



Nama System : IM Server (RICH - RPX Instant Communication Hub)  
Versi : 1.0  
Bulan release : Januari 2009  
Ditujukan untuk SBU : All SBUs - PT. Repex Wahana  
Manual oleh : Amin Yulianto, Web and System Administrator  
Menyetujui : Oky Kurniawan, RND Manager  
Mengetahui : Hasto S Baskoro, Senior Manager ITD

## RICH

(RPX Instant Communication Hub)

# Instant Messaging Server

# Installation Manual

## Table of Content:

Table of Content: .....	2
Preface.....	3
Purpose .....	3
Software used.....	3
Typography .....	3
Pre-Installation.....	3
Preparation.....	3
Install Prerequisites .....	3
Data Storage and Authentication Package .....	4
DNS Records .....	5
Installation .....	5
Basic configuration .....	6
Test Server.....	6
Adding JID - Jabber User ID.....	7
Errors .....	7
Install MU-Conference .....	8
Testing .....	9
Automatic Startup / Shutdown Script .....	9
Jabber / IM Client .....	12
Bibliography.....	12

## Preface

This manual is how to setup an Internal Instant Messaging Server with Jabberd2 and Conference Room using MU-Conferencing. The installation done on existing Linux Server (CentOS 4.6), with MySQL 5 and PHP 5 installed. MySQL 5 and PHP 5 is not the default version for CentOS 4, you must be installed it first from extra repository

## Purpose

The purpose of this IM server is:

- Maintain bandwidth usage for Internet
- Using Internal Instant Messaging for faster communication between staffs.

## Software used

Open source and/or free applications to create this proxy server are:

- CentOS – The Community ENTERprise Operating System
- Jabberd2 – Next Generation XMPP Server
- MU-Conferencing – a component for a Jabber server, an implementation of XEP-0045
- Pandion – An Easy to use XMPP and Jabber client
- MySQL - The world's most popular open source database

## Typography

- **bold phrase**: linux command
- *italic phrase*: content of a file and result of a command

All command executed as root user.

## Pre-Installation

### Preparation

Create Jabber User and Group

```
# groupadd jabber
```

```
# useradd -g jabber jabber
```

Create Directories for PID's and Logs

```
# mkdir -p /usr/local/var/jabberd/pid/
```

```
# chown -R jabber:jabber /usr/local/var/jabberd/pid/
```

Note that Jabberd writes messages to syslog by default. (Good it make my life easier)

### Install Prerequisites

Jabberd 2 has four prerequisites:

- OpenSSL (version 0.9.6b or higher)
- Libidn (version 0.3.0 or higher)
- Data Storage Package
- Authentication Package

```
# rpm -q openssl
openssl-0.9.7a-43.4
# rpm -q libidn
libidn-0.5.6-1
```

I have OpenSSL and libidn installed by CentOS, if your's not installed then type:  
**# yum -y install openssl libidn**

Additional info from README file

Required packages:

- expat - XML parsing libraries  
<http://expat.sourceforge.net/>
- GnuSASL (0.2.27 or higher) - Simple Authentication and Security Layer library  
<http://www.gnu.org/software/gsas/>  
 (please see contrib/ subdir for an important patch)
- UDNS - asynchronous DNS resolver library  
<http://www.corpit.ru/mjt/udns.html>

### Get latest udns package on Dag's RPM

```
# wget http://dag.wieers.com/rpm/packages/udns/udns-0.0.9-1.el4.rf.i386.rpm
# wget http://dag.wieers.com/rpm/packages/udns/udns-devel-0.0.9-1.el4.rf.i386.rpm
# rpm -ivh udns-*.rpm
```

### Install gsasl

```
# cd /usr/src/
# wget ftp://alpha.gnu.org/pub/gnu/gsas/gsas-0.2.29.tar.gz
# tar xzvf gsasl-0.2.29.tar.gz
# cd gsasl-0.2.29
# ./configure
# make
# make install
```

### Install libgsasl

```
# cd /usr/src/
# wget ftp://alpha.gnu.org/pub/gnu/gsas/libgsasl-0.2.29.tar.gz
# tar xvfz libgsasl-0.2.29.tar.gz
# cd libgsasl-0.2.29
# ./configure
# make
# make install
# cd /usr/lib
# ln -s /usr/local/lib/libgsasl.so.7.6.0 libgsasl.so
# ln -s /usr/local/lib/libgsasl.so.7.6.0 libgsasl.so.7
# ln -s /usr/local/lib/libgsasl.la libgsasl.la
```

## Data Storage and Authentication Package

MySQL5 is the default Jabberd2 package for storage and authentication.

