amino

Shaping IPTV

Installation and Upgrade Guide

March 2010

Copyright

Installation and Upgrade Guide

March 2010

Issue 105

The generally available software versions at the time of issue are as follows:

Platform	Software release version	CPU
103	0.15.1	IBM Vulcan
110	0.15.10	
110-H	0.15.10	
500	0.15.10	
130/130M	0.16.7	ST 71xx
130-H	0.16.7	
530	0.10.21	
125	0.16.7	TI - DM6443 or DM6441 (DaVinci)

This document describes components that undergo continual development. The information in this document is subject to change without notice at any time.

There may be visual deviations between graphics in the manuals and the released software.

Comments about the documentation are welcome. Please submit feedback to the Amino support site at http://stbsupport.aminocom.com using the **Request Support** > **Submit Feedback** option.

For further information about Amino or Amino products, please e-mail info@aminocom.com

Contents

Introduction 1
About Amino 1
About this document 1
Document history 2
Chapter 1—The software 3
1.1 Software builds 3
1.1.1 Build options 4
1.2 What is a software release? 4
1.2.1 Major core releases (0.X.0) 4
1.2.2 Release candidates (0.X.0-rcZ) 4
1.2.3 Minor core releases (0.X.Y) 4
1.2.4 Patch core releases (0.X.Ycp) 5
1.2.5 Release directory contents 5
1.2.6 Software images 6
1.2.7 Making new software images available 7
1.2.7.1 Bootstrap image 7
1.2.7.2 Upgrade image 7
1.3 What is the upgrade system? 7
1.4 Installation and upgrade prerequisites 8
1.5 Getting the software 8
1.6 Setting up a multicast system 8
Chapter 2—Installing the software 9
2.1 Starting the installation 9
2.1.1 Setting up the DHCP server 10
2.2 Configuring the set-top box time zone 15
2.2.1 To set the time zone over DHCP 16
2.2.2 To set the time zone in the netconf file 16
2.3 Setting up the multicast server 16
2.3.1 Example Image sections 17
2.3.2 Example Filesystem sections 18
2.3.3 DHCP and multicast server matching values 19
2.4 Software images 20
2.4.1 Creating a signed bootstrap image 20
2.4.2 Creating a signed upgrade image 21
2.4.3 Bootstrap/mc2 packet sizes 23
2.5 Starting the multicast server 23
2.6 Amino vendor options in DHCP 23

2.6.1 Standard DHCP options supported . . . 25
2.6.2 Class IDs . . . 26
2.6.2.1 Linux . . . 26
2.6.2.2 IntActOS - multicast versions . . . 26
2.6.2.3 IntActOS - TFTP versions . . . 27
2.6.3 Client IDs . . . 27
2.7 Using symbolic links to reduce multicast server configuration when adding new images 27
Chapter 3—Upgrading the STB . . . 31
3.1 Upgrade mechanisms . . . 31
3.1.1 Using STBremoteconf to upgrade software . . . 32
3.1.2 Using the Management pages to upgrade software . . . 32
3.1.3 Using the DHCP server to upgrade software . . . 32
3.2 Upgrading the software . . . 33
3.2.1 Create a signed bootstrap image . . . 33
3.2.2 Create a signed upgrade image . . . 33
3.2.3 Start the multicast server . . . 34
3.3 About set-top box operation . . . 34
3.3.1 Static DHCP settings . . . 36
3.4 Operational states . . . 36
3.4.1 Bootstrap (mboot) state . . . 36
3.4.1.1 Operation in the bootstrap (mboot) state . . . 37
3.4.1.2 Send DHCP request . . . 38
3.4.1.3 DHCP response . . . 38
3.4.1.4 Download image . . . 38
3.4.1.5 Assemble, verify and execute image . . . 39
3.4.2 Upgrade (upgrd) state . . . 39
3.4.2.1 Operation in the upgrade (upgrd) state . . . 39
3.4.2.2 Send DHCP request . . . 41
3.4.2.3 DHCP response . . . 41
3.4.2.4 Download image . . . 41
3.4.2.5 Assemble, verify and execute image . . . 41
3.4.3 Filesystem (fisys) state . . . 41
3.4.3.1 Operation in the filesystem (fisys) state . . . 41
3.4.3.2 Send DHCP request . . . 42
3.4.3.3 DHCP response . . . 42
3.4.3.4 Initiate upgrade or normal use . . . 42
3.5 Set-top box configuration . . . 43
3.6 Automatic upgrading . . . 43
3.6.1 To test automatic upgrading . . . 44
3.6.2 Setting set-top boxes with static network and multicast upgrade settings . . . 44
3.6.3 Network and multicast upgrade configuration settings . . . 44
3.6.3.1 To set static multicast upgrade values . . . 45
3.6.3.2 To set a static IP address . . . 45
3.7 Automating multicast server start-up . . . 46
3.8 Using the deployment index for automatic upgrade . . . 47
3.8.1 How it works . . . 47
3.8.2 Deployment indices and software versioning . . . 47
3.8.3 Migrating from pre-DI to DI-capable software . . . 48
3.8.4 Setting up automatic upgrade to minimum deployment index . . . 48
3.8.4.1 Resetting the deployment index . . . 48

