Linux NETMEETING HOWTO

Brent Baccala

baccala@freesoft.org

Martin Schiffers

mschiffers@axsi.net

Mark F. Komarinski

Revision History

Revision v1.0	13 January 2001	Revised by: bwb
Initial public release		
Revision v0.11	25 October 2000	Revised by: mfk
Conversion to DocBook		

This document aims to describe how to make Microsoft NetMeeting interoperate with Linux.

Table of Contents

1. Introduction

This is the Linux NETMEETING HOWTO; it describes how to configure Linux for interoperation with Microsoft NetMeeting. The latest copy of this document is available at http://www.freesoft.org/software/NetMeeting or from the Linux Documentation Project

NetMeeting is Microsoft's client implementation of the H.323 international standard teleconferencing protocol suite, providing audio and video conferencing over an IP network. NetMeeting also implements the T.120 protocol suite, providing shared whiteboard, file transfer and application sharing. As an extension, LDAP is used for directory service. NetMeeting is included in Windows 2000 and is freely available for download from http://www.microsoft.com/windows/netmeeting for Windows 95, 98, and NT.

Linux software is presently (October 2000) available to support H.323 (both audio and video) and LDAP directory service, but not T.120 shared whiteboard, file transfer, or application sharing.

If you don't know anything about H.323, I recommend these links:

- http://www.openh323.org/
- http://www.databeam.com/h323/h323primer.html
- http://www.hut.fi/~tttoivan/index4.html
- ftp://ftp.intel.com/pub/H.323/DOCS
- http://support.intel.com/support/videophone/trial21/H323_WPR.HTM

If you don't know anything about LDAP, I recommend these links:

- http://www.openldap.org/
- http://www.umich.edu/~dirsvcs/ldap/index.html
- RFCs 2251−2256

If you have other links to recommend, or other suggestions for improving this document, please email me at `<baccala@freesoft.org>`

---

2. OpenH323

2.1. What is it?

OpenH323 is an open source implementation of the H.323 protocol suite. As such, it can directly interoperate with Microsoft NetMeeting. At the time of this writing (October 2000), OpenH323 is still early in its development cycle; buggy and in flux, but useful.

OpenH323 consists of several C++ libraries and some C++ client programs.

The most useful client programs are:

Table 1. List of client applications

ohphone	H.323 interactive client. Linux equivalent to NetMeeting. Supports audio and video; no shared whiteboard, file transfer, or shared applications
openam	H.323 answering machine. Plays back a recorded message and records incoming audio. No video support at present.
forwarder	Forwards H.323 sessions from one IP address/port to another. Used to serve multiple H.323 destinations from a single IP address.
openmcu	Multipoint Control Unit. Connects multiple sessions together into a conference call.
PSTN Gateway	Allows NetMeeting clients to make phone calls onto the conventional phone system − the Public Switched Telephone Network (PSTN). Requires special hardware.

OpenH323 presently (October 2000) supports audio codecs G.711, G.723.1, LPC−10, and GSM 06.10, as well as video codec H.261.

2.2. Why is it needed?

OpenH323 is needed only if you want to make audio/video connections with NetMeeting clients directly from your Linux system. It is not needed to provide LDAP directory service to NetMeeting clients.

2.3. Where to get it?

The main site is http://www.openh323.org/ and contains links to a download page, mirror sites, mailing lists, and other resources.

OhPhone, OpenAM, and PSTNgw are available as part of the standard distribution, in both source and executable formats. forwarder and openmcu are presently (December 2000) only available from the CVS archive, as modules named "forwarder" and "openmcu".

2.4. Installation

For OhPhone, OpenAM, and PSTNgw, download the executables. If you want to build from source, perhaps because you need forwarder or openmcu, you'll need the source code to the programs, as well as to the pwlib and openh323 libraries. Compilation instructions are available on the openh323 website.

2.5. Gatekeepers

OpenH323 doesn't provide any gatekeepers itself, but several are under construction based on its libraries. As of the end of 2000, most of them are actively under development and quite primitive. I haven't used any of them myself, but you want may to examine the following links:

- OpenGatekeeper
- OpenH323 Gatekeeper
- OpenGatekeeper H323 Proxy

---

3. NetMeeting directory kit

3.1. What is it?

Each NetMeeting client can register with an LDAP server and has a directory window that lists other NetMeeting clients registered with the same server. The NetMeeting directory kit is an extension to the OpenLDAP server that provides directory service to NetMeeting clients.

