



Jami Account Management Server



25.02

Jami is a registered trademark and developed by Savoir-faire Linux Inc.
Further information is available at jami.net

Copyright

This document is Copyright © 2018-2025 by Savoir-faire Linux Inc. You may distribute it and/or modify it under the terms of either the GNU General Public License (<https://www.gnu.org/licenses/gpl.html>), version 3 or later, or the Creative Commons Attribution License (<https://creativecommons.org/licenses/by/4.0/>), version 4.0 or later.

All trademarks within this guide belong to their legitimate owners.

Community and support

Learn more about:

- Jami: <https://jami.net/>
- Jami extensions: <https://jami.net/extensions/>
- JAMS (Jami Account Management Server): <https://jami.biz/>
- Jami documentation: <https://docs.jami.net/>

Follow us for more:

- Mastodon: [@Jami@mstdn.io](https://mstdn.io/@Jami)
- Videos: <https://docs.jami.net/videos/>

We'd love to hear from you! Join the Jami community:

- Contribute: <https://jami.net/contribute/>
- Forum: <https://forum.jami.net/>
- GNU Mailman: jami@gnu.org
- Libera.Chat: [#jami](https://libera.chat/#jami)
- Matrix: [#jami:matrix.org](https://matrix.org/#jami:matrix.org) (bridged with Libera.Chat)



Caution

Everything you send to a mailing list, including your email address and any other personal information that is written in the message, is publicly archived and is unable to be deleted.

Table of Contents

Copyright.....	2
Community and support.....	2
JAMS manual.....	4
Introduction.....	4
Obtaining JAMS.....	4
System requirements.....	4
JAMS concepts.....	5
Getting started.....	6
Step 1: Create an administrator account.....	7
Step 2: Set up the Certification Authority.....	8
Step 3: Set up the user database.....	9
Option 1: Lightweight Directory Access Protocol (LDAP).....	9
Option 2: Microsoft Active Directory (AD).....	10
Option 3: Local embedded database.....	11
Step 4: Set up the server parameters.....	12
Private DHT node.....	13
Admin guide.....	14
JAMS and Nginx.....	14
Troubleshooting and resetting.....	15
Running JAMS as a GNU/Linux Service.....	16
Running JAMS as a Windows Service.....	16
A. Download and install JAMS.....	16
B. Download and install Java Development Kit (JDK).....	16
C. Download OpenSSL to generate a key and a certificate.....	16
D. Add OpenSSL to System Environment Variables.....	18
E. Configure OpenSSL.....	18
F. Expose the localhost to the Internet.....	19
G. Create a JAMS Windows Service (Embed Tomcat Server Windows Service) to start JAMS with the server.....	20
Client guide.....	21
Connect with desktop clients.....	22
Connect with Android clients.....	23
Connect with iOS clients.....	24
Connect with web clients.....	25

JAMS manual

The JAMS manual contains admin and user guides for the Jami Account Management Server. The Jami Account Management Server (JAMS) enables Jami to be easily deployed in any enterprise and allows users to connect using their centralized credentials and create local accounts. JAMS allows all enterprises to manage their own Jami community while taking advantage of Jami's distributed network architecture.

Introduction

JAMS is a server application used to enroll Jami clients in an enterprise environment. Currently, JAMS supports 3 sources for user authentication:

1. Lightweight Directory Access Protocol (LDAP),
2. Active Directory (AD), and
3. An embedded database.

Obtaining JAMS

The latest version of JAMS can be downloaded at <https://jami.biz/>. The source code is available at <https://git.jami.net/savoirfairelinux/jami-jams>.

System requirements

Requirement	Details
Processor	1 gigahertz (GHz) or faster with 1 or more cores on a compatible 64-bit processor or System on a Chip (SoC).
RAM	4 gigabytes (GB).
Storage	1 gigabyte (GB) of available storage space.
Operating system	GNU/Linux, Apple macOS, or Microsoft Windows.
Java	Version 11 or higher.
Database	LDAP-compatible directory (such as OpenLDAP), Microsoft Active Directory (AD), or a local embedded database.

JAMS concepts

JAMS was built with security in mind; therefore, it is intimately related to the [X.509](#) certificate management workflows.