3.8.5 To implement automatic upgrading . . . 48
Chapter 4—STB Security Features . . . 51
4.1 Security keys . . . 51
4.1.1 Hierarchy . . . 51
4.1.1.1 Amino Master key . . . 52
4.1.1.2 Customer key . . . 52
4.1.1.3 STBRC key . . . 53
4.1.2 Consequences of key compromises . . . 53
4.1.2.1 Amino Master key . . . 53
4.1.2.2 Customer key . . . 53
4.1.2.3 STBRC key . . . 54
4.2 Distributing software . . . 54
4.3 Deployment Index (DI) . . . 54
4.3.1 How it works . . . 54
4.3.2 How to use it . . . 55
4.4 Passwords . . . 55
4.4.1 Changing set-top box passwords . . . 55
4.4.2 Shadow file . . . 55
4.4.3 To change passwords in the shadow file . . . 55
Appendix A—LED flashing codes . . . 57
A.1 Major error groups . . . 57
A.2 Ethernet monitoring . . . 60
Appendix B—DHCP Configuration . . . 63
Appendix C—The mcastbootd.conf file . . . 83
C.1 mcastbootd.conf file components . . . 83
C.1.1 File header . . . 83
C.1.2 [Server] section . . . 83
C.1.3 Bootstrap and mc2 filesystem section commonalities . . . 84
C.1.4 Bootstrap Section . . . 84
C.1.5 mc2 Filesystem Section: . . . 84
C.2 Example mcastbootd.conf file . . . 85
C.3 Troubleshooting and tips to measure and test bandwidth/MTU/TTL . . . 85
C.3.1 Common network monitoring/bandwidth measuring tools . . . 85
C.3.2 dhcpd interaction . . . 86
C.3.2.1 Common problems/mistakes . . . 86
Index . . . 87

Introduction

About Amino

Amino Communications (http://www.aminocom.com) is the specialist in digital entertainment products. Amino's range of software and set-top box systems can be tailored for telecom, broadcast and hospitality firms to offer highly scalable and targeted services.

The award-winning AmiNET™ and Mood range is used by leading service operators in over 80 countries. Amino's 'AssetHouse' technology opens the door for content producers, telecoms companies, broadcasters and web TV firms to maximise opportunities through better services, targeted content and greater choice and taking IPTV to the next level by allowing clients such as BT Vision to think like retailers and package, personalise and refresh extra revenue-generating services to viewers.

Amino Communications and AssetHouse are wholly owned subsidiaries of Amino Technologies PLC. Listed on the London Stock Exchange AIM, symbol AMO. Amino's HQ is based near Cambridge, UK, with offices in the US, China and Sweden.

About this document

AmiNET set-top box operation is controlled by software installed on the set-top box. The Amino multicast upgrade system provides management, control and customisation for set-top boxes. This document describes how to install new software on your system, and describes the upgrade process. It assumes you have a set-top box and a suitable software release. It also assumes a basic understanding of the technologies involved, which include Linux command line operation.

Many of the control and customisation options described in this document can also be implemented using JavaScript Media Access Control Extensions (JMACX). JMACX is not covered in this document – see the Amino *JMACX API specification* for details.

Document conventions

The following document conventions are in use:

Formatting	Usage
< ...>	Indicates a value that you need to replace with a system specific value (except where used in HTML or XML examples, where it is used in tags, as normal).
[...]	Indicates optional parameters - for example in commands or functions.
... | ...	Indicates choices - for example where an input can take one of a number of values.
`code font`	Indicates input and output values (for example, at a command line), as well as function, configuration, parameter and file names.

Formatting	Usage
bold text	Used for emphasis and to indicate specific key presses. For example: Press the **Esc** key.
grey text	Commands or settings which are not in general use (for example, configuration settings that are reserved for Amino internal use).
blue text	Cross-reference to other sections and other documents (this is a “clickable” hyper-link if you are viewing the document electronically).