3.2. Why is it needed?

While NetMeeting can connect directly to another H.323 device by specifying an IP address or DNS name, normally you'll want to use an LDAP directory server. Using an LDAP server lets users see a directory listing of available destinations, and is required if you need to resolve aliases, for example if you want to serve multiple H.323 destinations from a single IP address. A directory server isn't required to connect directly from Linux to a NetMeeting client; use OpenH323 for this.

The NetMeeting client violates the LDAP protocol in several ways, so you'll have problems if you try using a standard LDAP server. The NetMeeting directory kit corrects for these problems and allows an OpenLDAP server to be used for NetMeeting directory service.

3.3. How it works

```
                 Block diagram of NetMeeting directory kit

___________________         _______    __________________        ______________
|    LDAP server   | request |      |   |   LDAP server   | request|            |
|                  | <-------| Perl |<--|                | <------| NetMeeting |
| on private port  |         |script|   | on public port  |       |  client    |
|  (i.e, 2345)     |-------> |      |-->|     389         |------->|            |
|                  | reply   --------   |                 |  reply --------------
|                  |                    |                 |
-------------------                     ------------------
```

The directory server consists of a 'master' LDAP server to receive requests, a Perl script to correctly interpret the Microsoft NetMeeting requests and, after interrogation of a 'hidden' LDAP server, formats the results in a way that the NetMeeting client can understand. OpenLDAP's 'shell backend' is used to call the Perl script. A custom schema is also required. The script presently handles all of the above problems, with the exception of timing out entries, which it doesn't do.

3.4. Where to get the software

First of all you need to get the OpenLDAP software.

> **Note:** Pre−built OpenLDAP software (i.e, RPMs) won't work unless configured with support for the shell backend.

You can download OpenLDAP from the main site located at ftp://ftp.OpenLDAP.org/pub/OpenLDAP/openldap–release/ or any mirror.

The NetMeeting directory kit is available from http://www.freesoft.org/software/NetMeeting/download.

You need Perl 5, available from http://www.perl.org, but already included in all common Linux distributions. You will also need the Net::LDAP module from the Perl CPAN archive, which can be downloaded and installed directly from Perl:

```
[root@y2k baccala]# perl -MCPAN -e shell

cpan shell -- CPAN exploration and modules installation (v1.58)
ReadLine support enabled

cpan> install Net::LDAP

... much output omitted ...

  /usr/bin/make install -- OK

cpan>
```

If you've never used CPAN before, you will be prompted first with a series of configuration questions. Once CPAN is configured, the Net::LDAP module will be downloaded, compiled, and installed automatically.

3.5. Installation

Building OpenLDAP will require approximately 60 MB of free disk space. Untar `open-ldap-2.0.x.tgz` and configure it.

> **Note:** Be sure to specify the shell backend function "––enable–shell"

```
bash$ ./configure --enable-shell
```

Now build and install it with:

```
bash$ make

... much output omitted ...

bash# make install
```

It will normally install under `/usr/local`:

Table 2. Directories used by OpenLDAP

`/usr/local/lib`	Shared and static libraries
`/usr/local/bin`	Client binaries for adding, deleting, and searching LDAP servers

/usr/local/sbin	Utility programs for manipulating the raw database files. Not needed for normal operation.
/usr/local/libexec	Various server programs, including the **slapd** binary
/usr/local/etc/openldap	Contains the default configuration files
/usr/local/etc/openldap/schema	The different schemas used by the LDAP servers.
/usr/local/var/...	The location of the LDAP databases (in subdirectories)
/usr/local/man/...	Documentation

Once OpenLDAP has been installed, next install the NetMeeting directory kit. Untar `ndk.tgz`. It contains these files:

Table 3. NetMeeting directory kit files

netmeeting.perl	Perl script used to correct NetMeeting protocol violations
netmeeting.schema	Custom NetMeeting schema used by the LDAP server
slapd.conf	Sample config file for the master LDAP server
slapd2.conf	Sample config file for the slave LDAP server
initialize	Shell script used once to initialize the slave LDAP database
slapd.rc	/etc/rc.d/ script
nmaddentry	Perl script to add entries to the NetMeeting directory
nmdirectory	Perl/Tk script to query the NetMeeting directory

Copy `netmeeting.perl` to the `/usr/local/libexec` directory, `netmeeting.schema` to the `/usr/local/etc/openldap/schema` directory, and copy both `slapd.conf` and `slapd2.conf` to the `/usr/local/etc/openldap` directory. Create the directory `/usr/local/var/openldap-netmeeting` to store the LDAP database, and make it world writable.

