25 AM
Agent:	aweiss@winvm
Visitor:	May
Email:	mays@acme.com
Question:	Can't log into Joomla!
Start Time:	05/22/2008 2:06 AM

- **Queue activity**: Let agents monitor their respective queues. Combined with the ongoing chats, the agents online, and the wait time, it lets users accept waiting queues even if they are already in the middle of one.



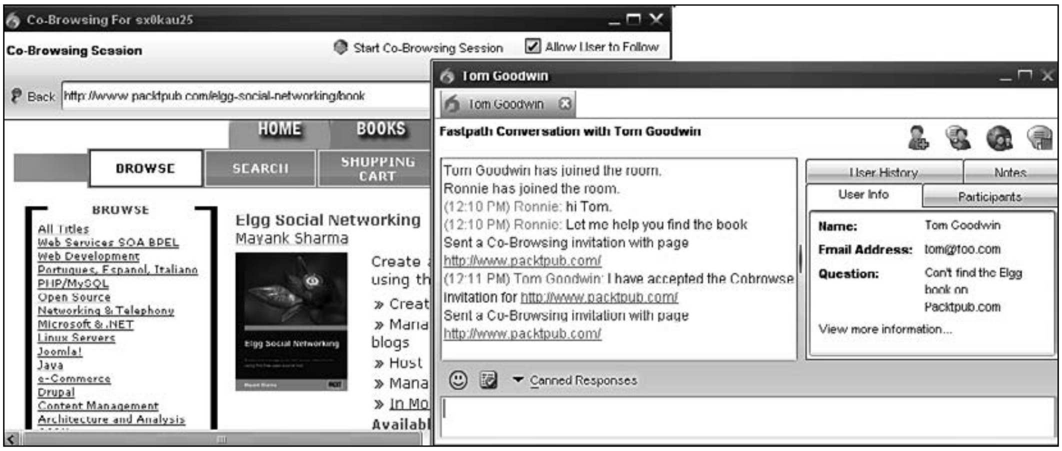
Name:	Joomla
# In Queue:	1
Average Wait Time:	15 Seconds
Last Queue Activity:	05/21/2008 8:34 PM

- **Transferring chats:** An agent has the ability to transfer chats to other agents. He or she could do so for a variety of reasons, but would most probably do so if he or she is unable to help the user, or has got too much on his or her plate while another agent is free. Fastpath displays a hierarchical menu of the agents who are online, categorized by their workgroups and queues. While transferring the user, an agent can also add a message to the new agent. The chat isn't transferred until the receiving agent accepts the request.



I'm transferring this conversation to you.

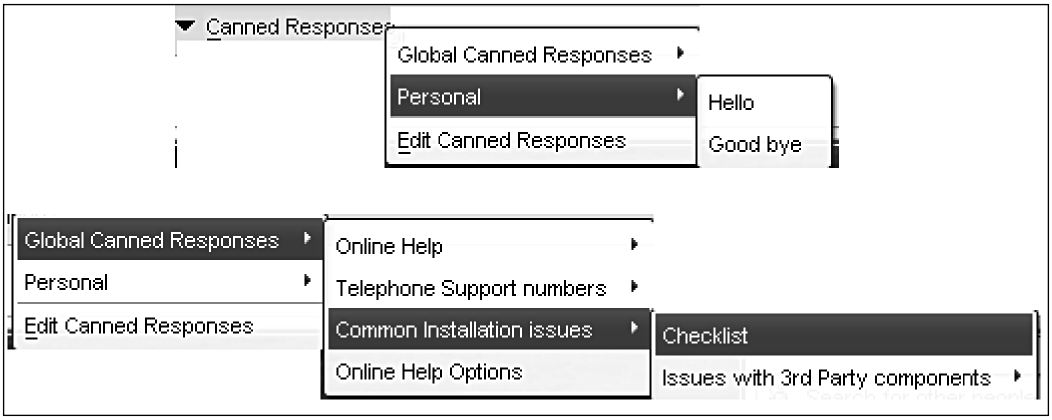
- **Inviting other agents:** If an agent is unable to help a user, it's not necessary to transfer the user to another agent. Instead, the agent can invite another agent to join the chat session. Again, when this option is used, Fastpath displays a hierarchical list of users currently online. When an invited agent joins in, he or she can both see and be seen by other members in the conversation. An agent can invite any number of other agents to join a conversation.
- **Initiate co-browsing:** This is another smart feature of Fastpath. If your agents want to help a user find a resource on the Web or demonstrate an example installation, the agent can initiate a co-browsing session with the user. In a co-browsing session, both the user and the agent share a browser window. The agent takes control of the browsing, and the user's window mimics the agent's movement across the website.



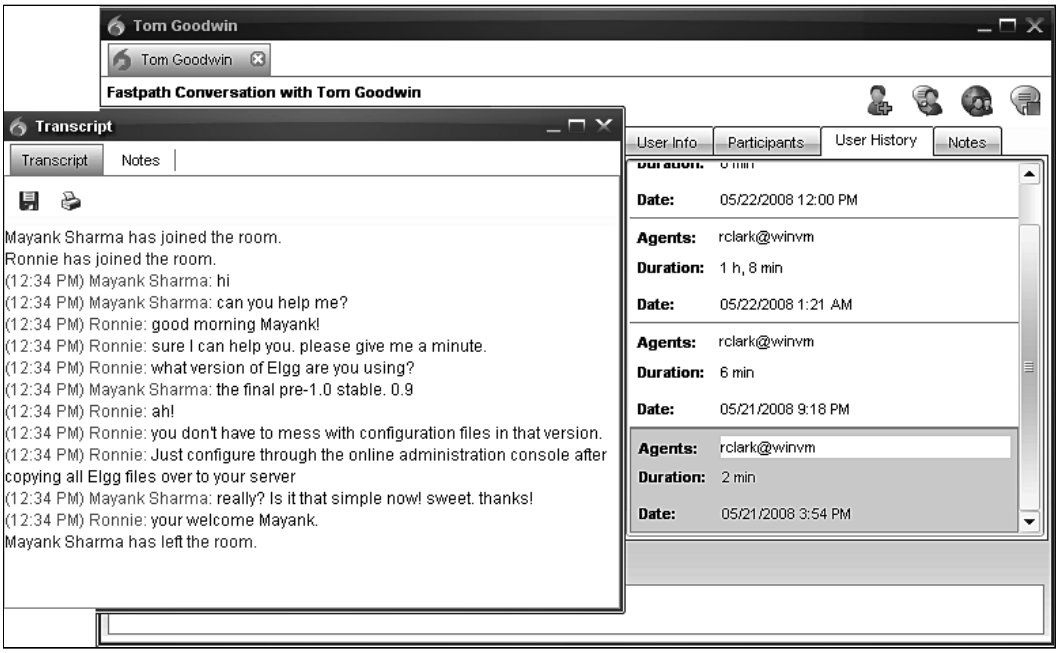
Tom Goodwin has joined the room.
Ronnie has joined the room.
(12:10 PM) Ronnie: hi Tom.
(12:10 PM) Ronnie: Let me help you find the book
Sent a Co-Browsing invitation with page http://www.packtpub.com/
(12:11 PM) Tom Goodwin: I have accepted the Cobrowse invitation for http://www.packtpub.com/
Sent a Co-Browsing invitation with page http://www.packtpub.com/

Name: Tom Goodwin
Email Address: tom@foo.com
Question: Can't find the Elgg book on Packtpub.com
View more information...

- Personal canned responses: As I mentioned earlier when discussing global canned responses, Fastpath allows individual agents to add personal ones, too. Personal canned responses are referred to as "Personal Macros", but are listed alongside the global canned responses in the agent's chat interface.



- **User history and notes**: During a conversation, an agent can take notes, which Fastpath saves along with the user's chat history and transcript. All of this information is available to any agent who chats with the user on the user's next visit.



Mayank Sharma has joined the room.
Ronnie has joined the room.
(12:34 PM) Mayank Sharma: hi
(12:34 PM) Mayank Sharma: can you help me?
(12:34 PM) Ronnie: good morning Mayank!
(12:34 PM) Ronnie: sure I can help you. please give me a minute.
(12:34 PM) Ronnie: what version of Elgg are you using?
(12:34 PM) Mayank Sharma: the final pre-1.0 stable. 0.9
(12:34 PM) Ronnie: ah!
(12:34 PM) Ronnie: you don't have to mess with configuration files in that version.
(12:34 PM) Ronnie: Just configure through the online administration console after copying all Elgg files over to your server
(12:34 PM) Mayank Sharma: really? Is it that simple now! sweet. thanks!
(12:34 PM) Ronnie: your welcome Mayank.
Mayank Sharma has left the room.

Date:	05/22/2008 12:00 PM
Agents:	rclark@winvm
Duration:	1 h, 8 min
Date:	05/22/2008 1:21 AM
Agents:	rclark@winvm
Duration:	6 min
Date:	05/21/2008 9:18 PM
Agents:	rclark@winvm
Duration:	2 min
Date:	05/21/2008 3:54 PM

Isn't that something? As you can see, Fastpath is not only a breeze to setup but is also adequately equipped for the agents to make sure the chatting experience is a good one for your users.

Summary

This was a very loaded chapter. It examined at just one aspect of the Openfire server which isn't even implemented as a core feature, but which is invaluable if you wish to use Openfire to interface with your online users. It could be used for a casual, brief session to help a user find your store, or for something more complex and critical such as running a comprehensive online support business. Adding this online chat facility is a breeze thanks to the Fastpath plugin. To give your users a nice web-based interface, the Openfire project also bundles the web chat application, which you can download and install on your application servers.

