ATM on Linux HOWTO

Paul B Schroeder

IBM Corporation

paulsch@us.ibm.com

ATM support for Linux is currently in pre−alpha stage. There is an experimental release, which supports raw ATM connections (PVCs and SVCs), IP over ATM, LAN emulation, MPOA, Arequipa, and some other goodies.

2001−10−18

Revision History

Revision 2.4.0	2001−10−18	Revised by: PBS
Converted from LaTeX to DocBook along with some other additions and changes.		

This document describes how to install, setup, and configure the necessary drivers and tools to support ATM networking under Linux.

For the latest information, please check the *ATM on Linux* home page.

Table of Contents

1. Introduction

1.1. Acknowledgements and Thanks

This document is largely derived from the *Usage Instructions* document that was included with the *ATM on Linux* distribution up until version 0.79. That previous document was written by Werner Almesberger `<wa@almsesberger.net>` while he was at the Institute for computer Communications and Applications (ICA).

The section *Running Two ATM NICs Back−to−Back* was primarily written by Richard Jones `<rjones@imcl.com>`.

1.2. Copyright

1.3. Mailing List

There is also a mailing list on which to discuss ATM on Linux. If you have any comments, questions, suggestions, or would just like to get involved, please join the list. You can *subscribe* and *unsubscribe* to it at http://lists.sourceforge.net/lists/listinfo/linux−atm−general.

The mailing list is archived at http://www.geocrawler.com/lists/3/SourceForge/6487/0/.

1.4. CVS Access

Users are encouraged to continue to use the releases instead of automatically assuming they should grab the latest version out of CVS. However, if you like living on the edge, here is how to do it.

First, log in anonymously:

```
% cvs −d:pserver:anonymous@cvs.linux−atm.sourceforge.net.:/cvsroot/linux−atm login
```

Just hit return when prompted for a password. Then, checkout the repository:

```
% cvs -z6 -d:pserver:anonymous@cvs.linux-atm.sourceforge.net.:/cvsroot/linux-atm co -P linux-atm
```

You may also specify a branch to check out specifically:

```
% cvs -z6 -d:pserver:anonymous@cvs.linux-atm.sourceforge.net.:/cvsroot/linux-atm co -r V2_5_0 linux-
```

In either case, this will create a directory called "linux−atm" with the latest sources in it. When working inside this directory you will not need to specify the '−d' option to CVS. For instance, you could just do

```
% cvs -z6 up -d
```

To grab any changes that have been put in the repository (the '−d' option in the above example is to the "up" sub−command and is different than the '−d' used to specify the CVS root directory)

After you have checked out the source tree, you will need to run the autotools script in the top level directory before you can configure, build, and install from that source tree:

```
# ./autotools
Running aclocal...
Running autoconf...
Running autoheader...
Running automake...
automake: configure.in: installing `./install-sh'
automake: configure.in: installing `./mkinstalldirs'
automake: configure.in: installing `./missing'
configure.in: 26: required file `./ltconfig' not found
automake: Makefile.am: installing `./INSTALL'
automake: configure.in: installing `src/lane/ylwrap'
Finished...  Now run './configure' and 'make'...
```

If you wish to create a tarred, gzipped distribution file or a RPM distribution file, run **`make dist`** or **`make rpm`** respectively. The tarred, gzipped file will be placed in the top level of the source tree and the RPM file will be placed in the `src/extra/RPMS` directory.

The CVS archive may also be browsed on the web at:
http://cvs.linux−atm.sourceforge.net/cgi−bin/viewcvs.cgi/linux−atm/linux−atm/.

Finally, if you would like to receive email including every diff that is committed to the repository as they go in, there is a mailing list called "linux−atm−commits":
http://lists.sourceforge.net/lists/listinfo/linux−atm−commits.

This mailing list should be treated as receive−only. NO discussion or questions are allowed (even of patches which are sent through that list). All discussion should be kept on the linux−atm−general mailing list.

---

2. Installation

In order to install this package, you'll need

- the package itself from http://linux−atm.sourceforge.net/dist.php
- the Linux kernel, version 2.4.x, e.g. from ftp://ftp.kernel.org/pub/linux/kernel/v2.4/
- Perl, version 4 or 5
- if you want memory debugging: MPR, e.g. from ftp://ibiblio.org/pub/Linux/devel/lang/c/

2.1. The Binary RPMs

If you do not wish to futz with extracting and building the source yourself, the ATM tools are also distributed in RPM format. The RPM can be installed as follows:

```
rpm −ivh linux−atm−x.x.x−x.rpm
```

2.2. The Source Tree

First, extract the ATM on Linux distribution:

```
tar xzvf linux−atm−x.x.x.tar.gz
```