Structure

This document consists of the following chapters:

Chapter	Outline
”Introduction“	General introduction.
”The software“	A description of software builds, and the upgrade system.
”Installing the software“	A step-by-step guide to installing the software.
”Upgrading the STB“	Methods of upgrading and updating the soft-ware.
”STB Security Features“	Describes the use of the Amino security keys.
Appendix A, ”LED flashing codes“	Describes the sequence of short and long flashes of the STB LED which communicates general information and error details.
Appendix B, ”DHCP Configuration“	An example configuration of the complete `dhcp.conf` file.
Appendix C, ”The mcastbootd.conf file“	Details of `mcastbootd` configuration.
Appendix D, ”Troubleshooting“	Common pitfalls and problems.

Document history

Version	Date issued	Changes
105	March 2010	Minor corrections and updates. Troubleshooting section removed.
104	August 2009	Appendix C on `mcastbootd.conf` added. References to the AmiNET 120 and 124 removed. Customers requiring information on these platforms should consult previous versions of the documentation. Updates for Opera9 based platforms
103	June 2009	Removed confidentiality requirement. Password protection warnings.
102	May 2009	Minor corrections.
101	February 2009	Document created. This document replaces parts of the Amino *Set-Top Box Operations Guide* v0.14.x Revision 2

Chapter 1—The software

AmiNET set-top box operation is controlled by software installed on the set-top box. The Amino multicast upgrade system provides management, control and customisation for set-top boxes. This document describes how to install the software on your system. It assumes you have a set-top box and can obtain a suitable software release. It also assumes a basic understanding of the technologies involved, which includes Linux command line operation.

1.1 Software builds

AmiNET set-top boxes support a wide range of functions. The software enables as much user customisation as possible, but in order to minimise the size of the software image loaded on a set-top box, it is built to support a particular set of functions. Once a build has been created, it is possible to customise it with customer specific security keys, additional files and configuration settings. However, it is not possible to enable functionality that has not been compiled into the software build. For this reason, it is useful to understand what is included in a software build, and what can be added later.

Note: The build stage shown in this diagram is internal to Amino.

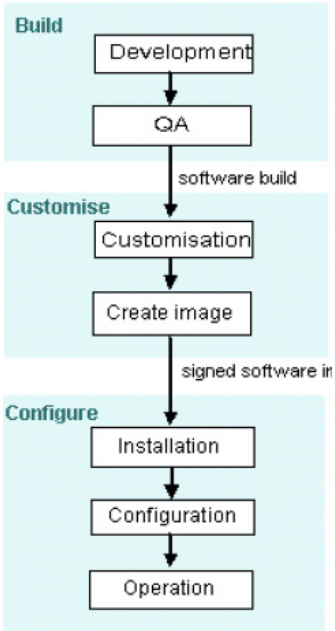


Figure 1.1 Software build and customisation stages

The following table shows the stages at which some of the key functionality can generally be added or changed:

Stage	Options
Build	Browser and middleware support Hardware platform Subtitles BitBand support Some CA system support Some language support
Customise	RSA security keys Enable/disable Telnet/SSH Graphic shown during upgrade Some CA system support Some language support Packet size and file format for multicast software images
Configure	Configuration settings – e.g. browser display, language choice, output format

1.1.1 Build options

Various set-top box functions are locked or disabled during a hardware build, to protect the set-top box from interference. This includes identity settings such as MAC address and serial number. Set-top boxes with these functions enabled are referred to in this documentation as “development” builds (these are sometimes also called “production”. Set-top boxes with the functionality locked are known as “live” builds. Note that software releases designed to support functions only available during development include `production` in the name of the release version.

1.2 What is a software release?

Software releases are planned to primarily provide a major core release and then interim point releases for urgent customer fixes.

1.2.1 Major core releases (0.X.0)

These are planned by date and driven by new features, such as core improvements or support for new platforms.

1.2.2 Release candidates (0.X.0-rcZ)

Released as part of the major core development, release candidates allow the new features and improvements to be tested before customer builds use the core.

1.2.3 Minor core releases (0.X.Y)

Minor point releases are driven by bug fixes and are not planned to a specific date as closely as major releases. Point releases will typically occur once all the scheduled bugs against them are fixed, but may be pulled forwards if required for specific customer releases.

1.2.4 Patch core releases (0.X.Ycp)

It is assumed that whenever a new major core is released, it contains all the features found in the previous major core including those in point releases for that major core. Not all customers are able to move to the latest core version so they may subsequently require patches to older core versions. In the past these have been considered as point releases for the older core but this leads to confusion.

