

# Setting up ssh for Running and Testing NMM

**Motama GmbH, Saarbruecken, Germany**  
**(<http://www.motama.com>)**

**April 2010**

Copyright (C) 2005-2010  
Motama GmbH, Saarbruecken, Germany  
<http://www.motama.com>

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with the Invariant Sections being all sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license can be found in the file COPYING.FDL.

THE DOCUMENT IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE DOCUMENT BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE DOCUMENT OR THE USE OR OTHER DEALINGS IN THE DOCUMENT.

NMM can be used for locally running multimedia applications and for networked multimedia applications. For running networked multimedia applications, using remote login with ssh greatly simplifies starting and stopping different NMM applications, such as clic or serverregistry. This document describes how to set up ssh for NMM. We only discuss a simple setup, for further information, please see the official ssh documentation.

# 1. ssh Packages for different Operating Systems

First, you need to install the ssh packages provided for your operating system.

## 1.1. Linux

For GNU/Linux, you will find a suitable installation package for your distribution easily. Please follow the instructions provided for this package.

## 1.2. Windows

copSSH is a minimal cygwin distribution for ssh. Instead of using a full cygwin installation, we recommend using copSSH as ssh server for Windows because it is much faster and simpler to install.

### 1.2.1. Installation of copSSH

- Download from <http://www.itfix.no/copssh/> (current version is 1.4.5)
- Install (as Administrator)
- Open TCP on port 22 for own LAN only in Windows firewall.
- Select Start -> All Programs -> COPSSH -> Activate a user and follow the instructions provided.
- Log in from remote host as usual using ssh (using your account and password for Windows). You should have a bash shell after login. You can verify this by running

```
echo $SHELL
```

which should print `/bin/bash`
- In `/cygdrive/c/` you can access the C drive of Windows. In `/myhome` or `/myhome/<account>` you will find the personal Windows directories for the user logged in

### 1.2.2. Testing NMM

- Build NMM as usual
- If your NMM installation is located in `C:\<path to NMM>` you can access it in the cygwin file system at the location `/cygdrive/c/<path to NMM>`
- Log in to Windows system using ssh as described above

- Set up environment: If, for example, your NMM build is located in `c:\home\mlohse\NMM\` then setup your environment as follows, e.g. by adding following lines to your personal `.bashrc` located in `c:\Program Files\copSSH\home\<account>`

```
export PATH=/cygdrive/c/home/mlohse/NMM:/cygdrive/c/home/mlohse/NMM/winxp/dll:$PATH
export NMM_DEV_DIR="c:/home/mlohse/NMM"
```

Please make sure that you use paths in the cygwin file system for `PATH` and paths in the Windows file system for `NMM_DEV_DIR`. If you want to use backslashes, use quotes for `NMM_DEV_DIR`.

- Set up NMM locally on Windows: in ssh bash konsole:

```
./serverregistry -s
```

If you do not see any output at all (i.e. `serverregistry` simply terminates), the above described environment variables have not been set correctly.

- Test NMM on Windows: in ssh bash konsole:

```
./serverregistry
```

On other system, e.g. Linux:

```
./clic wav_play.gd -i /home/data/media/audio/wav/big.wav
```

with

```
WavReadNode
! DirectXPlaybackNode #setLocation("<windows host>")
```

You should hear the audio file read from the Linux system playing on the Windows system.

## 2. ssh Login without Password

For allowing ssh access without the need to manually type the password upon every log in, following steps have to be performed.

### 2.1. With NFS

For systems with a global file systems, e.g. exported/imported using NFS:

- Run:

```
cd .ssh
```

- Run:

```
ssh-keygen -t rsa
```

- Run:  
`cp id_rsa.pub authorized_keys`
- Run:  
`ssh-keygen -t dsa`
- Run:  
`cp id_dsa.pub authorized_keys2`

## 2.2. Without NFS

For systems without NFS: For example, if you want to login from host1 to host2.

- Run:  
`cd .ssh`
- Run:  
`host1 > ssh-keygen -t rsa`  
  
which creates the files `id_rsa` and `id_rsa.pub` in directory `.ssh`
- Add the content of `id_rsa.pub` of host1 to `.ssh/authorized_keys` on host2
- Run:  
`host1 > ssh host2`  
  
and allow to add host2 to `know_hosts`
- Run:  
`host2(ssh) > exit`  
  
i.e. log out from host2
- Try:  
`host1 > ssh host2`

This should work without any user interaction.