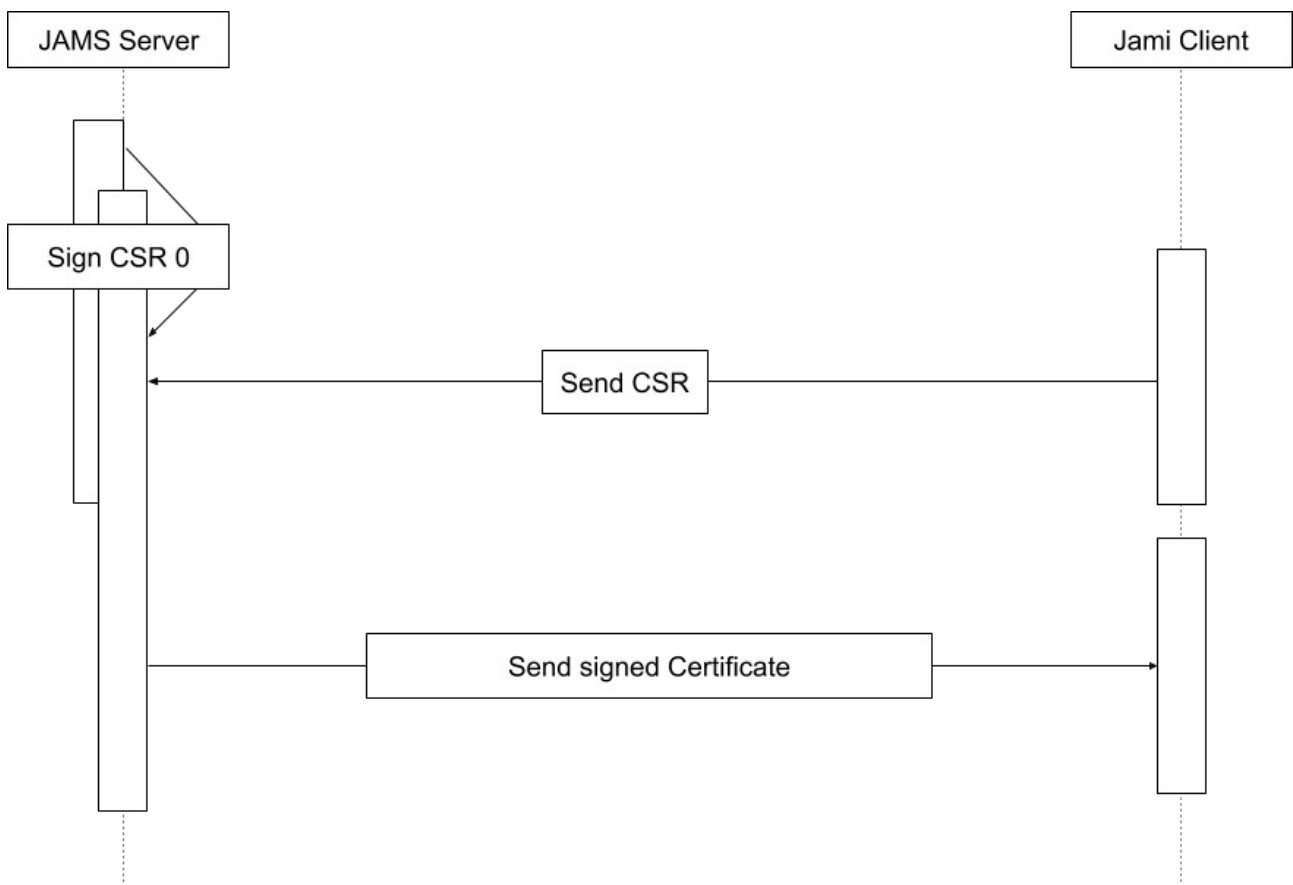
The central concepts that are used in JAMS are:

- the [Certification Authority \(CA\)](#) and
- the [Certificate Signing Request \(CSR\)](#).

In the JAMS paradigm, a device (Jami client) requests a certificate from the server and then presents it to other devices to be recognized as a valid member of the organization. Therefore, JAMS must be provided with a certificate authority in order to work properly.

In order to be completely secure, JAMS does not generate certificates for devices. JAMS instead issues certificates based on a certificate signing request sent to it by the device. This therefore removes the requirement to send a private key over the wire.

The following diagram shows the entire process of how a device enrolls with JAMS:



Getting started

1. Download the latest version of JAMS from <https://jami.biz/>.
2. Unpack the .tar file to any directory.
3. It is mandatory to run JAMS using a secure SSL connection.

A domain name is required to request a key and a certificate. A domain name can be purchased if one is not available. Set the domain name to point to the server before proceeding to the next step.

A pair of key certificates can be purchased from any online provider. However, obtaining a free pair using [Let's Encrypt](#) is recommended.

In order to generate a pair of key certificates, **Certbot** can be used following the instructions on the <https://certbot.eff.org/> page.

Certbot will provide specific instructions when the web server software and operating system are entered.

Install Certbot using snap:

```
sudo snap install --classic certbot
```

Ensure that the Certbot command can be run:

```
sudo ln -s /snap/bin/certbot /usr/bin/certbot`
```

In order to get a certificate, execute:

```
sudo certbot certonly
```

and follow the instructions.

The certificate and key are generated in a specific folder; please see the output from Certbot to locate them.

It is required to copy them in the current folder where the `jams-launcher.jar` file is located.



Current limitation

JAMS currently does not support reading encrypted private keys that require a password unlock.

4. Navigate to the directory where the JAMS package has been extracted and execute the following command:

```
java -jar jams-launcher.jar PORT SSL_CERTIFICATE SSL_CERTIFICATE_KEY
```

Argument	Details
PORT	The TCP port on which JAMS is to listen for incoming connections.
SSL_CERTIFICATE	The location of the PEM-formatted SSL Certificate file.
SSL_CERTIFICATE_KEY	The location of the PEM-formatted key file, which is used with the SSL Certificate file from above.

An example of the command would be:

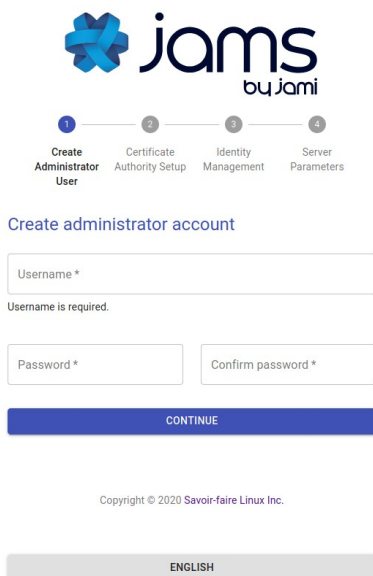
```
java -jar jams-launcher 443 server.pem server.key
```


Note

Any port above 1024 can be safely used to run JAMS.

Step 1: Create an administrator account

The administrator account manages Jami users and groups.