So you should install MySQL5 in on your CentOS

```
# yum -y install mysql-server
```

Note: Don't forget install mysql development package, just in case. Because it's configure-make-make install application. Oh yeah you should have gcc, gcc++, make, and other development packages

## DNS Records

Create hostname for your IM server on your DNS server that serving your domain zone, I use im.rpxholding.com

(ps: I won't tell you how, because it's a separate topics)

```
# vi /var/named/rpxholding.com
; RPX Instant Messaging Server
im      IN      A      10.10.105.6
_xmpp-server._tcp.rpxholding.com. 86400 IN SRV 5 0 5269 im.rpxholding.com.
_xmpp-client._tcp.rpxholding.com. 86400 IN SRV 5 0 5222 im.rpxholding.com.
# service named reload
```

## Installation

Download latest jabberd from <http://ftp.xiaoka.com>, this is what I use when I make this IM server / howto

```
# wget http://ftp.xiaoka.com/jabberd2/releases/jabberd-2.2.4.tar.bz2
```

Extract it

```
# tar xjvf jabberd-2.2.4.tar.bz2
```

Configure it

```
# cd jabberd-2.2.4
```

Prior to configuring Jabberd, you can view all configuration options by running the command:

```
# ./configure --help
```

The authentication and storage packages should be specified with the `--enable-` option.

Options are `--enable-mysql`, `--enable-pgsql`, `--enable-db`, `--enable-pam` or `--enable-ldap`.

Although `--enable-mysql` is the default, it is recommended that this be specified if MySQL is to be used. For example, this command would be used to enable MySQL as the authentication and storage package without debugging support:

```
# ./configure --enable-mysql --enable-ssl --enable-idn
# make
# make install
```

Default File Location for Jabberd

Configuration Files: `/usr/local/etc/`

Binary Files: `/usr/local/bin`

Set Ownership

```
# chown -R root:jabber /usr/local/etc/*
```

```
# chmod -R 640 /usr/local/etc
```

## Basic configuration

### Set Host Name

Enter your IM server hostname that you've defined before

```
# vi /usr/local/etc/sm.xml
<id>rpxholding.com</id>
# administrative users
<acl type='all'>
  <jid>helpdesk@rpxholding.com</jid>
  <jid>ayuliastanto@rpxholding.com</jid>
</acl>
uncomment the line
<roster>/usr/local/etc/templates/roster.xml</roster>
<auto-create/>
```

```
# vi /usr/local/etc/c2s.xml
change line
<id register-enable='true'>localhost.localdomain</id>
into
<id>rpxholding.com</id>
```

```
# vi /usr/local/etc/templates/roster.xml
<query xmlns='jabber:iq:roster'>
<item name='Helpdesk' jid='helpdesk@rpxholding.com' subscription='none'><group>ITD</group></item>
<item name='Your-Beloved SysAdmin' jid='ayuliastanto@rpxholding.com'
subscription='none'><group>ITD</group></item>
</query>
```

### Provision and Configure for Storage and Authentication Package using MySQL

```
# cd tools/
# mysql -u root -p
mysql> \. db-setup.mysql
mysql> GRANT select,insert,delete,update ON jabberd2.*
  -> to jabberd2@localhost IDENTIFIED by 'secret';
# ln -s /var/lib/mysql/mysql.sock /tmp/mysql.sock
```

### Configure for Storage using MySQL (sm.xml)

```
# vi /usr/local/etc/sm.xml
Didn't do anything as by defaults Jabberd2 is use MySQL for Storage :D
```

### Configure for Authentication using MySQL (c2s.xml)

```
# vi /usr/local/etc/c2s.xml
Didn't do anything as by defaults Jabberd2 is use MySQL for Storage :D
```

## Test Server

Note: while you are running the server, you can monitor jabber log for output and error by giving command

```
# tail -f /var/log/messages
of course on different terminal / putty screen
```

```
# su jabber
$ cd /usr/local/bin/
$ ./jabberd
ERROR: router died. Shutting down server.
```

**ooopppsss !!!?????**

```
$ exit
# chown root.jabber /usr/local/etc/
# chmod 750 /usr/local/etc/
# su jabber
$ cd /usr/local/bin/
$ ./jabberd
```

**yiippppyy... no error**

## Adding JID - Jabber User ID

```
# mysql -u root -p
mysql> use jabberd2
mysql> insert into authreg (username, realm, password) values ('myusername', 'somedomain.com', 'mypassword');
```

But I am using another script to synchronize existing UserAccess database to this Jabberd2's database. I am not going to discuss it here. **S-E-C-R-E-T**

## Errors

These errors happen when connecting from Jabber-Client into server.