In this chapter, we've looked at installing and configuring these plugins and using them to create an online helpdesk with different queues to manage different products. We have also looked at some useful functions for both clients and agents, such as transcripts, canned responses, offline email forms, and so on, and configured them before adding users and user groups to our helpdesk and putting it online. The last section of the chapter illustrated what a Fastpath session feels like to a user and the various tools (transferring calls, user history, personal canned responses, and so on) at the agent's disposal to enable him or her to better serve the users.

This also brings us to the end of the book. It took us a mere eleven chapters to get from a compressed archive of files and folders to a comprehensive, multioffice, and multiserver chat server deployment, handling thousands of users across continents and over the Internet.

A
Using Spark

Spark is an instant messaging client developed by the developers of Openfire. It is open source, so it can be optimized and tailored for any organization. It is cross-platform, so your users can run it on any of the three popular platforms—Windows, Linux, and Mac. This is very important if you are supporting a large-scale deployment. And you get all of this for free, because Spark is offered as a free download. But Spark's Openfire connection isn't its only shining point—it's as good an IM client as any. In this Appendix, we'll look at some of the most important features of Spark, which, when combined with Openfire make it Spark the easiest alternative to insecure public IM networks.

For you, as the Openfire administrator, there are various aspects of Spark you can control from the Openfire admin interface (we've already covered this in an earlier chapter). You can control the distribution of a particular Spark client across the three supported platforms. You can set up download locations, send update notifications, and post installation, usage, or other Spark-related notes for your users. You can also add a list of bookmarks, which will be available to all Spark users automatically. The admin interface also lets you control several features of Spark such as allowing broadcasting, and changing avatars.

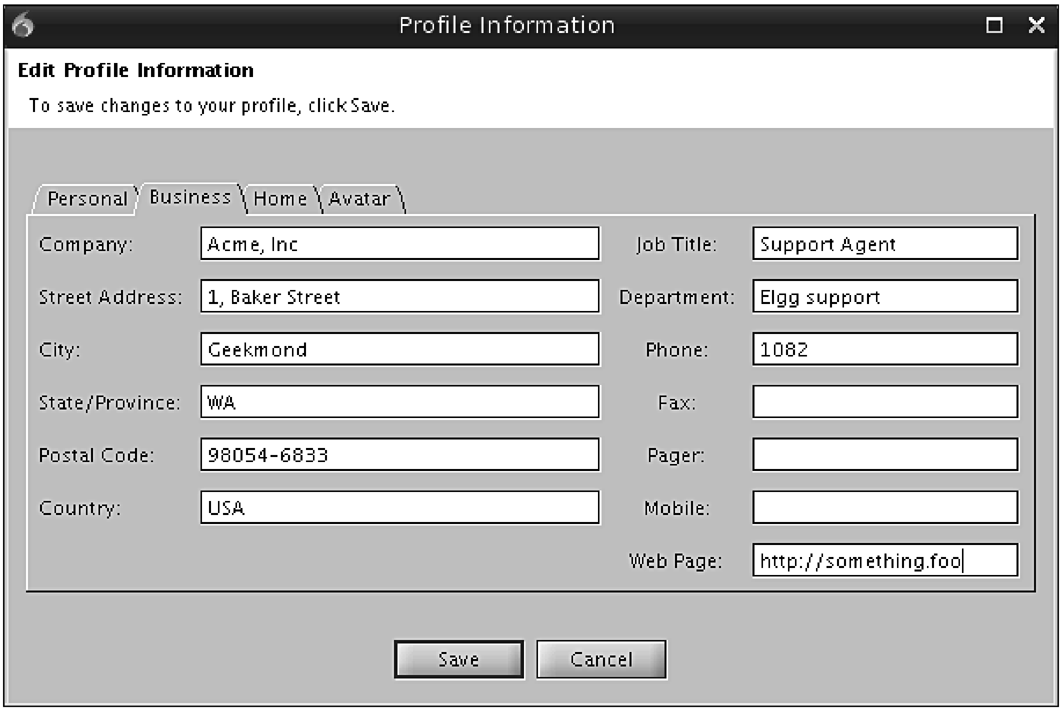
Getting Started

A my network admin friends, asking users to switch IM clients is known to incite rebellion. Just because it makes "your job" easier, isn't reason enough for users to switch. Why should they switch clients just because it's developed by the same folks who developed the IM server?

But, unlike many clients, Spark sports several features that'll make using it enjoyable for you and your users alike. And it all begins with logging in. Depending on how you have configured users on the Openfire server, your users will be able to use Spark's Single Sign On feature (SSO) to sign onto the IM along with other network services.

If you don't use SSO on your network, all that the users really need are their usernames and passwords. Or they can create their own from Spark by clicking on the **Accounts** button and filling in the login details. Your users don't have to bother fiddling around with server settings, and Spark will automatically discover the host and the port it needs to establish a secure connection with the Openfire server. If they need to change their password after logging in, which they should periodically, they can do so from Spark's preferences menu (**Spark** | **Preferences** | **General Chat Settings**).

The first thing users should be encouraged to tinker with is their profile. Here, irrespective of which method you are using to add users to the IM server (direct or through a directory server), Spark offers a wide range of profile settings that users can complete, such as their business and personal contact details, and choosing an avatar. The best part is that users can control and make changes to their own profiles.



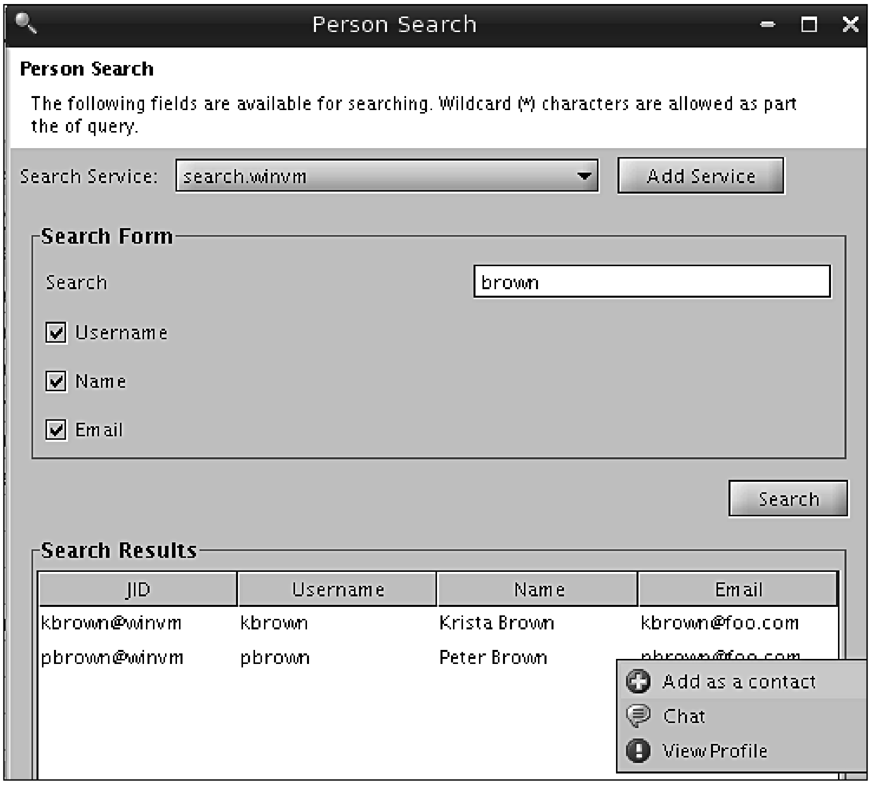
Edit Profile Information

To save changes to your profile, click Save.

Company:	Acme, Inc	Job Title:	Support Agent
Street Address:	1, Baker Street	Department:	Elgg support
City:	Geekmond	Phone:	1082
State/Province:	WA	Fax:	
Postal Code:	98054-6833	Pager:	
Country:	USA	Mobile:	
		Web Page:	http://something.foo

For an administrator, allowing users to control their own profile is the best example of the delegation powers of Spark, and helps ease your workload.

At the bottom of the main Spark interface, there's a small text bar running across the width of the client. Using this text bar, you can search for users on the system—just type in a username, email address, or a part of his or her name, and press *Enter*. This will launch the **Person Search** window. You can use this window to continue searching for the user until you've found them.


Person Search
Person Search
The following fields are available for searching. Wildcard (*) characters are allowed as part the of query.
Search Service: search.winvm
Add Service
Search Form
Search
brown
Username
Name
Email
Search
Search Results
Add as a contact
Chat
View Profile

JID	Username	Name	Email
kbrown@winvm	kbrown	Krista Brown	kbrown@foo.com
pbrown@winvm	pbrown	Peter Brown	pbrown@foo.com

On finding the user, you can right-click on the search results entry to:

- Add the user as a contact
- Initiate a chat with him
- View their profile

The other way to add users is by using the **Add a Contact** button in the Spark interface, or going to **Contacts** | **Add Contact**. From this interface, you can add the username of the user you wish to add, and optionally his nickname, by which he'll be listed in your roster and in the group in which you want to categorize him. By default, Spark has only one group (in addition to the ones created by you when adding users to Openfire) called **Friends**. You can add more groups by heading over to **Contacts** | **Add Contact Group**.