Consider the following hypothetical example: Development of 0.13 stops at 0.13.7 because 0.14.0 (which contains fixes up to and including 0.13.7) is released. Point releases of 0.14 begin. Customer A requires a fix to their 0.13 release but cannot move to the 0.14 baseline for whatever reason. Instead of a 0.13.8 point release being created, which may lead to confusion as to whether that fix was included in 0.14.0 or not, 0.13.7 gets patched to create a 0.13.7a1 release. Subsequent patches for customer A will increase the final number - 0.13.7a2, 0.13.7a3, etc. If other customers require different patched versions, the letter is changed. For example, 0.13.7b1 or 0.13.7c1

1.2.5 Release directory contents

A compressed standard release file supplied by Amino unzips to the directory structure shown in the table below. The actual contents of the directories depends upon the release – for example, CA-specific files and browser-specific files will be included with some releases, while other releases may not include keys or the full set of management utilities. The table below shows the location of the main files described in the documentation.

Note: In the following table, **bold** text indicates a directory.

File / subdirectory	Description
bootstrap	Directory containing the components needed to create a signed bootstrap image.
`bootstrap.unsigned`	The unsigned bootstrap image.
`signbootstrap`	The script to use to create a signed bootstrap image.
romupgrade	Directory containing files needed to upgrade a TFTP booting set-top box to use the multicast upgrade system.
tftpboot	TFTP to multicast upgrade files.
server	Directory containing files for setting up the multicast cd ../ upgrade system on the server.
`dhcpd.conf`	Configuration file containing settings that you need to edit and add to your DHCP server's configuration file.
`mcastbootd`	Amino multicast server binary file.
`mcastbootd.conf`	Multicast server configuration file.
upgradeimage	Directory that contains the components needed to create a signed upgrade image.
`flashcontents`	File that lists the files to use to create the software upgrade image (i.e. the `imagecomponents` files).
`signupgradeimage`	The script to use to create a signed software upgrade image.
`createtftpboot`	Takes the contents of the `imagecomponents` directory and creates a TFTP bootstrap in the same manner as `signupgradeimage`.
`mc2.mcfs`	Multicast upgrade image. This is created after the `signupgradeimage` script is run.

File / subdirectory	Description
imagecomponents	Directory containing the files to be used as the contents of a software upgrade image.
config.txt	Browser configuration file (Fresco).
fkeys.conf	Configuration file defining actions associated with IR remote control and keyboard keys. You are not recommended to edit this file directly.
loading.gif	Graphic shown on screen when the set-top box is loading a new software upgrade image.
mcnfg.tar	Tar file containing HTML used to display the set-top box Configuration pages.
netconf	Configuration file containing network settings.
opera.ini	Browser configuration file (Opera).
settings	Configuration file containing general set-top box settings.
splash.gif	Graphic shown on screen when the set-top box is booting.
utils	Directory containing management utilities.
commands	File containing commands used by STBremoteconf.
imgcfg	A utility used by signupgradeimage to create signed bootstraps or mc2 configuration files.
STBremoteconf	Utility that enables remote configuration of set-top boxes.
rsakey	RSA key generating utility.
customise	Script used to merge custom resources and postbuild packs with your release.
signrom	Creates mc2.mcfs and signs it.
verifysignature	Validates additional customer keys.
keys	Directory containing the private keys used to sign and verify software images and STBremoteconf commands.
amino	The Amino engineering keys directory.
KEY.private	The private customer key.
KEY.public	The public customer key.
Master.public	The public master key.
STBrc-KEY.private	The private configuration key.
STBrc-KEY-public	The public configuration key.

1.2.6 Software images

The set-top box NAND flash memory holds two types of signed software image:

- **Bootstrap image**

  The bootstrap image contains the software needed to manage the download of a multicast upgrade image. It is a combined Linux kernel and RAM disk image with an RSA digital signature.

  A signed bootstrap image is generated as a file called bootstrap.signed.

- **Upgrade image**

  The upgrade image is a file that contains a complete software upgrade in the form of a file system image with an RSA digital signature.

  A signed upgrade image is normally generated as a file called `mc2.mcfs`.

1.2.7 Making new software images available

When the set-top box is first connected, it inspects its NAND flash, and if this contains no valid software, it connects to the multicast server and downloads a software release: first the bootstrap image, and then the upgrade image.

If you have not been supplied with signed software images, you will need to create and sign your own. To do this, you will need the `/bootstrap` and/or `/upgradeimage` directories of a software release, and the customer key and its pass phrase. Keys are normally provided as part of a software release, but you can also generate and use your own keys, as long as they are in the set-top box ROM.

1.2.7.1 Bootstrap image

You should normally only need to sign and load a bootstrap image when the set-top box is supplied with a blank NAND flash (for example, units shipped direct from manufacture). The bootstrap image is created from the `bootstrap.unsigned` file, using the `signbootstrap` script and the customer key.