When extracted the distribution will create the `linux−atm−x.x.x/` directory with several sub−directories. The following sub−directories are of note:

doc/

Documentation (including this HOWTO) in SGML DocBook format

src/sigd/

UNI 3.0, UNI 3.1, and UNI 4.0 signaling demon: atmsigd

src/saal/

Signaling AAL library (SSCOP, SSCF, and SAAL)

src/qgen/

Q.2931−style message handling

src/ilmid/

ILMI address registration demon: ilmid

src/maint/

ATM maintenance programs: atmaddr, atmdiag, atmdump, atmloop, atmtcp, enitune, esi, sonetdiag, saaldump, and zntune

src/test/

Test programs: align, aping, aread, awrite, br, bw, isp, ttcp_atm, window

src/arpd/

ATMARP tools and demon: atmarp, atmarpd

src/led/

LAN Emulation demon: zeppelin

src/lane/

LAN Emulation servers: bus, lecs, les

src/mpoad/

Multi−Protocol Over ATM demon: mpcd

src/debug/

Debugging tools: delay, ed, encopy, endump, svctor, zndump, and znth

src/lib/

Libraries for applications and demons

src/man/

Miscellaneous man pages

src/extra/

Extra packages and RPM spec files.

src/config/

Configuration and rc file examples

src/switch/

Switch fabric control (under construction)

2.3. Kernel Configuration

NOTE

If you are not familiar with building and installing a new kernel, please see the *The Linux Kernel HOWTO*

After unpacking the kernel distribution, do the usual **make config**, **make menuconfig**, or **make xconfig** in the top−level of your Linux kernel source tree. First, enable

```
Prompt for development and/or incomplete code/drivers
  (CONFIG_EXPERIMENTAL)
```

You should then be able to find the following options:

```
Asynchronous Transfer Mode (ATM, EXPERIMENTAL) (CONFIG_ATM)
  Use "new" skb structure (CONFIG_ATM_SKB)
  Classical IP over ATM (CONFIG_ATM_CLIP)
    Do NOT send ICMP if no neighbour (CONFIG_ATM_CLIP_NO_ICMP)
  LAN Emulation (LANE) support (CONFIG_ATM_LANE)
    Multi−Protocol Over ATM (MPOA) support (CONFIG_ATM_MPOA)
ATM over TCP (CONFIG_ATM_TCP)
Efficient Networks ENI155P (CONFIG_ATM_ENI)
  Enable extended debugging (CONFIG_ATM_ENI_DEBUG)
  Fine−tune burst settings (CONFIG_ATM_ENI_TUNE_BURST)
    Enable 16W TX bursts (discouraged) (CONFIG_ATM_ENI_BURST_TX_16W)
    Enable 8W TX bursts (recommended) (CONFIG_ATM_ENI_BURST_TX_8W)
    Enable 4W TX bursts (optional) (CONFIG_ATM_ENI_BURST_TX_4W)
    Enable 2W TX bursts (optional) (CONFIG_ATM_ENI_BURST_TX_2W)
    Enable 16W RX bursts (discouraged) (CONFIG_ATM_ENI_BURST_RX_16W)
    Enable 8W RX bursts (discouraged) (CONFIG_ATM_ENI_BURST_RX_8W)
    Enable 4W RX bursts (recommended) (CONFIG_ATM_ENI_BURST_RX_4W)
    Enable 2W RX bursts (optional) (CONFIG_ATM_ENI_BURST_RX_2W)
ZeitNet ZN1221/ZN1225 (CONFIG_ATM_ZATM)
  Enable extended debugging (CONFIG_ATM_ZATM_DEBUG)
  Enable usec resolution timestamps (CONFIG_ATM_ZATM_EXACT_TS)
IDT 77201 (NICStAR) (CONFIG_ATM_NICSTAR)
  Use suni PHY driver (155Mbps) (CONFIG_ATM_NICSTAR_USE_SUNI)
  Use IDT77015 PHY driver (25Mbps) (CONFIG_ATM_NICSTAR_USE_IDT77105)
Madge Ambassador (Collage PCI 155 Server) (CONFIG_ATM_AMBASSADOR)
  Enable debugging messages (CONFIG_ATM_AMBASSADOR_DEBUG)
Madge Horizon [Ultra] (Collage PCI 25 and Collage PCI 155 Client)
  Enable debugging messages (CONFIG_ATM_HORIZON_DEBUG)
Interphase ATM PCI x575/x525/x531 (CONFIG_ATM_IA)
  Enable debugging messages (CONFIG_ATM_IA_DEBUG)
```

The burst settings of the ENI driver can be fine−tuned. This may be necessary if the default settings lead to buffer overruns in the PCI chipset. See the on−line help on "CONFIG_ATM_ENI_TUNE_BURST" for a detailed discussion of the implications of changing the burst settings.

Note that the file `drivers/atm/nicstar.h` contains a few configurable settings for the IDT 77201 driver.

Some drivers can also be used with certain compatible cards. The latest information about compatible cards can be found at *ATM on Linux* information page.

Then build your kernel and reboot.

