

Axis C++ Linux Installation Guide

<!-- --> <!-- -->

1. Axis C++ Linux Installation Guide

Contents

- [Introduction](#)
- [What You Need](#)
- [Installing Axis C++](#)

Note: The Expat XML Parser module is not currently maintained and also contains some bugs. So it is removed from the 1.5 release.

2. Introduction

This guide will help you to start with Axis C++. This guide will explain the minimum steps needed to build and run Axis C++, and warn you about the common pitfalls.

3. What You Need

You need a few helper libraries for parsing XML, WSDL processing and introspection. You need to have the following in order to run Axis C++ engine.

[Apache web server](#) (2.0.x or 1.3.x) - You need to have Apache built with module so support, hence you need to have the source of Apache web server.

Axis C++ uses Apache web server to host services.

[Xerces C++ \(2.2.0\)](#) XML parser

Axis C++ needs an XML parser to parse SOAP messages and WSDD files. It has a parser abstraction layer that helps users to select/switch between parsers. However only one parser library could be used at a time. Currently Xerces parser is supported by Axis C++.

4. Installing Axis C++

4.1. 1. Install Apache Web Server

In case you have already installed Apache , make sure that 'so modules' are enabled. This is because Axis C++ server engine is implemented as a 'so module'. (For Apache 1.3.x use --enable-module=so; for Apache 2.0.x use --enable-so when configuring. See Apache web server documentation for more details)

4.2. 2. Install Xerces C++ (2.2.0)

See the Xerces parser's documentation for installation instructions.

4.3. 3. Download Axis C++

[Download Axis C++](#) source or binary distribution and extract the package.

4.4. 4. Define the Environment Variables.

```
AXISCPP_HOME="Path to Axis C++ source or binary extracted folder"
e.g. AXISCPP_HOME="/my/home/axiscpp"
AXISCPP_DEPLOY="Path to the folder where you want to install Axis C++"
e.g. AXISCPP_DEPLOY="/usr/local/axiscpp_deploy"
LD_LIBRARY_PATH="<path to parser library you
use>/lib:$AXISCPP_DEPLOY/lib:$LD_LIBRARY_PATH"
export AXISCPP_HOME AXISCPP_DEPLOY LD_LIBRARY_PATH
```

4.5. 5. Build Axis C++

Note: The following steps are for Axis C++ source downloads. Currently Ant build system is used to build Axis C++ source code. [ANT Build Guide](#) explains how you can use Ant to build Axis C++.

Note: The following steps are common to both source and binary downloads. You need global access rights to the Axis C++ deploy folder to make sure that Axis C++ works properly.

```
chmod -R 777 $AXISCPP_DEPLOY
```

4.6. 6. Configure Apache Module

Note: to execute the following steps, you may need to have **super user rights** on your machine.

Now you need to edit **httpd.conf** file in <path to Apache web server installation>/conf and add the following lines at the bottom of that file (assuming you are using Apache 2.0.x):

LoadModule axis_module modules/libaxiscpp_mod2.so

<Location /axis>

SetHandler axis

</Location>

For Apache 1.3.x LoadModule line should read as:

LoadModule axis_module libexec/libaxiscpp_mod.so

4.7. 7. Configure Server Deployment Descriptor File

Now you need the server deployment descriptor (server.wsdd) to deploy server samples you built.

There is a sample deployment descriptor called server.wsdd_linux in \$AXISCPP_DEPLOY/etc to help to deploy samples.

Edit this file to match your system settings (because the server.wsdd_linux file assumes that you have installed the binaries in /usr/local/axiscpp_deploy, you may need to change the folder names) and copy it to server.wsdd.

server.wsdd file is an XML file, and the contents are self descriptive.

4.8. 8. Set Engine Wide Settings in Configuration File

Axis C++ uses a configuration file to let the user specify preferences such as log file locations, transport and parser libs to be used and location of deployment descriptor files.

A sample configuration file named axiscpp.conf_linux is installed in \$AXISCPP_DEPLOY/etc folder. Edit this file to match your systems settings (because the axiscpp.conf_linux file assumes that you have installed the binaries in /usr/local/axiscpp_deploy, you may need to change the folder names) and copy it to axiscpp.conf

Configuration file has the following **Syntax**:

The comment character is '#'

WSDDFilePath - Path to the server wsdd file

ClientWSDDFilePath - Path to the client wsdd

LogPath - Path to the Axis C++ server log

ClientLogPath - Path to the Axis C++ client log

Transport_http - HTTP transport library

Channel_HTTP - Channel transport library

XMLParser - XML parser library

A sample **axiscpp.conf** file:

LogPath:/usr/local/axiscpp_deploy/log/AxisLog

WSDDFilePath:/usr/local/axiscpp_deploy/etc/server.wsdd

ClientLogPath:/usr/local/axiscpp_deploy/log/AxisClientLog

XMLParser:/usr/local/axiscpp_deploy/lib/libaxis_xercesc.so

Transport_http:/usr/local/axiscpp_deploy/lib/libaxis3_transport.so

Channel_HTTP:/usr/local/axiscpp_deploy/lib/libaxis3_transport_channel.so

4.9. 9. Deploying with Apache Web Server

Now we need to copy Apache module (libaxiscpp_mod2.so for Apache 2.0.x and libaxiscpp_mod.so for Apache 1.3.x) to the correct places and start Apache web server. The steps to follow are:

1. Copy libaxiscpp_mod2.so to /<your Apache 2.0.x home>/modules (or copy libaxiscpp_mod.so to /<your Apache 1.3.x home>/libexec)
2. Start Apache /<path to Apache installation>/bin/apachectl start

To do the same you can use scripts in \$AXISCPP_DEPLOY/bin.

cd \$AXISCPP_DEPLOY/bin

To deploy with Apache 2.0.x

sh deploy_apache2.sh

To deploy with Apache 1.3.x

sh deploy_apache.sh

Note: please rename libaxis_xercesc.so (the default parser library) to libaxis_xmlparser.so.

```
cp -f ${AXISCPP_DEPLOY}/lib/libaxis_xercesc.so  
${AXISCPP_DEPLOY}/lib/libaxis_xmlparser.so
```

4.10. 10. See Axis C++ in action

Now the installation is complete. You can verify that the server side is working by accessing the URL <http://localhost/axis> using your web browser. You should get the Axis C++ welcome page and this page will show you a list of deployed services as specified by the \$AXISCPP_DEPLOY/conf/server.wsdd file.

Now you can run a client sample and see if it works.

cd \$AXISCPP_DEPLOY/bin

./base

To help you run several samples at once there is a script named **run_interoptests.sh** in \$AXISCPP_DEPLOY/bin folder. You can try running that as well.

4.11. 11. Simple axis server installation

1. Build the source distribution as mentioned above.
2. Make sure that you have set the **AXISCPP_DEPLOY** environment variable to point to

your deployment folder as mentioned above

3. Copy **\$AXISCPP_DEPLOY/etc/axiscpp.conf_linux** to
\$AXISCPP_DEPLOY/etc/axiscpp.conf

and make sure that the contents of that file match your system settings

4. Run simple axis server in **\$AXISCPP_DEPLOY/bin**

Synopsis: **simple_axis_server server-port** Where server-port is the port on which you would like the server to listen for client requests.

For Example

cd \$AXISCPP_DEPLOY/bin

./simple_axis_server 9090

5. Run clients in **\$AXISCPP_DEPLOY/bin**

On a different shell:

cd \$AXISCPP_DEPLOY/bin

./base http://localhost:9090/axis/base

Similarly you could run the other samples.

[PDF](#)

[PDF](#)