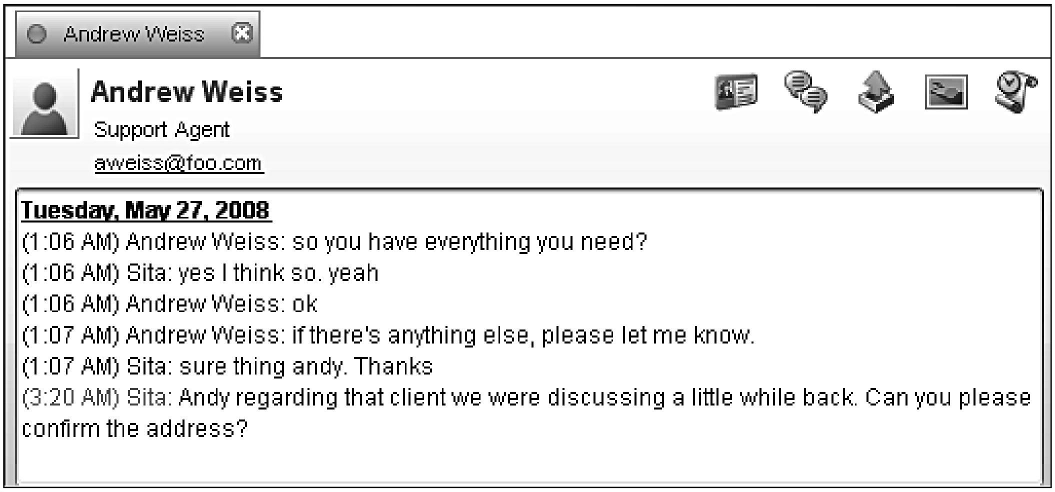
Chatter Away

In addition to being an easy-to-get-started IM client, Spark packs comprehensive tools and features that make it very useful in a corporate environment.

IM is infamous for its loosely bound lingo. But in a professional setup, users are a lot more cautious when it comes to wording their messages. Although Spark won't construct sentences for you, it'll do the next best thing and make sure that you type them in correctly. Spark can check messages and mark errors as you type them, or it can also be used to spell-check the whole message once you have finished typing.

This feature doesn't offer much to an administrator, unless he's messaging himself, but it does result in grammatically clean logs, which makes browsing through them easy.

Tracking conversations is one of the best features of Spark. When a new conversation is initiated between two users, Spark displays the last few lines of their previous communication. This is useful in setting context, especially if the new communication builds upon the previous one.



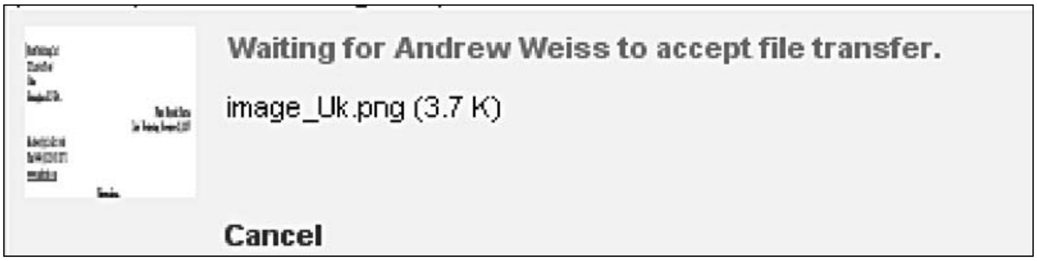
Tuesday, May 27, 2008
(1:06 AM) Andrew Weiss: so you have everything you need?
(1:06 AM) Sita: yes I think so. yeah
(1:06 AM) Andrew Weiss: ok
(1:07 AM) Andrew Weiss: if there's anything else, please let me know.
(1:07 AM) Sita: sure thing andy. Thanks
(3:20 AM) Sita: Andy regarding that client we were discussing a little while back. Can you please confirm the address?

Additionally, users also have access to their complete chat history. This is especially useful when accessed from the chat interface, as it gives a complete account of all conversations between the two users.

File transfer is another strong point of Spark. If you've allowed your users to transfer files, they'll be amazed by Spark's file transfer mechanism. Not only can Spark transfer multiple files of all types and sizes simultaneously, it can do so in a couple of ways.

You can use the traditional way of sending a file by locating it via the file select window. There's a button in the chat window to do this, or you can right-click on any user and choose the **Send a file** option from the context menu. The other Spark way of sending a file is to simply drag and drop the file into the chat window when you are conversing with the user. Before files can be transferred, they need to be accepted by the recipients, who are asked by Spark if they'd like to accept or reject the file transfer request.

Spark offers an exciting option for your users if they are collaborating on a project and need a quick and dirty way of exchanging information. For example, suppose that two accountants in different offices are going over the transfer of resources between the two offices in their spreadsheet program, and one of them needs to look into details of the other for a particular week. For this, they can use Spark's unique screen capture tool that captures and sends anything a user sees on his screen.


Waiting for Andrew Weiss to accept file transfer.

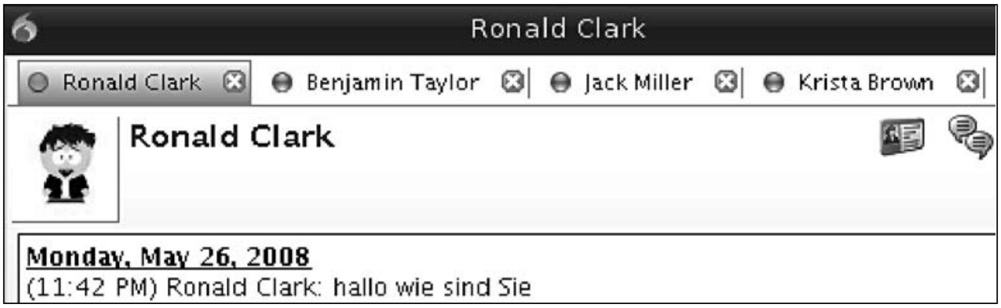
image_Uk.png (3.7 K)

When the screen capture button is clicked, the user's mouse pointer turns into a cross-hair, and the user can click, hold, and drag over the area he or she needs to capture. When the mouse button is released, Spark automatically converts that area into an image, names it, and then transfers it to the other user via a regular file transfer.

By default, Spark stores transferred files under the appropriate subdirectory, with in the directory in which it is installed. Under some operating systems, such as Windows, Spark stores the files under a directory, say **C: | Documents and Settings | Mayank Sharma | Spark | user | msharma@bhx.office-on-the.net | downloads**. For user convenience, Spark offers an option to **Open Containing Folder**, when a user has just received a file. Users can also head over to **Actions | View Downloads**, which opens the folder containing the downloads. But if you'd rather change the directoryc in which the file will be stored to, a place that is easily accessible, such as **My Downloads | From-Spark** or something similar, that's also possible.

To change the file transfer directory from within Spark, head over to **Spark | Preferences | File Transfer**. In this page, you can change your download directory as well as alter the default file transfer timeout. This is the amount of time after which Spark gives up trying to send the file across to the other user. By default, Spark tries for 30 minutes before giving up.

With Spark, your users can have multiple user chats in chatrooms. We'll cover this, as well as multiple one-to-one chats in detail in Appendix B. The multiple one-to-one chats are all listed as different tabs within a single chat window, which makes them easy to manage. Depending on the activity in a particular tab, the user is notified appropriately.

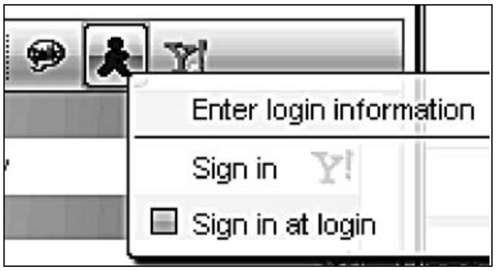


Monday, May 26, 2008
(11:42 PM) Ronald Clark: hallo wie sind Sie

To close a chat, a user can right-click on the **chat** tab, and choose from the following three options:

- Close this particular chat
- Close all chats except the selected one
- Close all stale chats—stale chats are chats which have been inactive for a particular duration of time

If your company policy permits, you can allow users to add external contacts from their public access IM networks. Spark is aware of the users added via a gateway, and handles them transparently without any issues. Depending on how the service has been configured, users have the ability to enter or change their login information. In addition to exporting the users, Spark also maintains them in the groups under which they have been arranged, on the public IM service.



Because gateways allow users to go beyond the limits of the controlled Openfire environment, enabling gateways is harrowing for an admin. But Openfire and Spark keep the external conversations under the purview of the internal policies, which provides a safeguard against abuse.

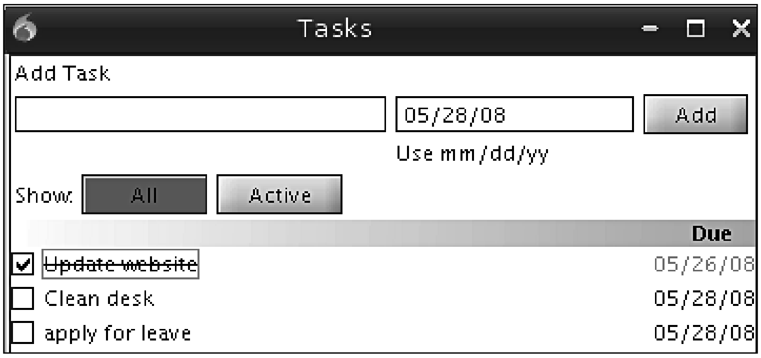
Remember the section where we configured Openfire to allow users to broadcast messages to other users? Depending on how you have configured the service (whether all users can broadcast or only some), users can broadcast a message by heading to **Actions | Broadcast Message**. This pops open a window that allows users to:

- Write the broadcast message
- Select the people/users to broadcast it to
- Decide whether to send the broadcast as a normal message, which opens a chat window, or an alert notification, which displays the message in a pop-up alert box

Of course, at the moment, Spark is the only client that works with Openfire's Fastpath feature. So if you are running an online helpdesk, like the one we set up in Chapter 11, your agents need to be running Spark in order to be accessible to online users.