Especially if you've changed the directories, edit `slapd.conf` and `slapd2.conf` and verify their configuration settings.

You will need to run two copies of **slapd**. One uses `slapd.conf` and must be started as root, since it binds to port 389. The `-u` option can be specified to cause **slapd** to `chown` to an unprivileged user after binding the port (a wise precaution). The other **slapd** uses `slapd2.conf`, binds to an unprivileged port, and only needs sufficient privilege to write the database directory.

```
bash# /usr/local/libexec/slapd -f /usr/local/etc/openldap/slapd.conf -u nobody
bash$ /usr/local/libexec/slapd -h ldap://localhost:2345/ -f /usr/local/etc/openldap/slapd2.conf
```

You now have to initialize the slave database with a single entry. This is only done once, by running the `initialize` script included in the kit. The "rootdn" and "rootpw" entries are in the slave config file to allow access for the initialization script, and must match the `-D` and `-w` options in the script. Once you've initialized the database with a single parent entry, you can comment out the "rootdn" and "rootpw" lines from `slapd2.conf`, though this is not critical.

The server should now be up and running. For systems with `/etc/rc.d/` style initialization scripts (like RedHat), the `slapd.rc` is provided to automate the starting and stopping of the **slapd**s.

3.6. Server Security

As shown above, I run both **slapd**s as an unprivileged user, minimizing the possibility of compromised security due to a bug in either the server software or the Perl script. Of course, this requires the database directory to be world writable so that the unprivileged slave server can update it. This isn't as glaring a hole as it might first appear, since the NetMeeting clients themselves use no authentication. Thus, even if the database directory were better protected, anyone on a local or remote host could use LDAP client programs to delete or modify any of the database entries.

3.7. DNS issues with Windows 2000

Windows 2000 implements a modified DNS SRV (RFC 2782), an enhanced means of locating network servers, including LDAP. To avoid delays in connecting to an LDAP server from a Windows 2000 client, SRV records should be provided by the DNS server. ISC Bind has supported SRV records since version 8.2.2. As described in the Bind FAQ, the "check−names ignore" option is required to permit underscores in the DNS names. Basically, if your NetMeeting server name is "ils.freesoft.org", Microsoft Active Directory will expect to use a subzone called "_msdcs.ils.freesoft.org". Within this subzone, the "global catalog" server will be called "gc._msdcs.ils.freesoft.org" and its LDAP SRV record will be called "_ldap._tcp.gc._msdcs.ils.freesoft.org", as described by Microsoft. Got it? Your DNS database entry will then look something like this:

```
$ORIGIN _msdcs.ils.freesoft.org.

_ldap._tcp.gc      IN      SRV      1 1 389 ils.freesoft.org.
```

I've not yet been able to test this myself, not having a Windows 2000 system, so any feedback would be appreciated.

3.8. Interoperation with other LDAP service

The instructions above assume that your LDAP server is only being used for NetMeeting directory service. Yet what if you want to use a single server for both NetMeeting directory service and other LDAP service? Only one server can be bound to port 389, but OpenLDAP allows multiple database sections to be specified in its configuration file, each serving different parts of the LDAP namespace. NetMeeting uses only the "objectClass=RTPerson" subtree, so as long as you avoid this subtree, you can configure additional database sections to serve other subtrees with other databases. The biggest problem you are likely to encounter is the custom NetMeeting schema, which conflicts slightly with the standard schema. Since the NetMeeting schema is more liberal than the standard schema, I'd suggest commenting out the conflicting parts of the standard schema. NetMeeting clients won't work with the standard schema. See the LDAP RFCs and the OpenLDAP documentation for more information about configuring LDAP servers.