**Error 1:**  
XML parse error (not well-formed (invalid token))

Fix:  
# vi /usr/local/etc/c2s.xml  
Comment this line  
<digest-md5/>

**Error 2:**  
SASL authentication succeeded: mechanism=PLAIN; authzid=ayuliastanto@rpxholding.com  
bound: jid=ayuliastanto@rpxholding.com/AM3N  
user not found, can't start session: jid=ayuliastanto@rpxholding.com/AM3N

Fix:  
# vi /usr/local/etc/c2s.xml  
Uncomment this line  
<auto-create/>

**Error 3:**  
couldn't stat roster template /usr/local/etc/templates/roster.xml: Permission denied

Fix:  
# chmod 750 /usr/local/etc/templates/

## Install MU-Conference

Multi User Conferencing for Jabberd2. For MU-conference 0.7 or better, libglib-2, expat and libidn11 are required. The installation of these libraries and/or packages is beyond the scope of this document.

```
# rpm -q glib2 expat libidn
```

```
glib2-2.4.7-1
```

```
expat-1.95.7-4
```

```
libidn-0.5.6-1
```

On my system, they all installed. Yippy

```
# wget http://download.gna.org/mu-conference/mu-conference_0.7.tar.gz
```

```
# tar -xvf mu-conference_0.7.tar.gz
```

```
# cd mu-conference_0.7
```

```
# make
```

```
# mkdir -p /usr/local/var/jabberd/spool/rooms
```

```
# mkdir -p /usr/local/var/jabberd/log
```

```
# cp muc-default.xml src/muc.xml
```

```
# vi src/muc.xml
```

```
<jcr>
```

```
<name>im.rpxholding.com</name>
```

```
<host>im.rpxholding.com</host>
```

```
<ip>localhost</ip>
```

```
<port>5347</port>
```

```
<secret>secret</secret>
```

```
<spool>/usr/local/var/jabberd/spool/rooms</spool>
```

```
<logdir>/usr/local/var/jabberd/log</logdir>
```

```
<pidfile>/usr/local/var/jabberd/pid/mu-conference.pid</pidfile>
```

```
<loglevel>124</loglevel>
```

```
<conference xmlns="jabber:config:conference">
```

```
<public/>
```

```
<vCard>
```

```
<FN>RPX Chatrooms</FN>
```

```
<DESC>This service is for RPX chatrooms.</DESC>
```

```
<URL>http://im.rpxholding.com/</URL>
```

```
</vCard>
```

```
<history>40</history>
```

```
<logdir>/usr/local/var/jabberd/log</logdir>
```

```
<stylesheet>/usr/local/etc/style.css</stylesheet>
```

```
<notice>
```

```
<join>has become available</join>
```

```
<leave>has left</leave>
```

```
<rename>is now known as</rename>
```

```
</notice>
```

```
<sadmin>
```

```
<user>ayuliasanto@rpxholding.com</user>
```

```
<user>helpdesk@rpxholding.com</user>
```

```
</sadmin>
```

```
<persistent/>
```

```
<roomlock/>
```

```
</jcr>
```

## Testing

```
# src/mu-conference -c src/muc.xml
```

```
Integration mu-conference with Jabberd2
# cp src/mu-conference /usr/local/bin/mu-conference
# cp style.css /usr/local/etc
# cp src/muc.xml /usr/local/etc/mu-conference.xml
# chown -R jabber.jabber /usr/local/var/jabberd/
```