One of the most powerful features that users have access to with Spark is the ability to search through chat transcripts based on a particular keyword. Remember the search text bar at the bottom used for searching people? On the far left of the text bar, you'll notice a small icon. Clicking on this reveals a drop-down menu, which lists two options—**Person Search** and **Chat Transcripts**. We've already looked at the person search feature. When the **Chat Transcript** option is selected, Spark searches through the chat logs looking for the entered text string.

Spark also doubles up as a personal note-taking application, and a basic task reminder. Both of these tools can be launched from within the Spark interface via their respective buttons. When taking notes, users need to click the **Save** button to save the note.

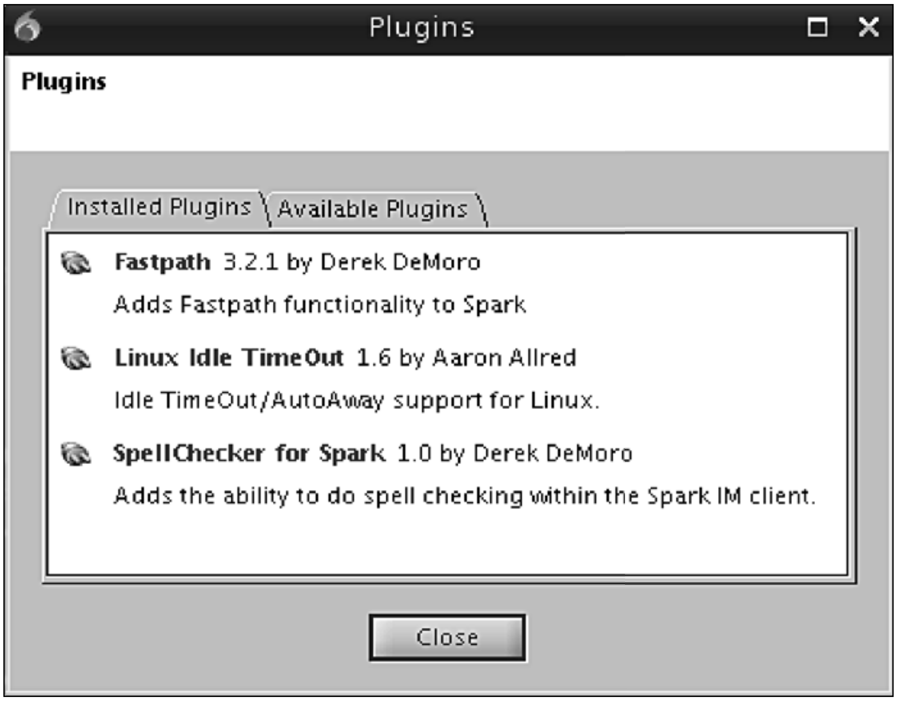


		Due
✓	~~Update website~~	05/26/08
☐	Clean desk	05/28/08
☐	apply for leave	05/28/08

To add a task, users have to give it a name that is descriptive, attach a planned date of completion to it, and click the **Add** button. When tasks are completed, you can click on the checkbox adjacent to them, which strikes off a particular task from the list of active tasks. The **All** tab lists all tasks, whether completed, or pending.

Spark Plugins

As if the native features weren't enough, Spark, similar to Openfire, can also be extended through the use of external plugins. You will find that some of the plugins are already installed on your client, depending on the operating system you are running it on, and the Openfire services you have enabled on the server.



Fastpath 3.2.1 by Derek DeMoro
Adds Fastpath functionality to Spark

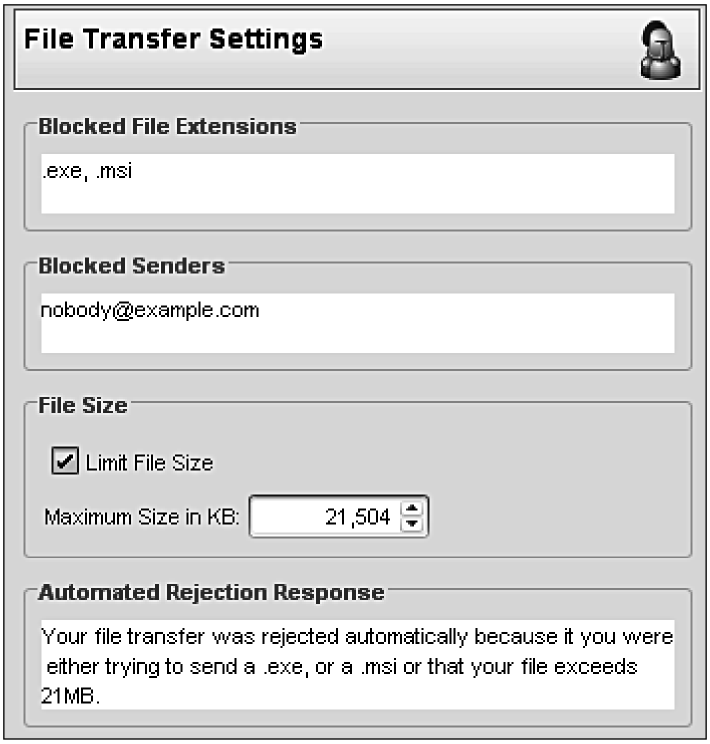
Linux Idle TimeOut 1.6 by Aaron Allred
Idle TimeOut/AutoAway support for Linux.

SpellChecker for Spark 1.0 by Derek DeMoro
Adds the ability to do spell checking within the Spark IM client.

The Spark Plugins interface is tucked away under **Spark | Plugins**. Listed under two tabs in this interface are installed plugins and installable plugins. If you are running Spark on Linux, you'll notice you have a plugin called **Linux Idle TimeOut** preinstalled. As the name suggests, this plugin is available for the Linux client only, and gives Spark its ability to detect and handle idle sessions. Also installed will be the **SpellChecker for Spark** plugin, thanks to which Spark gets its spell-check capabilities.

If you have Fastpath enabled on your Openfire server, the Spark client will also have the **Fastpath** plugin installed. This plugin adds a tab to the Spark interface to provide the user-centric Fastpath features we discussed at the end of the previous chapter. And if you have enabled Openfire to interact with your SIP telephony system, the **Phone Client** plugin is where the rubber hits the road. Using this plugin, you'll be able to make calls to other users from within Spark itself.

There are a couple of plugins that you can install in any Spark installation irrespective of the platform you are running it on. These are the **Transfer Guard** plugin, and the **Translator** plugin. Once installed, plugins can be controlled from under the Spark preferences menu accessible via **Spark** | **Preferences** | **Transfer Guard**.



.exe, .msi

nobody@example.com

[✓] Limit File Size
Maximum Size in KB: 21,504

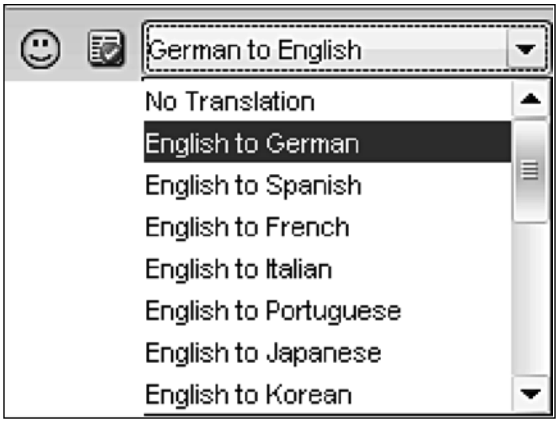
Your file transfer was rejected automatically because it you were either trying to send a .exe, or a .msi or that your file exceeds 21MB.

As the name suggests, with the **Transfer Guard** installed, users can block file transfers of:

- Files with a particular extension
- Files from particular senders
- Files exceeding a preset file size

The **Transfer Guard** plugin also lets you set an automated file rejection response.

The **Translator** plugin will be very useful when you interact with users who are non-native speakers of English. This plugin uses Google's translator service, and will translate your English messages into various other languages including German, Spanish, Italian, French, Korean, and Chinese.



The best thing about this plugin is that in your chat window, you'll still be able to see the messages in English, and it's only the recipient who sees the translated version. The bad thing is that if the other user doesn't use the plugin, or forgets to select the reverse translation method, you will see the response in a non-English language, as the plugin isn't designed to translate received messages.

Summary

This appendix offered a quick look at Spark, the official Openfire client. Developed by the same group of developers who worked on Openfire, Spark offers quite a few nifty features, tools, and plugins that make it as useful an IM client as any.

In this appendix, we've looked at Spark's wide range of features, from searching and adding users to the more interesting ones such as broadcasting messages, drag and drop file transfer, searching through chat transcripts, and so on. We've also looked at the pre-installed and available plugins that give Spark the ability to spell-check messages, control indiscriminate file transfers, and translate messages into a variety of languages while bouncing them between users.

B
Group Chat

From the user's point of view, one of the most important aspects of using an in-house Instant Messaging server is to have secured many-to-many discussions. In IM-speak, this text-version of a teleconference is known as a group chat. Not only does Openfire allow group chats, but it also offers comprehensive controls that make it malleable enough to host on-demand, secure, private group chats between senior management as well as always-on, casual, hang-out group chats for impromptu meetings around the virtual water cooler for all employees.

Group chats are isolated in virtual chat rooms. In this Appendix, we'll look at the various group chat configuration options at our disposal, while creating a couple of rooms.

Group Chat Settings

We'll start by looking at the group chat settings that are applicable to all groups. As an admin, you'll have to pay special attention to these settings as they'll apply to all group chat rooms. To set or modify any of the global settings, head over to **Group Chat | Group Chat Settings**.