4. Using the Software

4.1. Direct Connection

You can use OpenH323's **ohphone** program to connect directly to a NetMeeting client. Specify the `-n` option to indicate that you're not using a gatekeeper, and either the DNS name or IP address of the NetMeeting client:

```
bash$ ohphone -n 208.130.48.22
```

You can also start **ohphone** to receive incoming calls from NetMeeting clients:

```
bash$ ohphone -n
```

See the **ohphone** documentation for more information on its additional features, including video conferencing, codec selection, and auto−answer.

4.2. Directory Operation

Make sure you have an LDAP server running the NetMeeting directory kit, as described above.

On the NetMeeting client, select the Tools −> Options menu item to display a configuration dialog. Under the "General" (NetMeeting 3) or "Calling" (NetMeeting 2) tab, there will be a section for "Directory Settings". Here you can enter the IP address or DNS name of the server. The client will then attach to the server and register itself either automatically, if the "Log on to directory server when NetMeeting starts" checkbox is selected. You can also log on to the directory server manually, by selecting Call −> Log on .

If the user selects Call −> Directory , a directory window will be displayed showing all users registered on the LDAP server. Double−clicking on one of the names will initiate a connection to that user.

Querying the NetMeeting LDAP server from Linux can be done, but is tricky because the client's IP address is stored in decimal, and I don't mean dotted decimal. For example, the IP address 63.216.69.197 is stored as 3309688895. Here's some Perl code to convert back and forth from the NetMeeting IP address format:

```
# Convert $addr (IP address or DNS name) to a NetMeeting decimal IP address

use Socket;
$bytestring = inet_aton($addr);
if (defined $bytestring) {
    ($sipaddress) = unpack('V', $bytestring);
} else {
    die "Can't resolve $addr\n";
}

# Convert $sipaddress (from a NetMeeting LDAP server) into dotted decimal form

$packedipaddr = pack 'V', $sipaddress;
$ipaddress = join '.', unpack('C4',$packedipaddr);
```

Included with the NetMeeting directory kit is **nmdirectory**, a simple Perl/Tk script to query a NetMeeting LDAP server and display the clients registered with it. It's very primitive, and doesn't work well with large databases, but provides a rudimentary example of how to interpret search results from a NetMeeting LDAP server.

4.3. Linking From A Web Page

Microsoft Internet Explorer understands URLs with a "callto:" scheme that specify NetMeeting destinations in one of two forms. When a link with a "callto:" URL is selected, Internet Explorer runs NetMeeting and directs it to connect to the specified destination.

The first URL form, "callto:destination", where 'destination' is either an IP address or a DNS name, causes NetMeeting to open an H.323 connection to port 1720 on 'destination'. Use this form to connect directly to another NetMeeting or OpenH323 client.

The second URL form, "callto:server/alias", causes a directory lookup on LDAP server 'server', searching for a CN attribute of 'alias'. Assuming a match is found, a connection is made to the IP address specified in the entry's sipAddress attribute. NetMeeting clients, by default, register their user's E−mail addresses in the CN attribute. Use this form to perform a directory lookup based on E−mail address.

4.4. Permanent Directory Entries

NetMeeting clients aren't the only source of LDAP directory entries. In particular, permanent directory entries can be manually inserted into the LDAP server using the OpenLDAP client tools. Assuming the attributes are specified properly, these entries will then appear in NetMeeting directory listings and can be used as targets in "callto:" URLs. This is useful when working with OpenH323 clients that don't register themselves by default with the LDAP server.

To simply creating directory entries, the **nmaddentry** script is included in the NetMeeting directory kit. Run it without arguments for a usage message. For example, if you've started **ohphone** on "y2k.freesoft.org", you can register it with the LDAP server on "ils.freesoft.org" using alias "baccala@freesoft.org" like this:

```
bash$ nmaddentry −h ils.freesoft.org baccala@freesoft.org y2k.freesoft.org
Successfully added cn=baccala@freesoft.org, objectclass=rtperson
bash$
```