## Automatic Startup / Shutdown Script

Modified from: <http://www.opennms.org/index.php/Jabberd2InitScript>

```
# vi /etc/rc.d/init.d/jabberd2
#!/bin/bash
#
# Raymond 25DEC2003 support@bigriverinfotech.com
# /etc/rc.d/init.d/jabberd2
# init script for jabberd2 processes
# Tested under jabberd-2.0rc2 and Fedora 1.0 only
#
# processname: jabberd2
# description: jabberd2 is the next generation of the jabberd server
# chkconfig: 2345 85 15
#
if [ -f /etc/init.d/functions ]; then
    . /etc/init.d/functions
elif [ -f /etc/rc.d/init.d/functions ]; then
    . /etc/rc.d/init.d/functions
else
    echo -e "\ajabberd2: unable to locate functions lib. Cannot continue."
    exit -1
fi
#
progs="router sm c2s s2s mu-conference"
progsPath="/usr/local/bin"
confPath="/usr/local/etc"
pidPath="/usr/local/var/jabberd/pid"
statusCol="echo -ne \033[60G"
statusColorOK="echo -ne \033[1;32m"
statusColorFailed="echo -ne \033[1;31m"
statusColorNormal="echo -ne \033[0;39m"
retval=0
#
StatusOK ( ) {
    ${statusCol}
    echo -n "[ "
    ${statusColorOK}
    echo -n "OK"
    ${statusColorNormal}
    echo " ]"
    return 0
}
#
StatusFailed ( ) {
    echo -ne "\a"
```

```

    ${statusCol}
    echo -n "["
    ${statusColorFailed}
    echo -n "FAILED"
    ${statusColorNormal}
    echo "]"
    return 0
}
#
ReqBins ( ) {
    for prog in ${progs}; do
        if [ ! -x ${progsPath}/${prog} ]; then
            echo -n "jabberd2 binary [${prog}] not found."
            StatusFailed
            echo "Cannot continue."
            return -1
        fi
    done
    return 0
}
#
ReqConfs ( ) {
    for prog in ${progs}; do
        if [ ! -f ${confPath}/${prog}.xml ]; then
            echo -n "jabberd2 configuration [${prog}.xml] not found."
            StatusFailed
            echo "Cannot continue."
            return -1
        fi
    done
    return 0
}
#
ReqDirs ( ) {
    if [ ! -d ${pidPath} ]; then
        echo -n "jabberd2 PID directory not found. Cannot continue."
        StatusFailed
        return -1
    fi
    return 0
}
#
Start ( ) {
    for req in ReqBins ReqConfs ReqDirs; do
        ${req}
        retval=$?
        [ ${retval} == 0 ] || return ${retval}
    done
    echo "Initializing jabberd2 processes ..."
    for prog in ${progs}; do
        if [ $( pidof -s ${prog} ) ]; then
            echo -ne "\tprocess [${prog}] already running"
            StatusFailed
            sleep 1
            continue
        fi
    done
}

```

```

echo -ne "\tStarting ${prog}: "
if [ ${prog} == "router" ]; then
    ports="5347"
elif [ ${prog} == "c2s" ]; then
    ports="5222 5223"
elif [ ${prog} == "s2s" ]; then
    ports="5269"
else
    ports=""
fi
for port in ${ports}; do
    if [ $( netstat --numeric-ports --listening --protocol=inet |
        gawk '{ print $4 }' |
        gawk -F : '{ print $NF }' |
        grep -c ${port}$ ) -ne "0" ]; then
        StatusFailed
        echo -e "\tPort ${port} is currently in use. Cannot continue"
        echo -e "\tIs a Jabber 1.x server running?"
        Stop
        let retval=-1
        break 2
    fi
done
rm -f /var/lock/subsys/${prog}
rm -f ${pidPath}/${prog}.pid
args="-c ${confPath}/${prog}.xml"
su - jabber -c "${progsPath}/${prog} ${args} & 2> /dev/null"
retval=$?
if [ ${retval} == 0 ]; then
    StatusOK
    touch /var/lock/subsys/${prog}
else
    StatusFailed
    Stop
    let retval=-1
    break
fi
sleep 1
done
return ${retval}
}
#
Stop ( ) {
    echo "Terminating jabberd2 processes ..."
    for prog in ${progs}; do
        echo -ne "\tStopping ${prog}: "
        killproc ${prog}
        retval=$?
        if [ ${retval} == 0 ]; then
            rm -f /var/lock/subsys/${prog}
            rm -f ${pidPath}/${prog}.pid
        fi
        echo
        sleep 1
    done
    return ${retval}
}

```

```

}
#
case "$1" in
    start)
        Start
        ;;
    stop)
        Stop
        ;;
    restart)
        Stop
        Start
        ;;
    condrestart)
        if [ -f /var/lock/subsys/${prog} ]; then
            Stop
            sleep 3
            Start
        fi
        ;;
    *)
        echo "Usage: $0 {start/stop/restart/condrestart}"
        let retval=-1
esac
exit ${retval}
#
# eof

```

```

# chmod +x /etc/rc.d/init.d/jabberd2
# chkconfig --add jabberd2

```

```

Start, stop and restart
# /etc/rc.d/init.d/jabberd2 start
# /etc/rc.d/init.d/jabberd2 restart
# /etc/rc.d/init.d/jabberd2 stop

```

```

or you can do it in SysV style
# service jabberd2 start
# service jabberd2 stop
# service jabberd2 restart

```

## Jabber / IM Client

I use Pandion <<http://pandion.be/>> - An easy to use XMPP - and it's simple yet powerful. I love it

## Bibliography

- <http://xmpp.org/>
- [http://www.jabber.org/web/Main\\_Page](http://www.jabber.org/web/Main_Page)
- <http://jabberd2.xiaoka.com/>
- <http://www.unix-tutorials.com/go.php?id=143>
- <https://gna.org/projects/mu-conference/>
- <http://www.opennms.org/index.php/Jabberd2InitScript>