- **Name of the service:** As you already know, Openfire offers its "features" as services. We've already discussed some of them, such as the broadcast service. First and foremost, you'll have to select a name for the group chat service. By default, Openfire uses **conference** as the name.
- **History Settings:** Remember the **Recent History** feature of the Spark client we covered in Appendix A? In quite the same fashion, Openfire displays the last few lines of an ongoing conversation in a room to new users, when they join in. As the administrator, it's up to you to decide on the number of messages that should be displayed.

You can choose between three history display options to:

- Show the entire history
- Show a specific number of messages (the default is 25)
- Don't display old messages at all

- **Global Administrators:** You can specify a list of users with special powers. These global administrators will be able to join any room, and Openfire will give them the same powers as those of the room owner. The users added here should be selected very judiciously as they'll also be allowed to enter private and confidential rooms. Depending on the situation, it might be best to avoid adding any users to the list of global administrators.
- **Room Creation Permissions:** By default, all users have the power to create rooms. Some organizations might prefer to keep this default setting. But you can also choose to restrict the ability to create rooms to a list of specific users. Limiting this ability to team leaders or department heads is a good idea.
- **Handling Idle users:** Openfire, by default, will allow users to linger in a room for as long as they wish. You should go with this setting if you plan to have lounge-type rooms, or rooms that act as virtual bulletin boards. If on the other hand, you have a lot of rooms that limit the number of members (how to do this is covered in the next section), you might choose to kick out users after they've been idle for a particular period of time, say 30 minutes.
- **Conversation Logging:** You can choose to log the conversation of a particular room when creating the room—we'll cover how to do this in the next section. In the global settings, you just have to select a flush interval (in seconds), and batch size (the number of messages), based on which conversations are logged. It's alright to go with the default settings, unless you've tweaked some other Openfire parameters, say cache size, and so on.

Although for most cases it's quite alright to leave these settings with their default values, you might need to alter them—especially the room creation permission settings and history settings. As with all admin settings, you are treading dangerous waters here, when you take into account settings such as the global room administrators. Make this decision with due diligence.

Designing a Room

Openfire offers a good number of options when creating a group chat room. We could have gone over these choices when creating a room, but a better way is to list of all the choices on offer, and then see how they can be used to customize various groups.

When creating making a room, you'll have to take various decisions. Broadly, they can be classified under three categories:

1. General room characteristics that help you define a room and its members.
2. The user permissions that divides members of a room into different categories with different powers.
3. Occupant behavior, through which you control the actions of users once they are inside a room. Let's look at each of these in detail.

General Room Characteristics

The settings under this group help us to give shape to the room.

- **How many members do you want in the group chat:** When creating a room, you can decide and put a cap on the number of users allowed in a group. Openfire offers a range of 10-50 users, in increments of 10. Or, you can remove the cap and allow an unlimited number of users.
- **Presence notification:** In addition to the messages flowing between users, Openfire will also display presence notification for moderators, participants, and visitors. Depending on the traffic of the room, and how much of a distraction it will be, you can limit the presence notification to any of the three types of users, or choose to provide no notifications at all.
- **Password protect the room:** Openfire allows you to lock a private room with a password, for added privacy.
- **Members' real identity:** Users can change their names when entering rooms. Depending on the type of room you are creating, you can make them show their real Jabber IDs to moderators, or to everyone else.
- **Force members to log in with a registered nickname:** If you are very concerned about real identities, in certain rooms you may not want members to login with any names they like, but want them to use their registered nicknames. Openfire gives you the flexibility to make users use their registered nicknames when they log in to a room.
- **Listed in directory:** Because Openfire can create private rooms, you can choose whether you want a room to be listed in the directory of rooms available, or not.
- **Should the room be moderated:** A moderated room curtails the freedom of users in the room. If a room is moderated, users need to have appropriate permissions (see below) to post messages in a view-only room. If you run a bulletin-board type of a room, only some users will have the ability to "announce" messages, while the others can only read the messages.

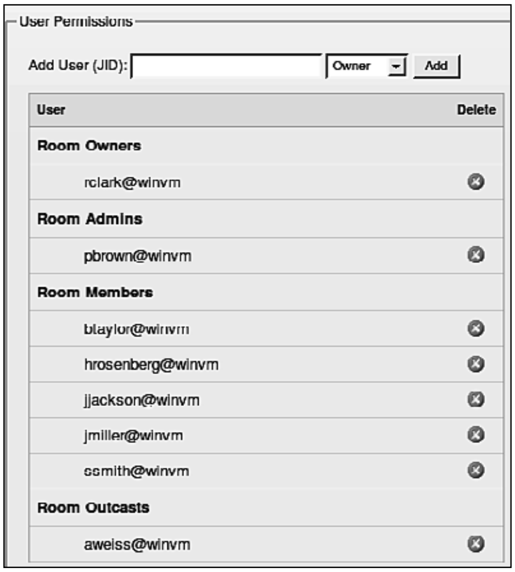
- **Restricted access:** This is similar to a moderated room, but goes one step further, and doesn't allow non-members to even enter the room. This is another way of limiting access to a room, without the necessity of a password.
- **Log room conversations:** It isn't necessary for all rooms under Openfire to have their conversations logged. If the room is being used for work-related discussions, you can enable logging of the conversation for the room, while leaving water-cooler rooms unlogged.

User Permissions

Users under Openfire are separated into groups, and all users are treated the same. But when it comes to group chats, the categorization of users helps to maintain decorum and order, and helps to keep private rooms private. Openfire categorizes users under four categories.

Owners: Users listed under this category have full control over the room for which they have been defined as the owner. They can tweak the room's configuration, have the power to grant ownership and bestow administrative privileges to other users of the room, and can even destroy the room. Global group room administrators are treated as room owners, and share the same permissions for all rooms!

Admins: The next level of users in the group chat hierarchy of users is the admins. These users are added by the room owners, and can grant membership and moderator privileges to other users. These users also have the power to ban members from the room.



User	Delete
Room Owners	
rclark@winvm	⊗
Room Admins	
pbrown@winvm	⊗
Room Members	
btaylor@winvm	⊗
hrosenberg@winvm	⊗
jjackson@winvm	⊗
jmiller@winvm	⊗
ssmith@winvm	⊗
Room Outcasts	
aweiss@winvm	⊗

Members: A member is the most common type of a group chat user. Members lack any power over other others, but can manipulate a room as described in the "Occupant Behavior" section , which comes next. If a room is marked "members-only", then only the users listed as members are allowed access to that room.

Outcasts: An outcast is a user who has been banned from the room by either the room owner, or the admin. If logged in, these users are automatically ejected from the room and all their subsequent attempts to log in to the room are denied.

All rooms have their own set of users in these categories. Also, users in one category for one room can be in another category for another room. For example, user **kbrown** can be owner of the **sales** room, but only a member of the **boardroom** room.

Occupant Behaviour

The above set of settings will help you decide on the general characteristics of a room and narrow down or broaden the list of members, or room occupants. But once inside, what powers do they have?

In addition to having a conversation, you can break down a user's activity inside a room into three different activities.

1. **Can occupants invite others?** Unless it's a restricted access room, letting users invite others is a good idea.
2. **Can occupants change the subject of discussion of the room?** This should be used as per the needs of the room. The subject of the discussion is very important if you've disabled the **History Settings** for the chat rooms. If there are no owners and admins present, allowing users to change the subject of a discussion is a good idea, but can be easily abused.
3. **Can occupants change their nicknames?** Depending on the type and mood of the room you might allow users to change their nicknames.

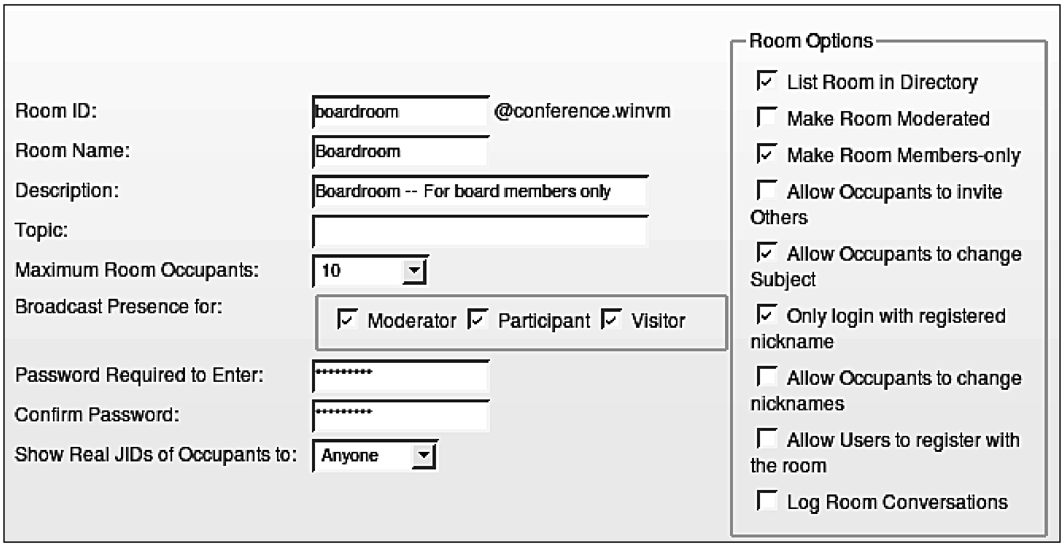
Again, similar to the previous settings, you have to set the behavior of the users for every room. Although these might appear to be minor decisions, don't take them lightly. As I've explained with each activity, it's imperative that you put the whole room in perspective before making these decisions.

Putting the Checklist to Test

As I told you at the start of this Appendix, creating a room is fairly complex business, considering the choices you have to make. Now that we know what we're dealing with, let's see how we can use the various settings to create a few rooms with varying degrees of user participation and control.