This entry will now appear in NetMeeting directory listings and can be addressed as "ils.freesoft.org/baccala@freesoft.org". The entry will automatically timeout after 30 minutes. The `−p` switch creates a permanent directory listing that won't time out, but this only works on OpenLDAP servers using the NetMeeting directory kit. To remove a permanent entry, use the **ldapdelete** program included with the OpenLDAP distribution, specifying the LDAP Distinguished Name returned by **nmaddentry**:

```
bash$ ldapdelete −h ils.freesoft.org 'cn=baccala@freesoft.org,objectclass=rtperson'
bash$
```

4.5. Serving Multiple Aliases

The attributes registered by a NetMeeting client include 'sport', the TCP port number it listens on for incoming H.323 requests, but since this attribute is never retrieved in search requests, it isn't as useful as it first appears. In fact, NetMeeting always opens H.323 connections to the default port (1720), which raises the question of how to serve multiple aliases from a single IP address.

The key to doing this is the **forwarder** program, included in the OpenH323 CVS archive. **forwarder** listens for connections on port 1720, and can be configured to redirect them based on the alias being called. This allows calls for each alias to be sent to a unique port number, where a program like **ohphone** or **openam** is listening.

To use aliases, an LDAP directory is required, with an entry for each alias. Each alias entry should specify a 'cn' attribute with the alias name, and a 'sipAddress' attribute with the IP address of the host where **forwarder** is listening.

I've successfully configured a single host to act as a combination LDAP server (on port 389), **forwarder** (on port 1720), and **ohphone** and **openam** clients on various private port numbers and remote systems.

4.6. Using the Answering Machine

The OpenH323 answering machine, **openam**, will listen for incoming H.323 connections, play a pre−recorded message, and then record any audio sent to it into a file. It can optionally be configured to run another program at the end of the call, to email the recorded audio, perhaps.

It's usefulness is currently (December 2000) limited by the lack of a gatekeeper program clever enough to redirect calls to it if there's no answer at the main address. Thus, it will only act as an answering machine if the **ohphone** program is running at the main address, and has been configured to redirect calls to another address, using the `−−forward−no−answer` and `−−forward−busy` options.

4.7. Conference Calls

The **openmcu** program, in the OpenH323 CVS archive, implements an H.323 Multipoint Control Unit (MCU). Multiple NetMeeting or **ohphone** clients can connect to the MCU and form a conference call. As of December 2000, the quality and reliability of the connection is problematic, but hopefully this will improve.

4.8. Routing Calls Through NAT

Special support is required on a NAT (IP Masquerade) router to allow H.323 traffic to pass through. If the NAT router is running Linux, a masquerading module can be found at http://www.coritel.it/projects/sofia/nat.html.

> **Note:** I have not actually tested this software.

4.9. Custom Configurations

The server capabilities can be customized by modifying the 'netmeeting.perl' script. For example, calls for stale entries could be redirected to an "forwarder" configured to hand off to "openam" answering machines. Thus, calls to a unavailable user would be answered and recorded for later playback.

As OpenH323's development continues, it's expected that these techniques will become more sophisticated, for example by ringing the user first and only forwarding to an answering machine if there's no answer after a given time. Such functionality would most likely be placed in a gatekeeper.

5. Debugging

For debugging the NetMeeting directory kit Brent Baccala suggests using **ethereal** (http://ethereal.zing.org/) to do a packet trace. It's LDAP support is quite good. There is also a trace file option in the Perl script "netmeeting.perl" that can be uncommented.

You might also try running the slapds with debugging turned on (−d 768 is a good start), but their messages are rather confusing.

For debugging H.323, try using the "−t" and "−o" options, supported by all the OpenH323 client programs.

A. LDAP attributes used by NetMeeting

Distinguished Names (DNs) used by NetMeeting must always end in "objectclass=rtperson". The following LDAP attributes are used by NetMeeting:

Table A−1. NetMeeting LDAP attributes

objectClass	must be "RTPerson"
cn	alias used for directory lookups; must be present
sappid	must be "ms−netmeeting"
sprotid	must be "h323"
sprotmimetype	typically "text/h323"; unused
smimetype	typically "text/iuls"; unused
sflags	must be 1
sappguid	unknown
smodop	unknown
sipaddress	decimal IP address
sport	TCP port number; unused
ssecurity	unknown
sttl	entry timeout value in minutes
c	two digit country code
rfc822mailbox	email address
givenname	optional
surname	optional
comment	optional
location	optional
ilsa39321630	1 = personal; 2 = business; 4 = adult
ilsa32833566	0 = not audio capable; 1 = audio capable
ilsa32964638	0 = not video capable; 1 = video capable
ilsa26214430	0 = not in a call; 1 = currently in a call
ilsa26279966	unknown