2.4. Driver Messages

If you've configured the ENI155p−MF driver, you should see two lines like these (512kB for the −C version, 2048kB for the −S version.):

```
eni(itf 0): rev.0,base=0xff400000,irq=10,mem=512kB (00-20-EA-00-07-56)
eni(itf 0): FPGA,MMF
```

If you've configured the ZN1221/ZN1225 driver, you will get something like:

```
zatm(itf 0): rev.3,base=0xf800,irq=11,mem=128kB,MMF (00-20-D4-10-2A-80)
zatm(itf 0): uPD98401 0.5 at 30.024 MHz
zatm(itf 0): 16 shapers, 32 pools, 2048 RX, 3958 VCs
```

Note that your board needs to be at least at revision level 3 if you want to use it in a Triton−based system.

Note that if you've configured only the ATM over TCP driver, there are no messages at startup, because ATM over TCP devices are created later using the **atmtcp** command.

2.5. Memory Debugging

If you want to enable debugging for options for memory allocations, you need to install MPR before compiling the ATM tools.

If you chose to download the binary RPM package, you can install MPR like so:

```
rpm -ivh mpr-x.x-x.rpm
```

If you chose to download the source, extract `mpr-x.x.tar.gz` like so:

```
tar xzvf mpr-x.x.tar.gz
```

Then do:

```
cd mpr-x.x
./configure x86-linux
make
make install
```

Detection of some general mis−use of `malloc` and `free` is automatically performed if the program was compiled with MPR present. Tracing of allocations is enabled by setting `MPRPC` and `MPRFI`. See `doc/mpr.html` or `doc/mpr.ps` in the MPR distribution for details.

Only little run−time overhead is incurred if memory debugging is included, but those environment variables are not set.

2.6. ATM Tools

Now, as the final step, configure and build the ATM tools. Configuration is only necessary if your switch uses UNI 3.1 or 4.0, or if it has certain bugs. The configuration options selected by passing the appropriate options to the **./configure** script in the linux−atm distribution.

NOTE

Issue **./configure −−help** from the top−level directory of the linux−atm distribution to view all possible options.

The ATM tools are built with the following commands:

```
cd linux-atm-x.x.x
./configure
make
make install
```

Unless otherwise specified when invoking **./configure**, **make install** will install executables in the directory `/usr/local/bin` and `/usr/local/sbin`, respectively. Configuration files (except for `hosts.atm` which is installed in `/etc`) are installed in `/usr/local/etc`. Libraries and header files are installed in `/usr/local/lib` and `/usr/local/include`, respectively. Man pages are installed in `/usr/local/man`.

2.7. Extra Packages

Some programs are based on large packages that are already distributed outside of the ATM context. For some packages, patches are contained in the ATM on Linux distribution. They are contained in the `src/extra` directory of the ATM on Linux distribution.

Currently, the following extra packages are available:

tcpdump

dumps network traffic (enhanced for ATM)

ANS

ATM name server (based on named 4.9.5)

Note that text2atm automatically uses ANS if available, so ans only needs to be installed on systems providing name server functionality or if ATM−aware maintenance tools nslookup, etc.) are needed.

A script hosts2ans.pl to convert a `/etc/hosts.atm` file to ANS zone files are provided in the `src/extra/ANS/` directory. Its use is described at the beginning of the file.

3. Device Setup

This section describes device−specific configuration operations, and general diagnostic procedures at the ATM or SONET level. Please see the adapter documentation for details on hardware installation and diagnosis.

3.1. ATM Over TCP Setup

If you have no real ATM hardware, you can still exercise the API by using the ATM over TCP ``driver''. It emulates ATM devices which are directly wired to remote devices (i.e. there is no VPI/VCI swapping).

To establish one (bidirectional) ``wire'', become root on both systems (or run both sides on the same system to create two connected ``interfaces'') and run the following command on one of them (let's call it ``a''):

```
# atmtcp virtual listen
```

Then, on the other system (``b''), run

```
# atmtcp virtual connect address_of_a
```

Both **atmtcp**s will report on their progress and the kernel should display messages like:

```
Link 0: virtual interface 2
Link 1: incoming ATMTCP connection from 127.0.0.1
```

and

```
Link 0: virtual interface 3
Link 1: ATMTCP connection to localhost
```

on the two systems. Note that **atmtcp** keeps running and that interrupting it breaks the virtual wire.

Multiple ``wires'' can be attached to the same machine by specifying a port number (default is 2812). Note that no AAL processing is performed. It is therefore not possible to receive data using a different AAL (e.g. AAL0) than the one with which the data was sent.

3.2. ZN1221/ZN1225 Tuning

The ZeitNet ZN1221 and ZN1225 adapters use pre−allocated pools of free memory buffers for receiving. Whenever a VC with a certain maximum SDU size is opened for receiving, the corresponding pool is filled with free buffers by the device driver. The adapter removes buffers while it receives data. When the number of remaining buffers falls below a certain threshold, the device driver replenishes the pool again.