1 — 2 — 3 — 4
Create Administrator User Certificate Authority Setup Identity Management Server Parameters

Create administrator account

Username *

Username is required.

Password * Confirm password *

CONTINUE

Copyright © 2020 Savorir-faire Linux Inc.

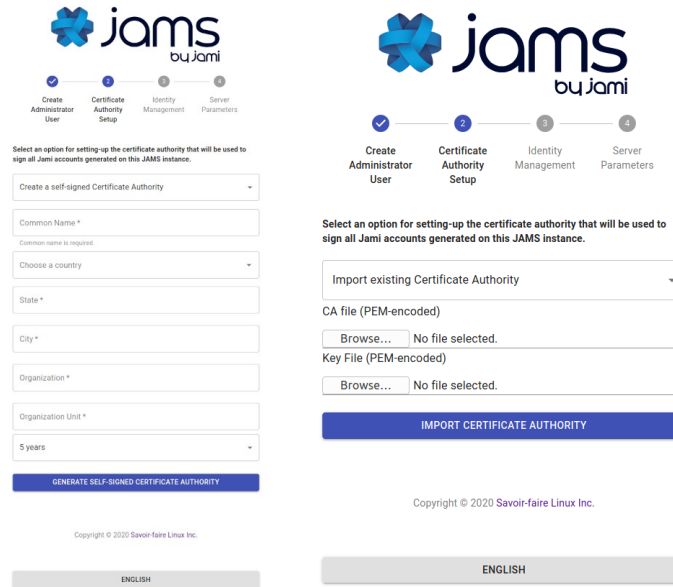
ENGLISH

Step 2: Set up the Certification Authority

The second step is to define the certification authority.

Important

A CA is not a server SSL certificate; it is a certificate that has the power to issue other certificates. Do not use the import option unless the enterprise's security officer has issued the CA certificate. Most commercially available certificates (i.e., those issued by GoDaddy, Let's Encrypt, etc.) are not CA certificates. It is highly recommended that end-users create and use a self-signed CA, as providing an incorrect certification type will lead to a non-functional server.



The image displays two screenshots of the JAMS Certificate Authority Setup interface. Both screenshots show a progress bar at the top with four steps: 1. Create Administrator User, 2. Certificate Authority Setup, 3. Identity Management, and 4. Server Parameters. The left screenshot shows the 'Create a self-signed Certificate Authority' option selected in a dropdown menu. Below this, there are input fields for 'Common Name *', 'Choose a country', 'State *', 'City *', 'Organization *', and 'Organization Unit *'. A '5 years' dropdown is also present. A blue button labeled 'GENERATE SELF-SIGNED CERTIFICATE AUTHORITY' is at the bottom. The right screenshot shows the 'Import existing Certificate Authority' option selected. Below this, there are two 'Browse...' buttons for 'CA file (PEM-encoded)' and 'Key File (PEM-encoded)', both showing 'No file selected.'. A blue button labeled 'IMPORT CERTIFICATE AUTHORITY' is at the bottom. Both screenshots include a copyright notice 'Copyright © 2020 Savoir-faire Linux Inc.' and an 'ENGLISH' button at the bottom.

This certificate will be used to sign the enrollment requests that come from Jami devices. It is highly recommended that the following articles are read to become familiar with the X.509 certificate standard processes and practices:

- <https://www.securew2.com/blog/public-key-infrastructure-explained/>
- <https://cheapsslsecurity.com/blog/understanding-the-role-of-certificate-authorities-in-pki/>

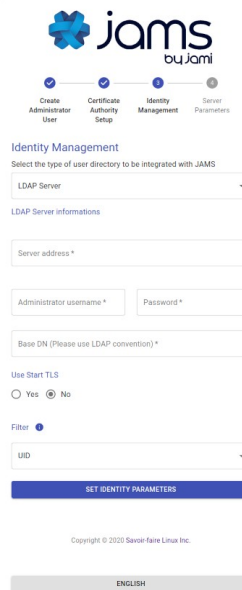
Step 3: Set up the user database

JAMS supports 3 different sources for the authentication of users:

1. LDAP-compatible directory (such as [OpenLDAP](#))
2. Microsoft Active Directory
3. Local embedded database

Option 1: Lightweight Directory Access Protocol (LDAP)

If the enterprise provides an LDAP directory for user management, it is required to know its access information and an automated account that has read-only rights to do use look-ups.



The admin should provide most of the required information; however, the following is a detailed overview of each field:

Field	Details
Use StartTLS	The LDAP server can be configured to use either TLS/STARTTLS or PLAIN sockets; if STARTTLS is used, mark the value as true.
Server Address	The address of the server with respect to the JAMS server, the LDAP is not required to be publicly accessible but should be accessible to JAMS. Either <code>ldap://</code> or <code>ldaps://</code> should precede the address.
Port	The port on which the LDAP server is listening for requests (usually 389 for PLAIN/STARTTLS and 636 for SSL/TLS).
Administrator Username	This is NOT the LDAP's administration account credentials but the credentials of the account that has <i>read</i> permissions to the LDAP database in order to look up users. The format is generally <code>cn=bot,ou=robots,dc=domain,dc=org</code> .
Password	The password used by the account above.
BaseDN	The base realm where the user accounts are located; in most cases, it is <code>ou=users,dc=enterprise,dc=org</code> .

Option 2: Microsoft Active Directory (AD)

If the enterprise provides Active Directory (AD) for user management, it is required to know its access information and an automated account that has read-only rights to do use look-ups.