NetMeeting uses a non−standard means of refreshing dynamic entries. The Microsoft server maintains an "sttl" attribute, which is a time to live for the entry in minutes. A search request for attribute "sttl" resets the timer. If the timer goes to zero, the entry is supposed to disappear from the database. Of course, the sttl attribute doesn't actually exist in the database, and the client doesn't bother to give us the whole DN it wants updated, only supplying the "cn" component in the search request.

B. NetMeeting LDAP protocol violations

As mentioned, NetMeeting violates the LDAP protocol in several ways. For the record, NetMeeting:

- Doesn't structure Distinguished Names (DNs) properly

  NetMeeting puts the most significant elements in the DN first, instead of last, using:

  ```
  C=US, O=Microsoft, CN=xxx@abc.com, OBJECTCLASS=rtperson
  ```

  instead of the proper formating, which is:

  ```
  CN=xxx@abc.com, O=Microsoft, C=US
  ```

- Doesn't include the required "objectclass" attribute

  Instead, it tacks an "OBJECTCLASS" element to the end of the DN, as shown above.
- Doesn't insert parents into the LDAP server

  This is a clear violation of the LDAP standard, which requires parents to exist before children can be created. I.e, to insert this DN:

  ```
  CN=xxx@abc.com, O=Microsoft, C=US
  ```

  this DN must already exist:

  ```
  O=Microsoft, C=US
  ```

  as must this one:

  ```
  C=US
  ```

- Doesn't understand attribute aliases, and is therefore unable to recognize that "sn" and "surname" refer to the same attribute.
- Requires that attributes in a search request be returned in exactly the same order they were requested, a requirement not guaranteed by the OpenLDAP server.
- Specifies "base" scope in search requests, when it really should use "sub", since it wants a list of entries, not just one
- Uses the "%" character as wildcard in search requests, instead of the "*" character specified by the standard.

* Uses a non−standard means of refreshing dynamic entries.

  The Microsoft server maintains an "sttl" attribute, which is a time to live for the entry in minutes. A search request for attribute "sttl" resets the timer. If the timer goes to zero, the entry is supposed to disappear from the database. NetMeeting 2 supplies an "sttl" attribute, but NetMeeting 3 doesn't actually create the "sttl" attribute at all. Also, the client doesn't bother to give us the whole DN it wants updated, only supplying the "cn" component.

C. Interoperation with Cisco

Both NetMeeting and OpenH323 can interoperate with Cisco's voice capable routers. To successfully initiate calls from a Cisco to an OpenH323 (i.e, Linux) client, the G.711 codec must be explicitly specified. For example, with the following configuration, dialing "911" on the Cisco will place a call to a Linux system (10.1.1.1) running OpenH323:

```
dial-peer voice 911 voip
 destination-pattern 911
 session target ipv4:10.1.1.1
 codec g711ulaw
```

To call from Linux to a Cisco, use **ohphone** with a `number@host` argument. `number` should be a phone number that's been configured on the Cisco using a **dial−peer** statement. For example, this will call number "111" on a Cisco (10.1.1.10):

```
bash$ ohphone -n 111@10.1.1.10
```

To call from NetMeeting to a Cisco, select the Cisco as a gateway. To do this from NetMeeting, select Tools−>Options. For NetMeeting 2, select Audio, check the box labeled "Use H.323 gateway", and enter the Cisco's DNS or IP address. For NetMeeting 3, select General+Advanced Calling..., check the box labeled "Use a gateway..." (not gatekeeper) and enter the Cisco's address. Now, you can type a phone number directly into NetMeeting's address panel and it will be relayed to the Cisco and resolved there, using the Cisco's configured dialing rules. If you're using NetMeeting 2, you'll need to select "H.323 Gateway" from the "Call using:" list when you initiate the call.

D. Thanks

Many thanks have to go to Brent Baccala, who wrote the NetMeeting directory kit, also for his 24−hour E−mail tech support, and encouragement. Without him I would have passed a many nights more to set it up at my own.