The lower and the upper limits for the number of free buffers, and the threshold for adapting to a new data offset (see below for details), can be set using the zntune program. Usage:

zntune [−l *`low_water`*] [−h *`high_water`*] [−t *`threshold`*] *`itf`* [*`pool`*]

The changes are applied to all pools if no pool number is specified. Pool 2 stores 64 bytes packets, pool 3 stores 128 bytes packets, etc. Pools 0 and 1 are currently unused.

The current settings and some usage statistics can be obtained by invoking **zntune** without specifying new parameters:

zntune [−z] *`itf`* [*`pool`*]

The ``Size'' column shows the buffer size in Bytes. The ``Ref'' column shows the number of open VCs using that pool. The ``Alarm'' column shows how many times the number of free buffers has fallen below the low−water mark since the counters were reset. Similarly, the ``Under'' column shows how many times an incoming PDU had to be discarded because the corresponding pool was empty.

The columns ``Offs'', ``NxOf'', ``Count'' and ``Thres'' show the alignment adaption status. ``Offs'' is the offset of user data the driver currently expects in incoming PDUs. For single−copy, receive buffers are aligned accordingly so that data is received at page boundaries. ``NxOf'' is the user data offset of the most recently received PDU, where the offset differs from the currently assumed offset. ``Count'' is the number of PDUs that have been received in sequence with an offset of ``NxOf''. Finally, ``Thres'' is the threshold value ``Count'' has to reach for ``NxOf'' to become the new current offset.

Use the *`−z`* option to reset the ``Alarm'' and ``Under'' counters.

3.3. Files in `/proc/net/atm/`

Some status information about the ATM subsystem can be obtained through files in `/proc/net/atm/`. The file `/proc/net/atm/arp` contains information specific to Classical IP over ATM, see section *CLIP*.

All active ATM devices are listed in `/proc/net/atm/devices`. For each device, the interface number, the type label, the end system identifier (ESI), and statistics are shown. The statistics correspond to the ones available via atmdiag.

Individual ATM devices may register entries of the form *`type:number`* (e.g. `eni:0`) which contain device−specific information.

The files `/proc/net/atm/pvc` and `/proc/net/atm/svc` list all PVC and SVC sockets. For both types of sockets, the interface, VPI and VCI numbers are shown. For PVCs, this is followed by the AAL and the traffic class and the selected PCR for the receive and the transmit direction. For SVCs, the SVC state and the address of the remote party are shown. SVCs with the interface number 999 are used for special control purposes as indicated in the ``State'' column.

Furthermore, `/proc/net/atm/vc` shows buffer sizes and additional internal information for all ATM sockets.

3.4. ATM Diagnostics

Various counters of the ATM device drivers can be queried with the atmdiag program. See the corresponding man page for details.

3.5. SONET Diagnostics

The SONET diagnostics tool can be used to monitor link performance and to simulate errors. In order to get current SONET statistics, run it with the ATM interface number as the argument, e.g.

```
% sonetdiag 0
```

The counters can be reset with the `-z` option:

```
# sonetdiag -z 0
```

The following network failures can be simulated:[1]

sbip

insert section errors (B1)

lbip

insert line errors (B2)

pbip

insert path errors (B3)

frame

force (RX) frame loss

los

insert loss of signal

lais

insert line alarm indication signal

pais

insert path alarm indication signal

hcs

insert header checksum errors

A failure is enabled by adding the corresponding keyword on the command line. The failure is cleared by prefixing the keyword with a minus sign, e.g.

```
a# sonetdiag -z 0 >/dev/null
b# sonetdiag -z 0 >/dev/null
a# sonetdiag 0 los
a# sonetdiag 0 -los
b# sonetdiag 0 | grep BIP
Section BIP errors:      56200
Line BIP errors:           342
Path BIP errors:           152
a# sonetdiag 0 | grep FEBE
Line FEBE:                 342
Path FEBE:                 152
```

If any diagnostic error insertions are active, their keywords are shown when sonetdiag is used to obtain statistics. Note that some error insertions may be automatically switched off by the hardware.

4. Native ATM PVCs

PVCs can be used for machines that are either connected back to back or via a switch. In the latter case, the cell forwarding has to be manually set up at the switch.

4.1. Traffic Tools

aread/awrite and br/bw are simple programs to access the ATM API. awrite sends the text string passed as its second argument in an AAL5 PDU. aread receives one AAL5 PDU and displays it in hex. Both programs also display the return values of the corresponding system calls and the current values of errno.

bw either sends its standard input or a stream of blocks containing arbitrary data (if a number is passed as its fourth argument) in 8 kB AAL5 PDUs. br receives AAL5 PDUs and writes them to standard output.