1.2.7.2 Upgrade image

Upgrade images need to be created whenever you need to upgrade set-top boxes with a new software release. The upgrade image is created from the contents of the `imagecomponents` directory specified in the `flashcontents` file, using the `signupgradeimage` script and the customer key.

The upgrade image can be downloaded automatically by a set-top box (controlled by deployment index mechanism) or you can initiate multicasting (or unicasting) the image manually from the DHCP server and Amino multicast server, for example, using the `STBremoteconf` tool. If you have not been supplied with signed bootstrap and upgrade images, you need to create them yourself from the files supplied in the Amino software release.

1.3 What is the upgrade system?

The Amino multicast upgrade system offers various means of instructing set-top boxes to replace their software images:

- **STBremoteconf** – remote configuration tool that enables you to send commands such as changing basic configuration settings or initiating software upgrade.
- **Management pages** – local configuration pages, using an Amino IR keyboard and television display to change configuration settings and simple commands such as rebooting and initiating software upgrades.
- **DHCP server configuration** – DHCP server configuration specifies where set-top boxes “go” to retrieve software images transmitted by the Multicast server; it can also be configured to force upgrades by specifying minimum deployment index numbers for software versions that the set-top box must be running. DHCP server and Multicast server together also enable the set-top box to connect and retrieve new software images when required (for example, if the software on the set-top box is corrupted).
- The **JMACX API** offers functions for controlling a range of set-top box operational areas, including software upgrades. This manual does not cover the use of the JMACX functions, please see the Amino *JMACX API Specification* for more information.

1.4 Installation and upgrade prerequisites

Before using the installation and upgrade guide you should be familiar with the following Linux usage and administration skills:

- Performing a SSH login.
- Contrasting full and relative pathnames.
- Understanding the file system hierarchy.
- Handling files with `cp` and `mv`.
- Making and navigating directories.
- Listing attributes with `ls`.
- Identifying multiple users and groups.
- Interpreting file and directory modes.
- Adjusting access permissions: `chmod`.
- Raising privilege with `su`, `sudo` and `setuid`.
- Extracting lines with GNU `grep`.
- Saving command output into files.
- Creating and modifying files with `vi` or `emacs`.
- Exporting variables to the environment.
- Calling scripts as a command.
- Monitoring processes with `ps`.
- Mounting storage devices.
- Measuring free space.
- Creating `tar` archives.

1.5 Getting the software

Amino provides software updates to our corporate customers with a valid support account and the appropriate software can be downloaded from the Amino support site at http://stbsupport.aminocom.com. Sign in to the site and select **Online support** > **Downloads**. On the left hand side of the screen select **Software upgrades**. From the available downloads select the STB type and Browser you require.

Note: Only the software to which you have access will be displayed.

A free 90 day “Sample” Account is available to all customers who have bought directly from Amino. After this period there are different paid support options to choose from. Customers who have received their Set Top Box from a 3rd party should contact them for software updates.

1.6 Setting up a multicast system

The instructions in the following chapters assume you have access to a compressed Amino release through the Amino support site and a computer with Linux and a DHCP server installed. They also assume familiarity with basic Linux command line functionality. If you intend to use `STBremoteconf` for remote control and communication with set-top boxes, then you will need Perl on the PC you are installing these components on. To complete the installation, you will also need to be able to find out settings for the network that the set-top box is connected to.

Chapter 2—Installing the software

The steps here describe how to set up a basic multicast system for the first time. Although it is not necessary to follow the installation methods exactly, Amino recommend that a similar directory structure is used in your installation, in order to simplify any subsequent support issues that may arise.

2.1 Starting the installation

Note: In all the following instructions, `<release_name>` indicates the full name of the software release image directory that contains your Amino software release. For example, `A110-0.14.0-frescoj27-ami_wm-subs-1` installs in `0.14.0-frescoj27-ami_wm-subs-1`, hence `<release_name>` is `0.14.0-frescoj27-ami_wm-subs-1`.

1. Open a terminal window and log on to your system as root, or change to root from a user using:

   `su`

2. Using `mkdir`, create directories to contain the software, for example:

   `mkdir /usr/local/amino`

   `mkdir /usr/local/amino/images` this is where the multicast server will get the images to broadcast.

   `mkdir /usr/local/amino/releases` this is where the downloaded files will be stored.

   Note: Ensure that you have the correct read/write permissions for each of these directories, using:

   `chmod 777 <directory>`

3. Copy the required `.tgz` file to the `/usr/local/amino/releases` directory.

   Note: If you require builds for different AmiNET models, it may be worth creating directories for each type, for example, `/usr/local/amino/releases/A130`.