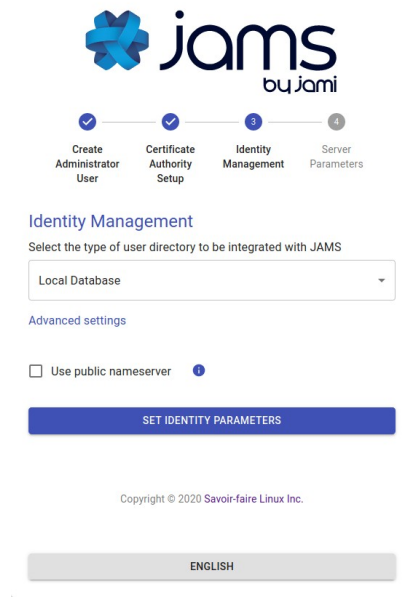
The screenshot shows the JAMS web interface for Identity Management. At the top, there is a progress bar with four steps: 'Create Administrator User' (checked), 'Certificate Authority Setup' (checked), 'Identity Management' (active), and 'Server Parameters'. Below the progress bar, the 'Identity Management' section is active. It includes a dropdown menu for 'Select the type of user directory to be integrated with JAMS' set to 'Active Directory'. Under 'Active Directory server information', there are input fields for 'Port *', 'Host *', 'Admin username *', 'Password *', and 'Domain Name *'. A 'Use SSL' section has radio buttons for 'Yes' and 'No', with 'No' selected. A blue button labeled 'SET IDENTITY PARAMETERS' is at the bottom. Copyright information for 'Savoir-faire Linux Inc.' and a language selector for 'ENGLISH' are also visible.

The admin should provide most of the required information; however, the following is a detailed overview of each field:

Field	Details
Port	The port on which Active Directory (AD) is listening (generally it is either 389 or 636).
Host	The address of the server with respect to the JAMS server, the Active Directory (AD). It not required to be publicly accessible but should be accessible to JAMS.
Administrator Username	This is NOT the Active Directory's administration account credentials but the credentials of the account that has <i>read</i> permissions to the Active Directory database in order to look up users. The format is generally <code>cn=bot,ou=robots,dc=domain,dc=net</code> .
Password	The password used by the account above.
Use SSL	Whether the server uses SSL for data transmission.
Domain Name	This is the legacy-formatted Windows Domain Name (i.e., WINDOMAIN).

Option 3: Local embedded database

The local database does not require any additional configuration; everything in the process is automated. This option allows for the creation of Jami users on the fly directly from the JAMS interface.



Advanced settings

By default, the “Use public name server” option is disabled. Jami usernames of JAMS users will not be stored on the public Jami name server. Users can communicate with users outside the organization by using their 40-character fingerprint. Enable this option to allow JAMS users in the organization to also search for external users on the public name server.

Step 4: Set up the server parameters

Server Parameters

The global parameters cover the general configuration of the server's engine.

CORS domain name

The domain name of your web client server. Requires http:// or https://

Domain *

https://localhost:8080

Certificate Revocation List Lifetime

5 minutes

Device Lifetime

1 Year

User account lifetime

1 Year

SIP Configuration Template

Browse... No file selected.

SET SERVER PARAMETERS

Copyright © 2020 Savoir-faire Linux Inc.