Room 1: A Locked down "boardroom"

The boardroom is an example of a room that pays special attention to exclusivity and takes strong measures to restrict the members who have access to the group. Here are the room settings for the boardroom.



Room ID:	boardroom @conference.winvm
Room Name:	Boardroom
Description:	Boardroom -- For board members only
Topic:	
Maximum Room Occupants:	10
Broadcast Presence for:	[✓] Moderator [✓] Participant [✓] Visitor
Password Required to Enter:	**********
Confirm Password:	**********
Show Real JIDs of Occupants to:	Anyone

Room Options

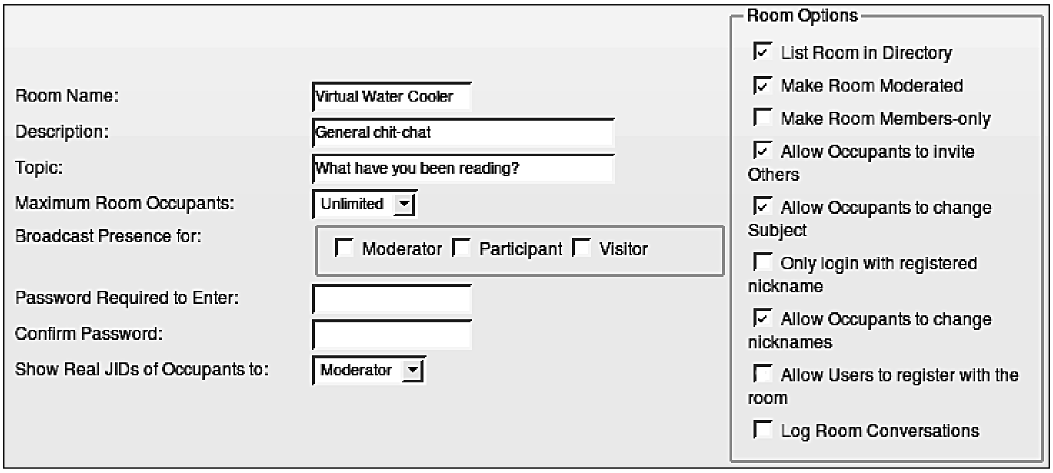
- [✓] List Room in Directory
- [ ] Make Room Moderated
- [✓] Make Room Members-only
- [ ] Allow Occupants to invite Others
- [✓] Allow Occupants to change Subject
- [✓] Only login with registered nickname
- [ ] Allow Occupants to change nicknames
- [ ] Allow Users to register with the room
- [ ] Log Room Conversations

Every room has a **Room ID**, a **Room Name**, and a brief **Description**. This helps users understand what a room is for. We want the boardroom to be a private room for board members only and have therefore locked it with a password. We've also made the room "members-only", so users have to be listed under one of the three allowed branches (owner, admin, member) of user permission to be allowed entry into the room.

Because this is a low volume room, due to its limited number of members, we've also allowed the presence of all members to be broadcast, to keep everyone updated about the movement of the other board members. To make sure who's who, members are only allowed to log in with their registered nicknames and everyone can see the real Jabber ID of every member in the room. There is no specific topic for this room because the board members can discuss anything, so every member has been given the power to change the topic. Despite its exclusivity, we've allowed it to be listed in the directory of available rooms, lest our dear board members forget the name of the room they want to log into. Finally, we've also disabled logging of the conversation in this room.

Room 2: A Free-for-all "Lounge"

The exact opposite of the boardroom, is the lounge, which doesn't restrict access, and gives members an almost free run through the room. Here are its settings:



Room Name:	Virtual Water Cooler
Description:	General chit-chat
Topic:	What have you been reading?
Maximum Room Occupants:	Unlimited
Broadcast Presence for:	☐ Moderator ☐ Participant ☐ Visitor
Password Required to Enter:	
Confirm Password:	
Show Real JIDs of Occupants to:	Moderator

Room Options

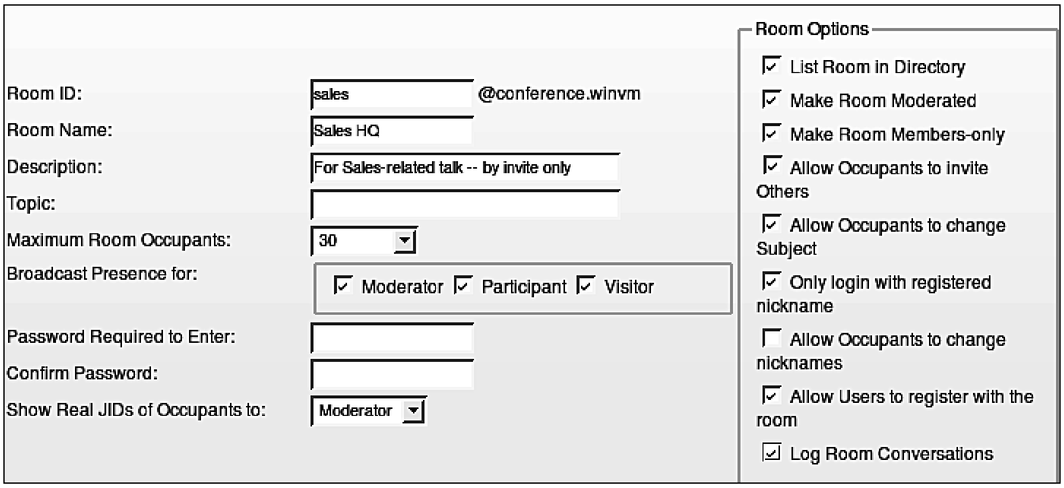
- ☑ List Room in Directory
- ☑ Make Room Moderated
- ☐ Make Room Members-only
- ☑ Allow Occupants to invite Others
- ☑ Allow Occupants to change Subject
- ☐ Only login with registered nickname
- ☑ Allow Occupants to change nicknames
- ☐ Allow Users to register with the room
- ☐ Log Room Conversations

The first thing you'll notice is that this room allows an **Unlimited** number of participants. Due to its size and the amount of chatter it'll generate, we'll spare everyone from seeing the presence notifications of members joining and leaving the room. Because this is an informal, but moderated room, the room members are allowed to change their nicknames, but moderators can find out the real Jabber IDs of users, which will help them ban users who aren't decent and courteous to other members.

A room such as this has to be listed in the public directory of available rooms, and members have the power to invite others to join the fun. It does start of with a topic, but everyone has the right to change it. There's no point clogging the logs with all the off-topic chatter, so we've left out that option.

Room 3: A Restricted "Sales HQ" Room for Sales-Related Discussions

Somewhere between the air-tight boardroom and the chaotic lounge is the "sales HQ". It's dedicated to sales-related talk and restricts access, but allows a little latitude. Here's how it's set up:



Room ID:	sales @conference.winvm
Room Name:	Sales HQ
Description:	For Sales-related talk -- by invite only
Topic:	
Maximum Room Occupants:	30
Broadcast Presence for:	✓ Moderator ✓ Participant ✓ Visitor
Password Required to Enter:	
Confirm Password:	
Show Real JIDs of Occupants to:	Moderator

Room Options

- ✓ List Room in Directory
- ✓ Make Room Moderated
- ✓ Make Room Members-only
- ✓ Allow Occupants to invite Others
- ✓ Allow Occupants to change Subject
- ✓ Only login with registered nickname
- ☐ Allow Occupants to change nicknames
- ✓ Allow Users to register with the room
- ✓ Log Room Conversations

Although the room is listed in the directory, it is limited to members-only. But there's no password, and members are allowed to invite other members. This is done assuming that, from time to time, the group may need to invite other users for brief discussions and clarifications. For example, our sales people might want to invite people from other departments such as accounts to clarify details regarding a particular transaction.

The group is a general purpose discussion group, and begins without a topic, allowing members to change the subject as they see fit. Most importantly, all activities and discussions in this room are logged.

Using Spark for Conferencing

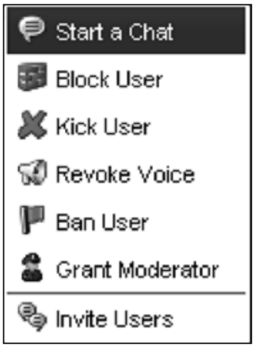
Now that you have a fair idea about the power and flexibility of group chats under Openfire, it's time to hop across the fence and quickly look at how it looks to your users.

After logging into their account from Spark, users need to head over to the **Conferences** tab to select the conference service to join. When a user double-clicks on the name of the conference service to which he or she wishes to subscribe, they'll be shown a directory of all of the available rooms.

Rooms that have been set up to be hidden from the public directory of rooms will not be listed here.

To join a room, a user needs to select the room from the list, and click the **Join Selected Room** button. Depending on how a room has been restricted, Openfire will prompt the user to enter a password and/or check whether the user is included in the list of members that are allowed access to this room.

Once inside, the chat interface is divided into two panes—the left pane is where the conversation takes place, and the right pane lists all of the logged-in members. Depending on how the room has been set up, and the user's privileges, the user can right-click on his or her name to change their nickname, as well as to invite others. Right-clicking on the left-pane will allow the user to change the subject of the room, if that's permitted.



If the user is the room owner, or an admin, they can also click on the name of the other members in the room and choose to kick them out of the room, block them from the room, and ban or un-ban them. Room owners can also bestow admin privileges to other members.