4. Navigate to the releases directory `/usr/local/amino/releases` and unpack the Amino software release using:

   `tar zxvf <archive_name>`

   where

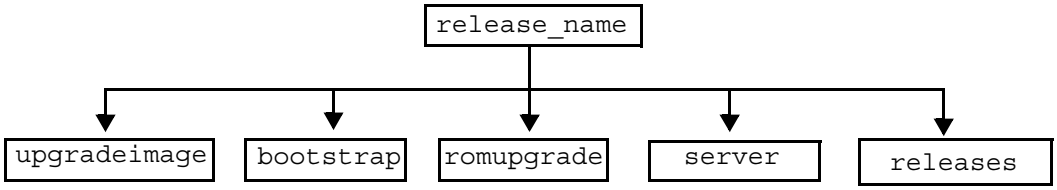
`z`	Filter the archive through gzip
`x`	Extract files from an archive
`v`	Verbosely list files processed
`f`	Use the specified archive file.

for example:

```
tar zxvf A110-0.14.0-frescoj27-ami_wm-subs-1.tgz
```

unpacks `A110-0.14.0-frescoj27-ami_wm-subs-1.tgz` to `0.14.0-frescoj27-ami_wm-subs-1`

This will then contain the following sub-directories.



See "Release directory contents" on page 5 for further details of this.

`bootstrap`	Contains the files needed for signing a bootstrap image.
`romupgrade`	Contains the files needed when converting from `tftp` to multicast upgrade.
`server`	Contains the files needed for the `mcastbootd` server.
`upgradeimage`	Contains the files needed for signing an upgrade image.
`utils`	Contains the generic Amino keys, `STBRemoteconf` and binaries required for all the above processes.

5. Copy the multicast server binary to a directory in the system path, using:

   ```
   cp <release_name>/server/mcastbootd /usr/local/bin
   ```

6. Copy the contents of the `utils` directory to the same location, using:

   ```
   cp <release-name>/utils/* /usr/local/bin
   ```

   Note: You may see a warning message about `cp` omitting the `/utils/keys` directory. This can be ignored safely.

7. Copy the multicast server configuration file to the `/etc` directory, using:

   ```
   cp <release_name>/server/mcastbootd.conf /etc
   ```

8. If you are using a dedicated DHCP server (only for this multicast system), then copy the sample configuration file `dhcpd.conf` from the software release to the `/etc` directory, using:

   ```
   cp <release_name>/server/dhcpd.conf /etc
   ```

   If you are using a DHCP server that has already been set up with an existing installation, you will be able to copy the configuration options settings from the sample file (`dhcpd.conf`) to your existing DHCP configuration file when you set up your DHCP server for use with Amino products

All the files needed to run the multicast system have now been copied to the locations where they are needed. You will also still need to access some of the other files in the software release directory.

2.1.1 Setting up the DHCP server

For a first set-up of a multicast system (for example, for trial purposes), you are recommended to set up a DHCP server that will only be used for this multicast system, in order to keep configuration simple while you are getting to know the system functionality.

In many deployments it is likely that there will be more than one type of set-top box installed, each needing different software images. To implement this, you will need to change both the DHCP server

and multicast server configuration, and ensure that software images for different set-top box types are uniquely identifiable (either give them different names or store them in separate directories).

A sample configuration file is supplied with the software release, and you can either edit this file or copy the settings from this file to your existing DHCP server's configuration file.

1. Open the sample DHCP server configuration file `dhcpd.conf` in a text editor.

   Note: If you copied the file over when you copied the binaries (Step 8. above), then this is located in `/etc`. Otherwise, it is in the `<release_name>/server/` directory and you should complete Step 8. above, before continuing.

   If you are adding settings to an existing DHCP server configuration file, for example for another set-top box, or an additional type of set top box, you will also need to open the **existing** configuration file for your DHCP server, and modify the settings in this file, as per the changes below. For an example of an edited DHCP configuration file, see Appendix B, "DHCP Configuration"

2. The following settings from the sample configuration file need to be copied to the DHCP server configuration file that you wish to use.

   Misc options

   ```
   allow bootp;
   ddns-update-style ad-hoc;
   filename="AMINET.txt";
   ```

   Extra Options for AMINO option space (used for multicast)

   ```
   option space AMINO;
   option AMINO.address          code 1 = ip-address;
   ```
   to..
   ```
   option AMINO.middleware2      code 17 = ip-address;
   ```

   For further information on this section, see "Amino vendor options in DHCP" on page 23

3. For each set-top box type that you want to multicast to, you will need classes that specify how set-top boxes can get software images when they are in boot state (`mboot` class), upgrade state (`upgrd` class) and normal boot state (`fisys` class).