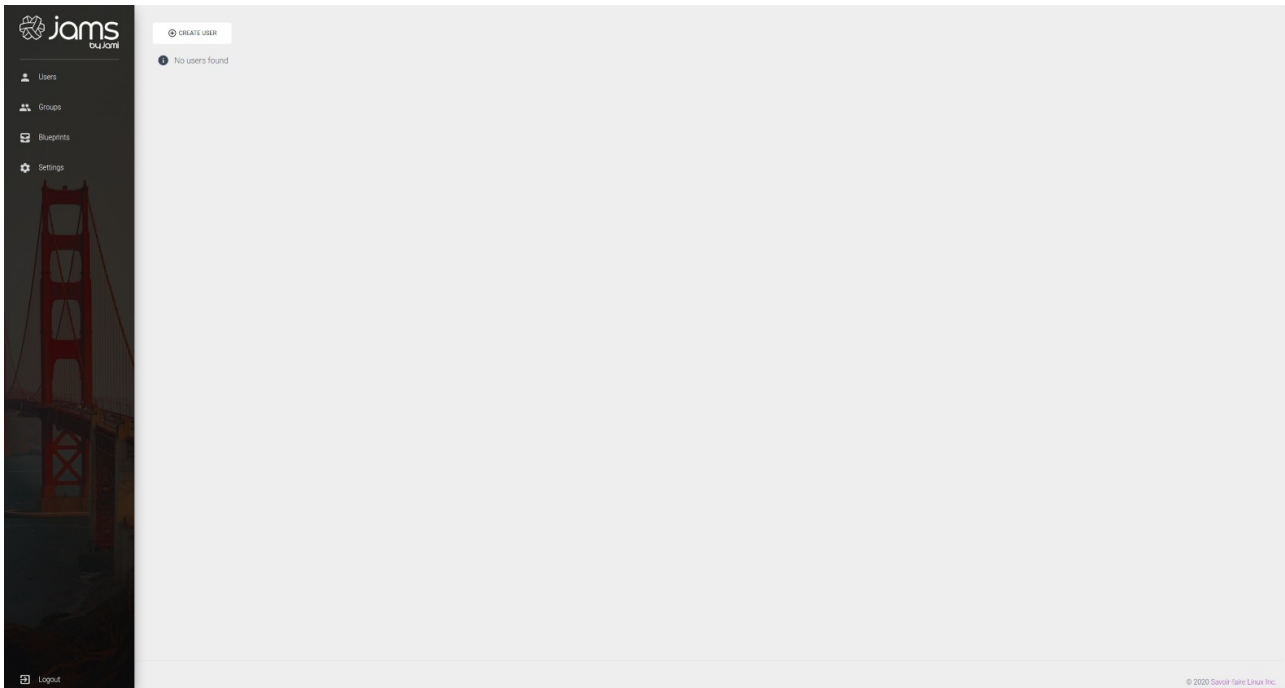
ENGLISH

Parameter	Details
CORS Domain Name	The domain on which the JAMS client and administration UI will be running.
Certificate Revocation List Lifetime	The frequency at which the CRL is updated in memory.
Device Lifetime	How long a device's certificate is valid before being considered stale and requiring re-enrollment.
User Account Lifetime	How long a user account is valid before being considered stale and requiring re-enrollment.

✔ Important

The *CORS Domain Name* corresponds to the web address used to access the Web UI. By default, it is set to the same URL address as the one where JAMS is deployed. Only set a different URL address if the Web UI has a different URL address from the one where JAMS is deployed.

Select the **Set Server Parameters** button to finalize the configuration and be redirected to the JAMS interface.



If JAMS has been configured with an LDAP database or Active Directory (AD), the list of users in the organization should be visible in JAMS. If JAMS has been configured with a local embedded database, new users can be created by selecting the **Create User** button.

Private DHT node

1. The JAMS server does not provide an [OpenDHT](#) bootstrap node. OpenDHT is required to be installed to provide bootstrap functionality.
2. If the network no longer has Internet access, clients should be able to communicate with this single DHT node, but it can take some time (10-15 minutes) to update the table.
3. The DHT is only used to allow nodes to find each other: data is sent P2P or through a TURN server. If required, the TURN server is installed separately.

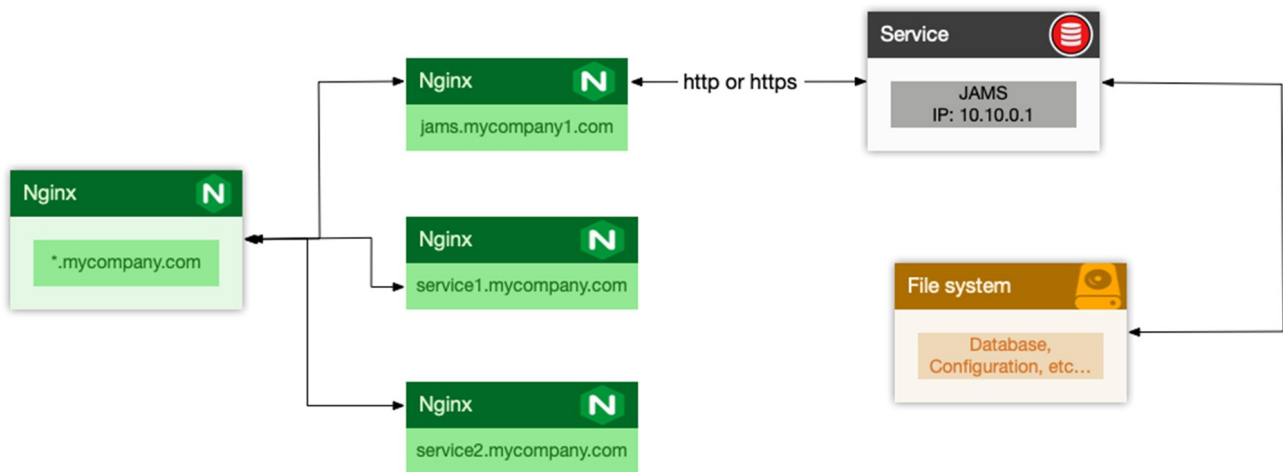
Admin guide

By default, the [JAMS \(Jami Account Management Server\)](#) runs an embedded [Apache Tomcat](#) server visible on port 8080. However, this is impractical for many reasons. This guide is designed to help with setting up a JAMS to run in a production environment.

JAMS and Nginx

It is generally not recommended to expose JAMS directly to the outside world. JAMS is required to run in SSL mode. It is recommended to place JAMS behind an [Nginx](#) or similar web server. The Nginx, or similar, web server would proxy requests between the outside world and JAMS.

The following is an example map of how the JAMS could be configured behind an Nginx server. The process would be similar if any other type of proxy solution is used.



The IP 10.10.0.1 is random and should be seen as an example.

Typically a new site called `jams-site.conf` would be added to the Nginx configuration. It would contain the following entries if an SSL certificate was placed at the Nginx level:

```
server {
    listen 443 ssl;
    listen [::]:443 ssl;
    ssl on;
    ssl_certificate /etc/certificates/mycertificate.pem
    ssl_certificate_key /etc/certificates/mycertificatekey.pem
    ssl_client_certificate /jams/installation/path/CA.pem;
    ssl_verify_client optional;
    ssl_verify_depth 2;
    client_max_body_size 100M;
    server_name jams.mycompany.com;
    location / {
        # Block client-supplied headers that could be used to spoof
        if ($http_x_client_cert) {
            return 400;
        }
        proxy_pass          http://10.10.0.1:8080/;
        proxy_set_header    X-Real-IP $remote_addr;
        proxy_set_header    Host $http_host;
        proxy_set_header    X-Client-Cert $ssl_client_escaped_cert;
    }
}
```

This is the preferred setup method by most admins, as local traffic is usually run unencrypted since it is usually either an inter-VM connection, a VLAN, or another dedicated link.