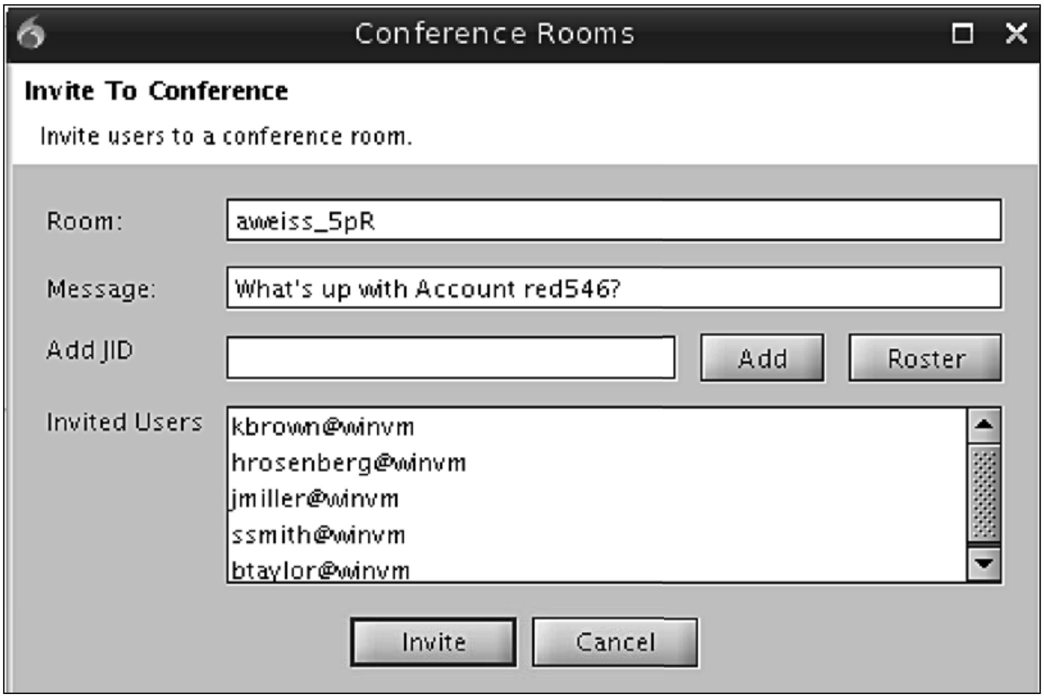
Impromptu Multi-User Discussions

What we have been looking at up to now are permanent rooms. Openfire and Spark also offer impromptu group chat rooms that can be created on-the-fly, and cease to exist when there are no members in the room.

To create such a room, a user needs to select multiple users (more than one) from his or her roster, right-click on the section, and select the **Start a conference** option. This opens a window that will help him compose an invite to other users. From the window, they can:

1. **Name the room:** This is automatically composed by Openfire/Spark, and is used internally to uniquely identify every conference.
2. **Write an invite message:** A simple message that will be sent to all of the selected users inviting them to join the conference. Users can change the default message body to add a little information about the topic of the conference.
3. **Invite more users:** If a user wants to add more users in addition to their original selection, they can do so by either adding user's JIDs by hand, or by selecting users from their roster.

Finally the window displays a list of all invited users. As soon as the creator of the room hits the **Send** button, all invited members get a message inviting them to a multi-user chat. Invited users can then choose to accept or decline the invitation.


Conference Rooms

Invite To Conference

Invite users to a conference room.

Room: aweiss_5pR

Message: What's up with Account red546?

Add JID

Invited Users
kbrown@winvm
hrosenberg@winvm
jmiller@winvm
ssmith@winvm
btaylor@winvm

Impromptu sessions are useful when users want to have a brief private chat with some other users. As an admin, these temporary discussions save you the trouble of having to go through creating a room every time a user needs to talk to a bunch of other users at the same time.

Summary

If deployed properly, group chats can offer a lot of advantages over one-to-one chats. And Openfire has a comprehensive group chat mechanism, which we have explored in this appendix.

We have broken down Openfire's group chat options into three broad groups—general room characteristics, user permissions, and occupant behavior—and looked at each of these in detail, and tried to understand their individual functions. We've then put the functions to the test by creating three rooms with different purposes and requirements.

In the last section of the appendix, we hopped over the fence and looked at how a multiuser chat works from the vantage point of our users. Finally, we've seen another display of Openfire's flexibility through impromptu multiuser discussions. In addition to the comprehensive group chat system, Openfire also allows for a quick and dirty mechanism for having many-to-many discussions.

C
Prepare for Roll Out

There is absolutely no doubt about the advantages of Instant Messaging, especially within the regulated, diverse, and fast-paced environment of an enterprise. If you don't already have an in-house IM server, you'll be surprised to see how many employees are relying on the public IM providers for their instant communication needs. But these public services expose the user, and ultimately your whole enterprise, to the ever-growing threat of IM viruses and IM spam. And depending on the nature of your firm's business and your country of operation, the lack of an IM communication trail, an inherent 'feature' of public IM, this might also put you on the wrong side of the law! On the other hand, if you think just deploying an IM server makes you any safer, think again. Thanks to the nature of the risks involved, deploying an IM server requires a lot of due diligence on the part of the administrator.

Many admins have the wrong notion about what makes good server software, and tend to pay more attention to things like a server's quick setup time, and graphical menus. Every aspect of software that powers critical services such as IM has to be handled with care, from selection to deployment. It doesn't take long for an ill-planned and hurriedly-executed deployment to become a distraction instead of enhancing productivity.

This is true, and even more so, in the case of IM. Your IM server has to enhance productivity by being pervasive, but with suitable restraints, and adequate flexibility. For example, you want a system that provides everyone with access to IM, caters to a person on tour, and is flexible enough to allow temporary suspension of a person's account to prevent misuse they are on sick leave, or on vacation. You'd also want your system to discourage users from using public IM services (except, possibly, during certain periods such as lunch breaks, or on the weekends), and yet be flexible enough to provide the service to individual users on an as-needed basis.

As you can see, there's a lot of thought involved here. However, you can make what might seem like a daunting task a walk-in-the-park. All you need is a set of documents for yourself and for your users. Let's look at these in detail.

Identify the Required Objects and their Properties

The first thing you need to do, before deploying an IM server, is to make a list of all of the IM services available, and consider how you want to distribute them between users. This can be an exhaustive task. I'll list some of the more essential services here:

1. **Identify global services.** These are services available to all users. For sure, you'd want everyone to be able to exchange IMs with each other. But what are the add-ons and conveniences you'd want them to be able to use? Some of the things you'd probably want everyone to do would include searching for other users on the network, editing their profile, and changing their password.
2. **Identify restricted services.** These are services that will only be available to a select few, in addition to the global services. The ability to broadcast is one such service. You also wouldn't want every user to be able to connect to external IM services via the gateway extension. Neither would you want everyone to have the right to create a persistent chat room.
3. **Identify the override method.** There will be times when you'll have to give people access to restricted services. You can prepare for such a situation by identifying the override authority for each restricted service. There will be some services that a user could request directly from you, while others would require approval from a user's supervisor. You'll also need to take into account a user's role when deciding their override methods. For example, if you are a newspaper, a journalist can directly request a gateway access, while an accountant's request might require a supervisor's approval.
4. **Identify key users.** Typically, you'd want to delegate or share your admin powers with other admins as well, unless you are a one-man team. Then, there are some services that have their own hierarchy of users. One such service is multi-user chat rooms. You'll need to determine which users own which rooms, and which users have administrator privileges for those rooms. Additionally, the IM system also has the ability to notify certain users of various events such as policy violations. These will also have to be identified.

Once you have the list of services and their distribution across the various categories of users in your organization, you might want to run this plan past the management, and if possible get input from the various department heads as well.

Plan the Deployment Process

In addition to identifying the various components of an IM system, it's advisable to have a rough outline of the deployment process. This will help you estimate the time required for the process, as well as serve as a checklist while you are busy deploying the server. Here's an overview of a typical IM service deployment process:

1. **Plan the infrastructure and determine the server locations**. If you have a large organization, you'll need to introduce redundancy and setup servers at multiple locations to distribute loads. Also, determine if you'll be running the IM server on a dedicated server, or on a shared one. If you want to deploy multiple services on one server, you should understand that a heavily-utilized chat service requires significant server resources, in terms of memory, CPU, and network throughput, to meet customer response-time expectations.
2. **Name your chat servers** in line with the naming standards already in use for other servers on the network or in your organization.
3. **Install the server and hook it up with the external services.** This would include:
    - Hooking up with an external database for storing IM-related data
    - Connecting with a directory server for adding users
4. **Pre-populate rosters.**
5. **Create the required Domain Name System** (DNS) resource records (typically, a host record) for the chat server.
6. **Configure Network Address Translation (NAT) and Firewalls.** Verify that the proper service records have been entered in all internal DNS servers, and in case of server-to-server communication, in external DNS servers as well. Make sure that all required ports are open and forwarding traffic to the correct addresses.
7. **Make the IM client software available to users.** Upload the clients on the server, and send a pre-drafted notice about the new IM service, with (a link to) adequate end-user documentation to all users.

Instant Messaging – Best Practices

Once you have successfully planned and deployed the IM server, you have to make sure it stays that way. This can be a tricky proposition as there's very little you can do here, other than lay down rules and/or point out user responsibilities.

- **Treat IM as email**: Many organizations say IM is the new email. Considering the similarities between the two electronic communication methods, it'd be best if you treat IM as email. If you have an email policy already in place, append it and apply it to IM as well.

  If you wish to have separate policies for email and IM, make sure you consult with the appropriate internal legal, HR, IT, and executive management members and groups to construct a policy appropriate for your organization's needs and requirements.
- **Define prohibited use**: No matter what sort of organization yours is, having a "you can't do this" list of things is always encouraged. Make sure this list clearly points out what constitutes disruptive or offensive messages, and the types and nature of files that can't be transmitted between users. Also, point out actions that will be taken in case of a violation of the policy.
- **Define personal use**: You should expect and allow for a reasonable amount of personal exchanges. But define limits and identify how non-work related or off-topic conversations are to be handled.
- **Define monitoring terms**: Depending on the nature of your organization, it might be imperative for you to log all IM conversation as per applicable trade regulations. In any case, if you log messages, make sure your users are aware that the organization is doing so, and may monitor messages without prior notice.