The first argument of aread, awrite, br and bw is always the PVC address, i.e. the ATM interface number, the VPI and the VCI number, with a dot between elements. The interface number can be omitted if it is zero. Example:

```
% awrite 1.0.42 hi
```

Note that some adapters only support VPI == 0. Also, the VCI range may be limited, e.g 0 to 1023. The interface number can be obtained from the initialization message the driver printed during startup. atm0 is interface 0, atm1 is interface 1, etc. If the system is equipped with a real ATM adapter (e.g. not only atmtcp), that adapter is normally at atm0.

aping receives and sends small AAL5 PDUs on a PVC. It expects that messages it sends are either echoed back or that a similar program on the other side generates a stream of messages. aping reports an error if no messages are received for too long. aping is invoked by specifying the PVC, like aread.

For "real" tests, you should use the modified version of ttcp that comes with this package. The original is available at ftp://ftp.sgi.com/sgi/src/ttcp/. The following options have been added:

`-a`

> use native ATM instead of UDP/TCP. The address must be in the format `[itf.]vpi.vci` for PVCs, or a valid ATM end system address for SVCs.

`-P num`

> use a CBR connection with a peak cell rate of `num` cells per second. Default is to use UBR.

`-C`

> disable (UDP) checksums

Example:

```
%a ttcp_atm -r -a -s 0.90
```

```
%b ttcp_atm -t -a -s 0.90
```

4.2. Direct Cell Access

On adapters where the device driver supports access to raw cells (``AAL0"), individual cells can be composed and received with the atmdump program. Here is an example:

```
a% sleep 10; date | ./atmdump -t 1 -c 0.51
b% ./atmdump 0.51
825079645.192480: VPI=0 VCI=51, GFC=0x0, CLP=1, Data SDU 1 (PTI 1)
   46 72 69 20 46 65 62 20 32 33 20 31 32 3a 34 37
   3a 32 35 20 47 4d 54 20 31 39 39 36 0a 00 00 00
   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

5. Signaling

5.1. ATM Hosts File

Because ATM addresses are inconvenient to use, most ATM tools also accept names instead of numeric addresses. The mapping between names and numbers is defined in the file `/etc/hosts.atm`. The structure of this file is similar to the `/etc/hosts` file:

```
numeric_address name(s)
```

e.g.

```
47.0005.80FFE1000000F21A26D8.0020EA000EE0.00 pc2-a.fqdn pc2-a
47.0005.80FFE1000000F21A26D8.0020D4102A80.00 pc3-a.fqdn pc3-a
```

The numeric address can be specified in any of the formats described in [api]. The numeric address(es) of a Linux system can be determined with the command **atmaddr -n** (see also section *Manual Address Configuration*).

Many ATM tools also attempt to find the corresponding name when displaying an address. When translating from the numeric form to a name, the first applicable name in the file is used.

In addition to ATM addresses for SVCs, also PVC addresses can be stored in `/etc/hosts.atm`. If different address types are stored under the same name, the first suitable one will be chosen, i.e. if an application explicitly requests only SVC addresses, any PVC addresses will be ignored.

5.2. ANS

If you have access to the ATM Name Service (ANS, e.g because you've installed the ANS extension), you can use it instead of or in addition to the hosts file by specifying the host that runs ANS in the `/etc/resolv.conf` file.

For performing reverse lookups of E.164 addresses, the list of telephony country codes needs to be known. That list can be obtained from the International Telecommunications Union. The *List of ITU−T Recommendation E.164 Assigned Country Codes* is currently available in PDF and Word document formats.

NOTE

Should the URL become out of date, the document should easily be found by searching for the document's title at the ITU web site.

The script **src/lib/pdf2e164_cc.pl** in the atm−linux distribution can be used to create the E.164 county codes table with the PDF version of the country code list, e.g.