Tip

Since the CA is generated during the JAMS initial configuration, Nginx needs to be restarted once the initial setup is completed.

Troubleshooting and resetting

If a restart from 0 (i.e., reset everything and drop existing data) is required, delete the following files in the distribution folder (`<project-root-folder>/jams`):

The internal JAMS folder: `<project-root-folder>/jams/jams`
derby.log
oauth.key
oauth.pub
config.json

This will reset the server to its original state, and the configuration wizard is able to be run again. Before performing this operation, please ensure that the server is shut down.

Running JAMS as a GNU/Linux Service

Running JAMS as a GNU/Linux Service is fairly straightforward with systemd—simply create a service unit file with the following structure:

```
[Unit]
Description=JAMS Server

[Service]
Type=simple
WorkingDirectory=[DIRECTORY WHERE JAMS WAS UNZIPPED]
ExecStart=/usr/bin/java -jar [DIRECTORY WHERE JAMS WAS UNZIPPED]/jams-
launcher.jar PORT SSL_CERTIFICATE SSL_CERTIFICATE_KEY

[Install]
WantedBy=multi-user.target
```

The parameters **PORT**, **SSL_CERTIFICATE** and **SSL_CERTIFICATE_KEY** are optional (however, **PORT** can be used alone, whereas the **SSL_CERTIFICATE** comes in a pair with **SSL_CERTIFICATE_KEY**).

Running JAMS as a Windows Service

A. Download and install JAMS

1. Visit <https://jami.biz/> and download JAMS.
2. Extract JAMS to C:\jams

B. Download and install Java Development Kit (JDK)

1. Download JDK 11 from <https://www.oracle.com/java/technologies/javase-jdk11-downloads.html> (choose the corresponding VM architecture).
2. Install it using the installation wizard.

C. Download OpenSSL to generate a key and a certificate

1. Download the OpenSSL Binary Distributions for Microsoft Windows from <https://kb.fireDaemon.com/support/solutions/articles/4000121705>. Alternatively, select another [OpenSSL binary](#).
2. Once downloaded, extract it to C:\openssl.
3. Create a bin folder inside, i.e., C:\openssl\bin.
4. Create a new file inside the bin folder named openssl.cnf (make sure that the file extension is .cnd and not .cnd.txt).
5. Copy and paste the following default configuration from <http://www.flatmtn.com/article/setting-openssl-create-certificates.html>:


```

#
# OpenSSL configuration file.
#

# Establish working directory.

dir                = .

[ca]
default_ca         = CA_default

[CA_default]
serial            = $dir/serial
database          = $dir/certindex.txt
new_certs_dir     = $dir/certs
certificate        = $dir/cacert.pem
private_key        = $dir/private/cakey.pem
default_days      = 365
default_md         = md5
preserve          = no
email_in_dn       = no
nameopt           = default_ca
certopt           = default_ca
policy            = policy_match

[policy_match]
countryName       = match
stateOrProvinceName = match
organizationName  = match
organizationalUnitName = optional
commonName        = supplied
emailAddress       = optional

[req]
default_bits      = 1024      # Size of keys
default_keyfile   = key.pem   # Name of generated keys
default_md        = md5       # Message digest algorithm
string_mask       = nombstr   # Permitted characters
distinguished_name = req_distinguished_name
req_extensions    = v3_req

[req_distinguished_name]
# Variable name          Prompt string
#-----
0.organizationName      = Organization Name (company)
organizationalUnitName  = Organizational Unit Name
                        (department, division)
emailAddress             = Email Address
emailAddress_max        = 40
localityName            = Locality Name (city, district)
stateOrProvinceName     = State or Province Name (full name)
countryName             = Country Name (2 letter code)
countryName_min         = 2
countryName_max         = 2
commonName              = Common Name (hostname, IP, or username)
commonName_max          = 64

# Default values for the above, for consistency and less typing.
# Variable name          Value
#-----
0.organizationName_default = My Company
localityName_default      = My Town
stateOrProvinceName_default = State or Province

```

```

countryName_default      = US

[v3_ca]
basicConstraints         = CA:TRUE
subjectKeyIdentifier     = hash
authorityKeyIdentifier   = keyid:always,issuer:always

[v3_req]
basicConstraints         = CA:FALSE
subjectKeyIdentifier     = hash

```

D. Add OpenSSL to System Environment Variables

Go to Edit the system environment variables → Environment Variables. In System variables, edit **Path** and add C:\openssl\.

E. Configure OpenSSL

1. Open the **Command Prompt**.
2. Execute the following command to set the path to the OpenSSL configuration file.
set OPENSSL_CONF=C:\openssl\bin\openssl.cnf
3. Type cd C:\jams
4. To generate the **Key** and **Certificate**, type:
openssl req -newkey rsa:2048 -new -nodes -x509 -days 3650
-keyout server.key -out server.pem
5. Follow the wizard.
6. Once the key and certificate are generated, type dir. The output should look like:

```

C:\jams>dir
Volume in drive C has no label.
Volume Serial Number is BC94-9EF2

Directory of C:\jams

2020-11-10 12:38 PM <DIR>          .
2020-11-10 12:38 PM <DIR>          ..
2020-10-22 10:56 AM           5,186,016 jams-launcher.jar
2020-10-22 10:56 AM          33,413,882 jams-server.jar
2020-11-10 11:53 AM <DIR>          libs
2020-11-10 12:34 PM           1,732 server.key
2020-11-10 12:38 PM           1,336 server.pem
2020-10-22 04:05 PM          2,047,932 userguide.pdf
                5 File(s)          40,650,898 bytes
                3 Dir(s)          93,365,936,128 bytes free

```

7. Execute the following command to start JAMS:
java -jar jams-launcher.jar PORT_NUMBER (eg. 8443 or 443)
server.pem server.key
8. Open a navigator on the server and visit <https://localhost:443> or <https://localhost:8443> to validate that JAMS is working.
9. Type CTRL+C to close the application.