   The classes are specified as follows (for the A110) in the sample file:

   ```
   ################################################################
   #   AmiNET110 Configuration Section                            #
   ################################################################
   # class "AmiNET mboot" - boot state when requesting            #
   #                        boostrap image                        #
   # class "AmiNET upgrd" - boot state when requesting main       #
   #                        upgrade image                         #
   # class "AmiNET fisys" - boot state when in normal state       #
   #                                                              #
   #  The only states that need changing are as follows:          #
   #  option AMINO.address 225.50.50.50 - the multicast           #
   #  address you are streaming on                                #
   #  option AMINO.port 11111; - the port you are streaming on    #
   #                                                              #
   #  If you change any of these options you must also make       #
   #  sure that you make the appropriate changes to               #
   # /etc/mcastbootd.conf                                         #
   #                                                              #
   ```

```
# In the mboot class, the bootrom version (e.g. 1.32) must  #
# be given in the vendor-vlass-identifier. If multiple      #
# bootrom versions need to be supported multiple match      #
# cases may be used.                                        #
############################################################
```

AmiNET<STBtype> configuration section(s)

```
class "AmiNET<STBtype> mboot" ...
class "AmiNET<STBtype> upgrd" ...
class "AmiNET<STBtype> fisys" ...
```

Use these classes exactly as they are shown in the sample file, and edit just the following options in the `mboot` and `upgrd` classes, to use a unique IP address for each class.

```
class "AmiNET110 mboot"
{
  match if (optiion vendor-class-identifier="aminoAMINET11xmboot
             <bootrom version>) or
((substring(option vendor-encapsulated-options, 2, 9)-"AMINET11x")
and (substring(option vendor-encapsulated-options, 13, 5)="mboot"));

vendor-option-space AMINO;
option AMINO.address 225.50.50.52;
option AMINO.port 11111;
```

Note that `225.50.50.52` and `11111` are only example values.

4. The last two lines of the `mboot` and `upgrd` classes are in the format:

   ```
   #option AMINO.address <multicast_address>;
   #option AMINO.port <port_number>;
   ```

   Uncomment these lines.

   The multicast address/port number pair must be unique per set-top box. For this reason either the `multicast_address` or the `port_number` used for `mboot` and `upgrd` should be different.

   For `mboot` replace `<multicast_address>` with the IP address of the multicast server you are streaming on.

   For `upgrd` replace `<multicast_address>` with the IP address of the multicast server you are streaming on.

   Replace `<port_number>` with the port number of the port you are streaming on.

5. You will also see the following lines in the `fisys` classes in the file:

   ```
   option AMINO.middleware <middleware server IP address>;
   option AMINO.mw_port <middleware server port>;
   ```

   If you have Minerva middleware on the set-top box type, edit these lines with IP address and port values for the middleware server. For all other Middlewares, and set-top box models, comment out these lines by adding a `#` character to the start of each line.

6. **Make a note of the multicast IP addresses and ports you have specified for each set-top box type, as you will need to use the same values when you set up the multicast server.**

7. **The `mboot` and `upgrd` classes for each STB type should also have unique IP address and port pairs.**

8. Edit the following declarations with the settings for your network. You may need to edit the following lines:

```
#################################################################
#   Subnet Declaration                                          #
#################################################################
subnet 192.168.1.0 netmask 255.255.255.0 (
```

This is optional, but if required replace the existing value with:

subnet `<IPaddress of the subnet of DHCP>` netmask `<sub-net netmask>`

For example:

```
subnet 10.172.0.0 netmask 255.255.0.0 (
```

```
#################################################################
#   Default Gateway - This MUST be set!!                        #
#################################################################
option routers 192.168.1.1;
```

Replace this with `option routers <IPaddress of the default gateway the STB uses to connect to the internet>;`

For example:

```
option routers 10.172.0.1;
```

```
#################################################################
#   Subnet Mask - This MUST be set!!                            #
#################################################################
option  subnet-mask 255.255.255.0;
```

Replace this with `options subnet-mask <IPaddress of the subnet mask>;`

For example:

```
option subnet-mask 255.255.0.0;
```

```
#################################################################
#   Domain Name - Optional                                      #
#################################################################
option domain-name “blahblah.com”;
```

This should be replaced with your DNS name.