```
perl pdf2e164_cc.pl e164_xxx.pdf >/etc/e164_cc
```

It should be noted that pdftotext needs to be available in order to run the script above. It can be obtained with xpdf.

5.3. Signaling Demon

Man pages: atmsigd(8) atmsigd.conf(4)

Note that atmsigd's support for point−to−multipoint is very limited: only operation as a single leaf of a point−to−multipoint tree works.

By default, atmsigd is configured to conform to dynamically configure the UNI version. It can be compiled for UNI 3.0, 3.1, or 4.0 specifically by passing the *`--with-uni=VERSION`* to the **./configure** script in the top−level directory of the linux−atm source distribution.

Note that atmsigd is configured to be paranoid. If it detects unusual problems, it frequently terminates. This will (obviously) change in the future.

atmsigd also looks for a configuration file at the location specified with the *`-c`* option. The default location is `/usr/local/etc/atmsigd.conf`.

5.4. ILMI Demon

ILMI provides a mechanism for automatic address configuration. If there is no switch or if the switch doesn't support ILMI, the ATM addresses must be configured manually (see section *Manual Address Configuration*). Note that the ILMI demon should not be used on interfaces where addresses are manually configured.

The ILMI demon is started as follows:

ilmid [−b] [−d] [−i *`local_ip`*] [−l *`log_file`*] [−q *`qos`*] [−u *`uni_version`*] [−v] [−x] [*`itf`*]

`-b`

background. Run in a forked child process after initializing.

`-d`

enables debugging output. By default, ilmid is very quiet.

`-i local_ip`

IP address to tell switch when asked for one. Can be in either dotted decimal or textual format. By default, ilmid uses some heuristics to select a local IP address.

`-l logfile`

write diagnostic messages to the specified file instead of to standard error. The special name *syslog* is used to send diagnostics to the system logger.

−q qos

configures the ILMI VC to use the specified quality of service. By default, UBR at link speed is used on the ILMI VC.

−u uni_version

set UNI version. Possible values are *3.0*, *3.1*, and *4.0*. The dot can be omitted. The default value depends on how ilmid was compiled. Typically, it is *3.0*.

−v

enables extensive debugging output.

−x

disable inclusion of variable bindings in the ColdstartTrap. Some switches (e.g. the LS100) only work if this option is set.

If no interface number is specified, ilmid serves interface 0. You can check whether address registration was successful with the **atmaddr** command (see below).

The agent supports only the address registration procedures specified in section 5.8 of the ATM Forum's UNI 3.1 specification. These procedures involve the switch registering the network prefix on the host and the host registering the final ATM address back on the switch. The host accomplishes this by appending an ESI (End System Identifier) and a null selector byte to the network prefix registered by the switch. The ESI is the physical or MAC address of the ATM interface.

5.5. Manual Address Configuration

If your switch doesn't support ILMI, you have to set the ATM address manually on the switch and on the PC(s). On the Linux side, make sure that ilmid doesn't interfere, then use the **atmaddr** command to set the address(es).

Man pages: atmaddr(8)

Manual configuration of ATM addresses on the switch depends on the brand. On a Fore ASX−200, it can be done with the following command:

```
conf nsap route new nsap_addr 152 port vpi
```

e.g.

```
conf nsap route new 47000580ffe1000000f21510650020ea000ee000 152 1a2 0
                    |<---- NSAP prefix ----->||<--ESI--->|^^
```

```
                                                        SEL
```

The entire NSAP address always has to have a length of 40 digits. Note that you can also use addresses with a different prefix and an ESI that doesn't correspond to any ESI your adapters have. The value of the selector byte (SEL) is ignored.

5.6. Running Two ATM NICs Back−to−Back

It is also possible to run with two ATM NICs connected back−to−back, and no switch in between. This is great for simple test environments.

First, if you're using UTP or STP−5, you need a suitable cable. Our experience with standard 100Base−T back−to−back cables was not good. It appears that the pin−out they use is different. After some false starts, we found that the following cable works:

```
RJ45                            RJ45
   1        ------------        7
   2        ------------        8

   7        ------------        1
   8        ------------        2

Pins 3, 4, 5, 6 unconnected.
```

A better way to illustrate this may be to show the proper color schemes for the RJ45 connectors at each end of the back−to−back cable. The first connector should use the following scheme:

```
RJ45-1
   1 - Brown
   2 - White/Brown
   3 - Unconnected
   4 - Unconnected
   5 - Unconnected
   6 - Unconnected
   7 - Orange
   8 - White/Orange
```

And the second connector should use this scheme:

```
RJ45-2
   1 - Orange
   2 - White/Orange
   3 - Unconnected
   4 - Unconnected
   5 - Unconnected
   6 - Unconnected
   7 - Brown
   8 - White/Brown
```

You can also make up a loopback cable with 1 −− 7 and 2 −− 8 connected for ultra−cheap setups.

Here we have two machines called ``virgil'' and ``nestor''. Substitute your own names as necessary.

One side of the ATM connection needs to use the network version of atmsigd and the other side should use the normal user version. So here on nestor we start atmsigd with:

```
atmsigd -b -m network
```

and on virgil with:

```
atmsigd -b
```

Without a switch, you won't be able to use ILMI. Instead, create a `/etc/hosts.atm` file containing two dummy addresses. Our ATM hosts file contains:

```
47.0005.80FFE1000000F21A26D8.0020EA000EE0.00    nestor-atm
47.0005.80FFE1000000F21A26D8.0020D4102A80.00    virgil-atm
```

These are completely spurious addresses, of course, but as long as you're not connected to a public or private ATM network, I don't think it matters. To set the address correctly in the driver, we use:

```
atmaddr -a virgil-atm
```

on virgil, and:

```
atmaddr -a nestor-atm
```

on nestor. Now start atmarpd on both machines in the normal way. Now you (should) have a working ATM set−up. To get IP over ATM working, just follow the instructions in section *IP Over ATM*.

5.7. Q.2931 Message Dumper

The Q.2931 message compiler also generates a pretty−printer for Q.2931 messages. The executable is called q.dump is stored in the `src/qgen` directory. Note that it is not copied elsewhere by **make install**.

q.dump expects a sequence of whitespace−separated hex bytes at standard input and outputs the message structure if the message can be parsed. Example:

```
% echo 09 03 80 00 05 5A 80 00 06 08 80 00 02 81 83 00 48 \
  00 00 08 | ./q.dump
_pdsc = 9 "Q.2931 user-network call/connection control message"
_cr_len = 3
call_ref = 8388613 (0x800005)
msg_type = 0x5a "RELEASE COMPLETE"
_ext = 1
_flag = 0 "instruction field not significant"
_action_ind = 0 "clear call"
msg_len = 6 (0x6)
  _ie_id = 0x08 "Cause"
```

```
    _ext = 1
    cause_cs = 0 "ITU-T standardized"
    _flag = 0 "instruction field not significant"
    _action_ind = 0 "clear call"
    _ie_len = 2 (0x2)
      _ext = 1
      location = 1 "private network serving the local user"
      _ext = 1
      cause = 3 "no route to destination"
```

6. IP Over ATM

IP over ATM is supported with Classical IP over ATM (CLIP, defined in RFC1577 [RFC1577], LAN Emulation (LANE, defined in [lanev1] and [lanev2]) and Multi−Protocol Over ATM (MPOA, client only, defined in [mpoav1]).

6.1. CLIP

A demon process is used to generate and answer ARP queries. The actual kernel part maintains a small lookup table only containing partial information.

Man pages: atmarpd(8), atmarp(8)

atmsigd and ilmid must already be running when atmarpd is started. Use the *−b* option to make sure they're properly synchronized, e.g.

```
#!/bin/sh
atmsigd −b
ilmid −b
atmarpd −b
...
```

works, but

```
#!/bin/sh
atmsigd &
ilmid &
atmarpd &
...
```

frequently doesn't (yet).

The atmarp program is used to configure ATMARP. First, you have to start atmsigd, ilmid, and atmarpd, then create an IP interface and configure it:

```
# atmarp −c interface_name
# ifconfig atm0 local_address possibly_more_options up
```

e.g.

```
# atmarp −c atm0
# ifconfig atm0 10.0.0.3 up
```

If only PVCs will be used, they can now be created with a command like

```
# atmarp −s 10.0.0.4 0.0.70
```

NULL encapsulation is used if the `null` keyword is specified. Note that ARP requires LLC/SNAP encapsulation. NULL encapsulation can therefore only be used for PVCs.

When using SVCs, some additional configuration work may be necessary. If the machine is acting as the ATMARP server on that LIS, no additional configuration is required. Otherwise, the ATM address of the ATMARP server has to be configured. This is done by creating an entry for the network address with the option *arpsrv* set, e.g.

```
# atmarp -s \
  10.0.0.0 47.0005.80.ffe100.0000.f215.1065.0020EA000756.00 \
  arpsrv
```

Note that the ATMARP server currently has to be started and configured before any clients are configured.

The kernel ATMARP table can be read via \path{/proc/net/atm/arp}. The table used by atmarpd is regularly printed on standard error if atmarpd is started with the *−d* option. If atmarpd is invoked without *−d*, the table is written to the file `atmarpd.table` in the dump directory (by default `/var/run`; can be changed with *−D*), and it can be read with **atmarp −a**.

6.2. LAN Emulation

Besides Classical IP over ATM, LAN Emulation (LANE) can be used to carry IP over ATM. LANE emulates the characteristics of legacy LAN technology, such as support for broadcasts. LANE server support is described in the `src/lane/USAGE` file in the linux−atm distribution.

Man pages: bus(8), lecs(8), les(8), and zeppelin(8)

If you plan to run more than one LANE clients, LANE service or LANE clients and LANE service, you need to specify different local ATM addresses for each demon. Since all the LANE demons use similar service access points (SAPs) they need different ATM addresses to differentiate between connections.

Just as with CLIP, the LANE client consists of two parts: a demon process called zeppelin which takes care of the LANE protocol and kernel part which contains LANE ARP cache.

atmsigd and ilmid must already be running when zeppelin is started. When zeppelin starts, the kernel creates a new interface which can then be configured:

```
# zeppelin possibly_more_options &
# ifconfig lec0 local_address possibly_more_options up
```

In the example below, two LANE clients are started. The first client uses default interface lec0, default listen address and tries to join the default ELAN. The other LANE client gets interface lec2 assigned to it, binds to local address *mybox3*, tries to join ELAN called *myelan* and will bridge packets between ELAN and Ethernet segments. Address *mybox3* is defined in `/etc/hosts.atm`. Rest of the bridging can be configured by reading the Bridging mini−HOWTO. [bridge−howto]