F. Expose the localhost to the Internet

1. Press the Windows key or click the Windows icon and search for **Windows Defender Firewall with Advanced Security**.
2. Right-click **Inbound Rules** and click **New Rule...**
3. Select **Port**, click **Next**.
4. Specify the port to use, for example, **443** or **8443**, and click **Next**.
5. Select **Allow the connection** and click **Next**.
6. Leave Domain Private and Public unchanged, and click **Next**.
7. Name the rule **JAMS Inbound** and click **Finish**.
8. Right-click on **Outbound Rules** and click **New Rule...**
9. Select **Port**, click **Next**.
10. Specify the port to use, for example, **443** or **8443**, and click **Next**.
11. Select **Allow the connection** and click **Next**.
12. Leave Domain Private and Public unchanged, and click **Next**.
13. Name the rule **JAMS Outbound** and click **Finish**.
14. The localhost is now available on the Internet. The application can now be visited through the server domain name or IP address on port 443 or 8443.

G. Create a JAMS Windows Service (Embed Tomcat Server Windows Service) to start JAMS with the server

1. In order to create a JAMS Windows Service, the **NSSM (the Non-Sucking Service Manager)** can be used. NSSM is available at <http://nssm.cc/download> and <https://github.com/kirillkovalenko/nssm>.

2. Once NSSM has successfully downloaded, open a **Command Prompt** and change the directory to:

```
nssm-2.24\win64
```

3. To install and open a graphical user interface (GUI), type:

```
nssm.exe install JAMS
```

4. In the **Path** field, specify the path to the Java executable, for example:

```
"C:\Program Files\Common Files\Oracle\Java\javapath\java.exe"
```

5. In the **Startup directory**, for the installation folder path, type:

```
"C:\jams"
```

6. In the last field, add the following arguments:

```
-classpath "C:\jams" -jar jams-launcher.jar PORT_NUMBER server.pem  
server.key
```

where **PORT_NUMBER** is the port number to use to serve the application, for example, **443** or **8443**.

7. Now the JAMS application will start with the server.

Source: <https://medium.com/@lk.snatch/jar-file-as-windows-service-bonus-jar-to-exe-1b7b179053e4>

Client guide

This tutorial has instructions on how to connect to JAMS (Jami Account Management Server) with Jami desktop (GNU/Linux, macOS, and Windows), Jami mobile (Android and iOS), and Jami web clients.

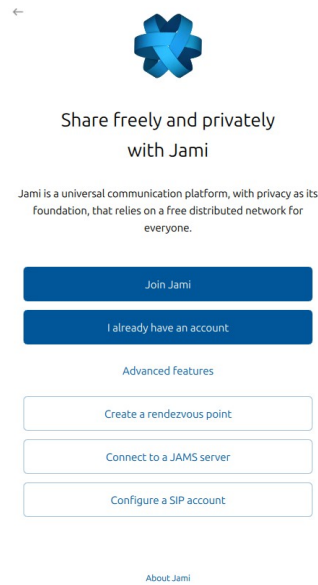
For the purposes of this tutorial, it is assumed that:

1. The server and the device attempting to connect are either:
 - On the same network, or
 - The server is publicly accessible to the outside world.
2. A valid username/password pair to connect to the server is entered.

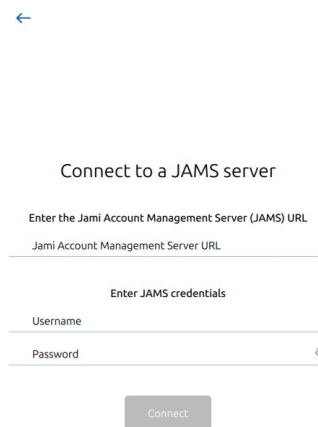
Connect with desktop clients

Jami desktop clients are available for devices with 64-bit versions of GNU/Linux, macOS, and Windows operating systems.

In Jami, open the **Add another account** page. Select **Advanced features**.



Select **Connect to a JAMS server**.

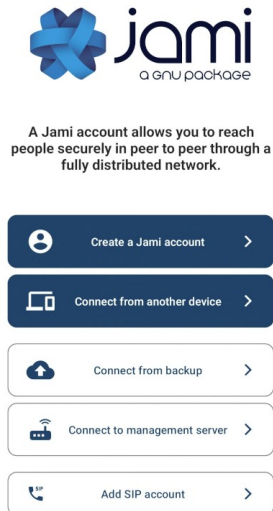


The **JAMS URL** is the DNS address of the server. The **Username** and **Password** correspond to the account. If a server has been configured with an LDAP/AD backend, the **Username** and **Password** would be the LDAP/AD username and password.

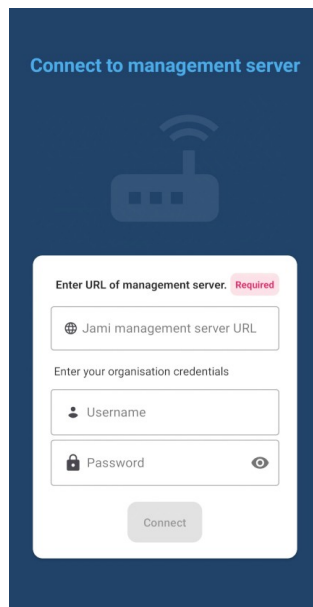
Connect with Android clients

Jami mobile clients are available for devices for smartphones, tablets, and TVs running the Android operating system.