User responsibilities

Usage policies can be long and a little too detailed for the average user to bother reading. So in addition to the usage policy, you can also prepare bulleted action items that are easier for users to follow. Some of the most common ones are listed here:

- Ensure that your IM account password meets the organization's recommendations for strong passwords. Make sure you change your passwords frequently.
- Look out for updates to the IM program and install them as soon as they are available.
- Don't allow your IM program to "remember" your password or automatically sign in to your account.

- Don't automatically accept incoming messages from users who are not on your contact list. Users should exchange IM usernames or nicknames via phone or email before sending IM messages.
- Don't accept file transfers from users outside the network. Instead, ask external users to send the attachment via email, which will be scanned at the mail server, in addition to the anti-virus application on your computer.
- Set Status messages. Use your client's ability to set status messages. Either use an existing one, or create your own to indicate when you are out for lunch, or not on your desk, or busy working and do not want to be interrupted.

Audit Servers and Logs

While your users make sure they use the IM service as it's meant to be used, it's your job as the administrator to make sure everything's hunky dory on the back end. Here's a brief checklist.

- **Keep an eye on the database**: There are various tools available that'll help you monitor the state of the database server you are running for your IM server. If the program doesn't allow for automated periodic checks, do one yourself every two weeks, or more frequently, if you have lots of concurrent IM users.
- **Directory Server**: Take special care of the directory server if you are using one, as it'll also be shared by other network services, and if it goes down, your users won't be able to log in to their IM accounts.
- **The physical server powering the system**: Depending on the operating system you are running on the server, configure and set up automated audits of the key areas of the system, and periodically monitor the logs. Also, set up regular automated backups of key areas as well as periodic system-wide backups.
- **Monitor the IM server logs**: Make provision to be notified when something is written to the error logs.

Summary

Your role as an admin in charge of rolling out an IM server doesn't begin at the deployment, nor does it end after the server has been deployed. You have to judiciously select the correct IM server to make sure it meets all your needs. Of course you'd need to have a shopping list of your IM needs before you get down to selecting an IM server!

In this final Appendix, we've broadly looked at what goes on behind the scenes in selecting an IM server, planning the deployment, and making sure that the server maintains a 100% up-time.

We've looked at the various decisions you, as the admin, will have to make, and the need to brainstorm with the management and other departments to maximize productivity and add flexibility to the system.

Index

A

Active Directory
  users, adding via 69-71
agent, functionalities
  canned responses 180
  chats, transferring 178
  co-browsing, initiating 179
  current chats 177
  other agents, inviting 179
  queue activity 178
  user, history 180
  user, notes 180
archives, searching for 122-124
archiving, settings 120-122

B

broadcasting messages 81
broadcast plugin
  setting up 82, 83
  uses 82
  using 83, 84

C

caches
  distributed caches 137
  replicated caches 137
chat
  archives 174, 175
  monitoring 173
  sessions, statistics 175, 177
chat usage statistics 175
cluster
  caches 137
  optimizing, tips 138
  setting up 135-137
config file, editing 71-74
connection manager
  configuring 140, 141
  deploying 139
  Openfire server, configuring 140
  running 141, 142
content, filtering 114-116
content, monitoring 114-116

D

database support
  checking 131-135
  database errors, troubleshooting 130, 131
  enabling 127, 128
  MySQL, setting up 128-130
  PostgreSQL, setting up 130
deployment process, IM service 207
directory service
  Active Directory 69
  OpenLDAP 63
DSL router 147
DynDNS
  need for 144, 145

E

EIM 8
Enterprise Instant Messaging. *See* **EIM**
exporting, users 90

F

Fastpath
  installing 153-155
features, Openfire 12-14

G

gateway, Openfire
access, controlling 100, 101
gateway plugin 100, 101
plugins 98-100
registrations 106-109
using 106-108
gateway plugin. *See also* **gateway, Openfire**
about 100, 101
Google talk, enabling 105, 106
IRC, enabling for IT and Devs 103, 105
yahoo! messenger, enabling 101-103
group chat, settings 193
conversation, logging 194
global administrators 194
history, settings 193
idle users, handling 194
room, creating permissions 194
service name 193
group chat room
about 194, 195
designing 194
impromptu, creating 201, 202
room, characteristics 195, 196
Spark, using for conferencing 200, 201
user, activities 197
users, categories 196, 197
groups
deleting 55, 56
editing 55, 56

H

helpdesk, configuring
canned responses 163, 164
chat transcript 165, 166
offline settings 164, 165
helpdesk, creating
global settings 157
workgroup settings 158
helpdesk, deploying 153-155
helpdesk, designing
contact form, designing 167, 168
HTML, uploading 170-172
images, reviewing 168, 170
text, reviewing 168, 170

I

iBall plugin 118
IM
about 8
advantages, over email and telephone 9, 10
email alerts, getting 79-81
guidelines 207, 208
logs, auditing 209
servers, auditing 209
user responsibilities 208
IM clients, configuring
about 28
Kopete 30
MirandaIM 28, 29
Pidgin 31, 32
importing, users 90
IM service
about 206
deployment process 207
installing, Openfire
in Linux/Unix 20
in Windows 21
Instant Messaging. *See* **IM**

L

light-weight third-party archiving apps
about 124
Open Archive, using 124-126

M

messages, auditing
message audit policy, setting up 117, 118
raw audit logs, filtering 120
raw audit logs, reading 118, 119
Monitoring Service plugin, archiving settings 121

O

online helpdesk
about 153
deploying 153-155
Open Archive
about 124

Openfire
administration console 24
as windows service 21, 22
broadcast plugin 81
cluster, setting up 135-137
config file, editing 71-74
connecting to, Asterisk setup used 109, 112
content, filtering 114-116
content, monitoring 114-116
database support, enabling 127, 128
external database, using 127, 128
Fastpath plugin, installing 153-155
features 13, 14
gateway 98-100
group chat, settings 193, 194
group chat room 194
groups, deleting 55, 56
groups, editing 55, 56
helpdesk, configuring 162, 163
helpdesk, creating 156
helpdesk, deploying 153-155
helpdesk, designing 167
impromptu group chat room, creating 201, 202
installing, from versions 19, 20
installing, in Windows 21
light-weight third-party archiving apps 124
messages, auditing 116
need for 98
plugins, installing 48
private data storage 89, 90
room, creating 197
room, designing 194
rosters, pre-populating 57-60
running, in Linux/Unix 20
running, in Windows 21
server 35
server-to-server communication 145
service, controlling 22
setup 23
Spark 183
system, preparing 17, 18
two branches, linking 144
user clients, managing 84-89
users, adding 51, 52
users, deleting 52, 53
users, editing 52, 53
users, organizing into groups 54, 55
users, suspending 53, 54
users adding, Active Directory used 69-71
users adding, OpenLDAP used 64-69
VoIP, connecting to 109, 112
Webchat plugin, installing 153-155
Openfire, scaling with cluster 135
Openfire, scaling with connection manager 139
Openfire, settings
about 22
administration console 24
database settings 23
network settings 24
server settings 24
Openfire installers 18, 19
Openfire servers
about 146, 147
accessing, DynDNS used 144, 145
DSL router 147
users, linking 144
users linking, DynDNS used 144
OpenLDAP
installing 64
users, adding via 64-69

P

plugin, using
askstatus 94
recvstatus 94
substatus 95
plugins
installing 48
MotD plugin 49
private data storage 89, 90

Q

queues
creating 158, 159
members, adding 161, 162
routing rules, adding 160
routing rules, rule book 160

R

remote connection, monitoring 148, 149
remote network
users, adding from 148
room. *See* **group chat room**
room, creating
boardroom 198
lounge 199
sales HQ room 200
rosters, pre-populating 57-60

S

Search Engine Optimization. *See* **SEO**
search plugin
installing 76
options, tweaking 76, 77
users, searching for 76
users, searching within admin interface 78, 79
server
administrative message, sending 47, 48
cache and properties 39-42
connecting to 26
curb indiscriminate registration 42-44
IM clients, configuring 28
new user, creating with Spark 27
offline users, handling 45, 46
ports 37, 39
resource conflicts, resolving 44, 46
Spark client, installing 25, 26
updates, managing 46, 47
vital statistics 36
server, connecting to
Google talk, enabling 105, 106
IRC, enabling for IT and Devs 103-105
yahoo! messenger, enabling 101-103
server-to-server communication
connectivity on port, checking 147, 148
drawbacks 149, 150
files, transferring accross servers 151
remote connection, monitoring 148, 149
road warrior 150, 151
secure communication, enabling 149, 150
setting up 145, 146
users, adding from remote network 148
sessions, monitoring 173, 174
Single Sign On (SSO) feature, Spark 183
Spark
about 183
chat, closing 188
conversation, tracking 186, 187
features 183, 184
file transfer, settings 191
file transfer directory, changing 187, 188
impromptu group chat room, creating 201, 202
person search window 185
plugins 190-192
profile information, editing 184
Single Sign On (SSO) feature 183, 184
task, adding 189, 190
users, adding 185
using, for conferencing 200, 201
Spark client
installing 25, 26

U

users
adding 51, 52
clients, managing 84-89
deleting 52, 53
editing 52, 53
exporting 90
importing 90
organizing, into groups 54, 55
plugin, using 91-94
searching, search plugin used 76, 77
searching, within admin interface 78, 79
suspending 53, 54

V

VoIP, connecting to 109, 112

W

Webchat
installing 153-155
workgroup
adding 156