```
# zeppelin &
# ifconfig lec0 10.1.1.42 netmask 255.255.255.0 \
                          broadcast 10.1.1.255 up
#
# zeppelin -i 2 -l mybox3 -n myelan -p &
# ifconfig lec2 10.1.2.42 netmask 255.255.255.0 \
                          broadcast 10.1.2.255 up
```

By default, zeppelin uses interface lec0, binds to local ATM address using selector byte value 0, tries to contact LECS using Well−Known LECS address, joins the default ELAN as defined by the LECS, accepts the MTU size as defined by the LES and will not act as an proxy LEC. These parameters can be tailored with command line options which are defined in zeppelin(8).

zeppelin will automatically join any ELANs which use higher MTU than the default MTU of 1516 bytes. The MTU of the LANE interface will adjust itself according to the MTU of the current ELAN.

The state of the LANE ARP cache entries can be monitored through `/proc/net/atm/lec`. For each entry the MAC and ATM addresses and status is listed. If the entry has an active connection, the connection identifiers are also listed.

The LANE service (lecs(8), les(8), and bus(8)) is configured using configuration files. The configuration file syntax is listed on the respective manual pages.

A more detailed description of Linux LANE services is discussed in Marko Kiiskilä's Master's Thesis [kiis].

6.3. MPOA

The Linux MPOA client continues the tradition of user space −− kernel divided ATM services. The demon process called mpcd processes MPOA control packets while the kernel holds MPOA ingress and egress caches and does the packet forwarding.

Man page: mpcd(8)

atmsigd and ilmid must already be running when mpcd is started. Since MPOA detects IP layer flows from LANE traffic, you need to have zeppelin running before MPOA can function. However, the order in which zeppelin and mpcd is started is not fixed. You can kill any of the demons at your will and restart it later without need to restart the other demon. The easiest way to disable MPOA is to kill the running mpcd.

Below is the example from Section *LAN Emulation* which starts two LANE clients. The configuration has been augmented with two MPOA clients which the LANE clients will serve.

```
# zeppelin &
# ifconfig lec0 10.1.1.42 netmask 255.255.255.0 \
                          broadcast 10.1.1.255 up
# mpcd -s mybox1 -l mybox2 &
#
# zeppelin -i 2 -l mybox3 -n myelan -p &
# ifconfig lec2 10.1.2.42 netmask 255.255.255.0 \
                          broadcast 10.1.2.255 up
# mpcd -i 2 -s mybox4 -l mybox5 &
```

The MPOA demon needs two different local ATM addresses which it uses when initiating and receiving data and control connections. The addresses can be the same as with e.g. zeppelin but must be different among other mpcd demons. By default, mpcd does not retrieve configuration information from the LECS. The necessary command line options and an example of using LECS are shown on the mpcd manual page. The manual page also lists the rest of the available options.

The contents of MPOA ingress and egress caches can be monitored through the `/proc/net/atm/mpc` file.

The Linux MPOA client also supports CBR traffic class for shortcuts SVCs instead of default UBR. The QoS specifications for future shortcuts can be set and modified using `/proc/net/atm/mpc`.

```
# echo add 130.230.54.146 tx=80000,1600 rx=tx > /proc/net/atm/mpc
#             # generate enough traffic to trigger a shortcut
# cat /proc/net/atm/mpc
QoS entries for shortcuts:
IP address
  TX:max_pcr pcr     min_pcr max_cdv max_sdu
  RX:max_pcr pcr     min_pcr max_cdv max_sdu
130.230.54.146
     80000   0       0       0       1600
     80000   0       0       0       1600

Interface 2:

Ingress Entries:
IP address      State     Holding time  Packets fwded  VPI VCI
130.230.4.3     invalid   1160          0
130.230.54.146  resolved  542           151            0   109
...
```

The shortcut to IP address *130.230.54.146* was established with the parameters shown above. There also exist patches which extend the flow detection to fully support layer 4 flows. The layer 4 flows are expressed as a 5 tuple (proto, local addr, local port, remote addr, remote port) and they identify application to application flows. If you are interested, see ftp://sunsite.tut.fi/pub/Local/linux−atm/mpoa/ for the latest patch.

Bibliography

References

[api] *Linux ATM API*, Werner Almesberger, http://linux−atm.sourceforge.net/API/ , July 1996.

[RFC1577] *Classical IP and ARP over ATM (RFC1577)*, Mark Laubach, January 1994.

[lanev1] *LAN Emulation Over ATM −− Version 1.0*, ATM Forum, February 1996.

[lanev2] *LAN Emulation Over ATM −− Version 2 −− LUNI Specification*, ATM Forum, July 1997.

[mpoav1] *Multi−Protocol Over ATM −− Version 1.0*, ATM Forum, July 1997.

[bridge−howto] *Bridging mini−Howto*, Christopher Cole,
http://www.linuxdoc.org/HOWTO/mini/Bridge.html , March, 2001.

[kiis] *Implementation of LAN Emulation Over ATM in Linux*, Marko Kiiskilä,
ftp://sunsite.tut.fi/pub/Local/linux−atm/misc/ , October 1996.

Notes

[1]
Some adapters may only support a subset of this.