In Jami, open the **Add another account** page.



Select **Connect to management server**.

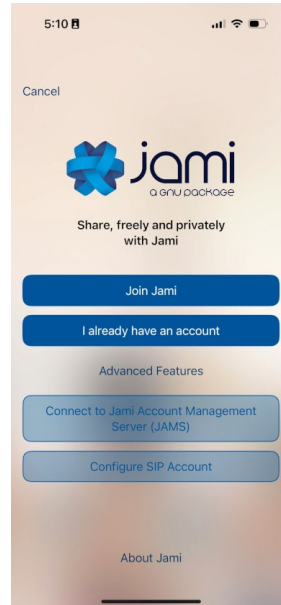


The **JAMS URL** is the DNS address of the server. The **Username** and **Password** correspond to the account. If a server has been configured with an LDAP/AD backend, the **Username** and **Password** would be the LDAP/AD username and password.

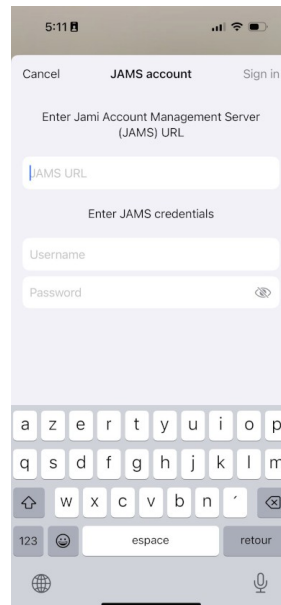
Connect with iOS clients

Jami mobile clients are available for devices for iPhones and iPads running the iOS operating system.

In Jami, open the **Add another account** page.



Select **Connect to JAMS (Jami Account Management Server)**.



The **JAMS URL** is the DNS address of the server. The **Username** and **Password** correspond to the account. If a server has been configured with an LDAP/AD backend, the **Username** and **Password** would be the LDAP/AD username and password.

Connect with web clients

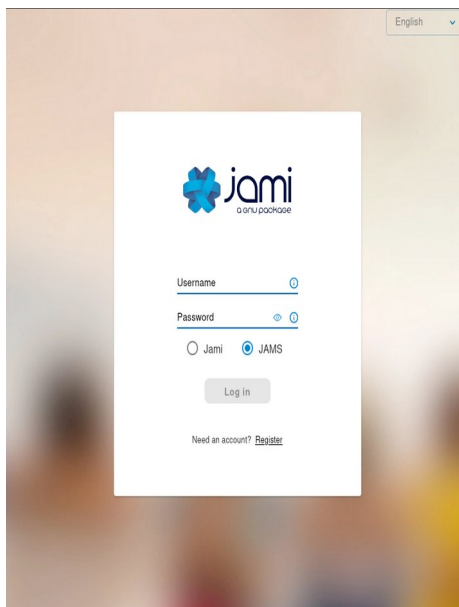
Jami web clients run on modern [web browsers](#).

In a web browser [address bar \(also location bar or URL bar\)](#), enter the JAMS URL address.

If the JAMS administrator has disabled Jami authentication, enter the **Username** and **Password**.



If the JAMS administrator has enabled Jami authentication, the **Jami** or **JAMS** authentication method is required.



Select **Log in**.



Jami is the only software required for peer-to-peer (without a server) communication that respects the freedom and privacy of its users.

Jami is the simplest and easiest way to connect with people (and devices) with instant messaging, audio, and video calls over the **Internet** and LAN/WAN **intranets**.

Jami is a free/libre, end-to-end encrypted, and private **communication platform**.

Jami is **open-source** software that **prioritizes user privacy**.

Jami has a professional-looking design and is available for a wide range of platforms. Unlike the alternatives, Jami **calls are directly between users**, as it does not use servers to handle calls.

This gives the greatest privacy, as the **distributed** nature of Jami means all calls are only between participants.

One-to-one and **group conversations** with Jami are enhanced with instant messaging, audio and video calling, recording and sending audio and video messages, file transfers, screen sharing, and location sharing.

Jami can also function as a **SIP client**.

Multiple **Jami extensions** are available: Audio Filter, Auto Answer, Green Screen, Segmentation, Watermark, and Whisper Transcript.



Jami can be easily deployed in organizations with the **JAMS (Jami Account Management Server)**, allowing users to connect with their corporate credentials or create local accounts. JAMS allows you to **manage your own** Jami community while taking advantage of Jami's distributed network architecture.

Jami is **available for** GNU/Linux, Windows, macOS, iOS, Android, Android TV, and web browsers, making Jami an interoperable and cross-platform communication framework.

Manage multiple SIP accounts, Jami accounts, and JAMS accounts with the Jami client installed on one or multiple devices.

Jami is free, unlimited, private, advertising-free, compatible, fast, autonomous, and anonymous.

Jami: <https://jami.net/>

Extensions: <https://jami.net/extensions/>

JAMS: <https://jami.biz/>

Documentation: <https://docs.jami.net/>

Videos: <https://docs.jami.net/videos/>

Mastodon: <https://mstdn.io/@Jami>

Forum: <https://forum.jami.net/>

Contribute: <https://jami.net/contribute/>

Build **IoT projects** with Jami. Re-use the universal communications technology of Jami with its portable library on your system of choice.

Jami is published under the **GPL license**, version 3 or higher.

Copyright © Savoir-faire Linux Inc.