For example:

```
option domain-name aminocom.com.
```

```
#################################################################
#   DNS Servers - Optional                                      #
#################################################################
option domain-name-servers 192.168.1.1,192,168.1.2;
```

This is optional, but if required replace this with:

```
option domain-name-servers <IP address of the above DNS name>,<alter-
native IP address of the above DNS name>;
```

```
#####################################################################
# NTP Servers  - Optional                                           #
#####################################################################
# option ntp-servers europe.pool.ntp.org; # example free NTP service
```

```
#######################################################################
# Time Offset - Optional                                              #
#                                                                     #
# This can be in one of three string formats e.g.                     #
#   UNIX style Region/City - "Europe/London"                          #
#   Commonly used alias - "GB-Eire"                                   #
#   Fully-specified Posix 1003.1 timezone string -                    #
#      "GMT0BST-1,M3.5.0/01:00,M10.5.0/02:00"                         #
#                                                                     #
# TZ info can be found in the following links:                        #
#   http://www.twinsun.com/tz/tz-link.htm                             #
#   http://h71000.www7.hp.com/doc/73final/6497/6497pro_007.html       #
#   http://en.wikipedia.org/wiki/List_of_tz_zones_by_name             #
#                                                                     #
#######################################################################
option AMINO.timezone "Europe/London";

#################################################################
#   Time Offset - Optional                                      #
#################################################################
#option time-offset -5; # Eastern Standard Time
#option time-offset +8; # Eastern Standard Time
#option time-offset -5; # Pacific Standard Time
option time-offset 0; # Greenwich Mean Time

#################################################################
#   Address Pool - This MUST be set!!                           #
#                                                               #
#   In this address pool we list the classes which we wish      #
#   to give addresses to, unless a device is in this list       #
#   it will not be given an address!                            #
#################################################################
pool {
      range dynamic_bootp 192.168.1.50 192.168.1.100
      range 192.168.1.101 192.168.1.200

      ...
     }
```

For a range of set-top boxes

Replace the `pool` section of the `dhcpd.conf` file with:

```
pool{
    range 192.168.1.50 192.168.1.100;

    }
```

where `range` is the range of the IP addresses for multicast booting and upgrades, in this case from `192.168.1.50 - 100`.

For individual set-top boxes

In the case where you have a smaller number of set-top boxes and wish to manage them individually, as in a development network, the pool information should be replaced with STB specific information similar to the following:

```
host aminet110_001 {
              hardware ethernet 00:03:0A:0C:92:91;
                fixed-address 10.172.227.64;
                   }

host aminet110_002 {
             hardware ethernet 00:02:0A:13:AA:61;
             fixed-address 10.172.227.65;
                   }
```

9. Make any further settings you require for your system.

   Note: From the following software versions

Platform	Software release version	CPU
103	0.13.0	IBM Vulcan
110	0.13.0	
110-H	0.13.0	
500	0.13.0	
130	0.8.2	ST 71xx
130-H	0.8.2	
125	0.12.1	TI DaVinci

   you can add User Class options to the configuration file if required, and they will be passed to the set-top box in the DHCP response.

10. When you have added classes for each set-top box type that you are supporting, save your changes, then copy the edited `dhcpd.conf` file to the `dhcp3` directory.

    ```
    cp dhcpd.conf /etc/dhcp3/
    ```

11. Reboot the DHCP server with the following command:

    ```
    /etc/init.d/dhcp3-server restart
    ```

    The minimum DHCP server set-up for a multicast system is now complete.

2.2 Configuring the set-top box time zone

Set-top boxes support time zone handling through an NTP client.

The time zone can be set in one of the following ways:

- Dynamic configuration, using the DHCP server
- Static configuration, in the `netconf` file

In both cases, time zone strings can be in the following formats:

- Fully specified Posix 1003.1 time zone string (for example, `GMT0BST-1`, `M3.5.0/01`, `M10.5.0/02:00`

- Unix-style definition in the format `Region/City` (for example, `Europe/London`)
- Commonly used alias (for example, `GB-Eire`)

2.2.1 To set the time zone over DHCP

If the set-top box retrieves its settings dynamically from the DHCP server, you can include the time zone in the settings that it retrieves.

1. Open the DHCP server configuration file (`dhcpd.conf`) in a text editor. This file should be found in the following location:

   `/etc`

2. Add the following to the list of Amino vendor extensions (if it is not already there):

   ```
   option AMINO.timezone code 16=text;
   ```

3. In the `subnet` declaration in the configuration file, add a line that specifies the time zone you want to set.

   Example:

   ```
   option AMINO.timezone “Europe/Stockholm”;
   ```

   The `subnet` declaration will now look similar to the following:

   ```
   subnet 10.172.0.0 netmask 255.255.0.0 {
   option routers 10.172.0.1;
   option subnet-mask 255.255.0.0;
   option domain-name "aminocom.com";
   option domain-name-servers 10.171.22.7,10.171.22.9;
   option AMINO.timezone “Europe/Stockholm”;

   pool {
       range 10.172.230.50 10.172.230.100;
        }
   }
   ```

Note: Save your changes and restart the DHCP server.

Next time the set-top box sends a DHCP request (for example, when it is rebooted), the time zone will be included in the information that the DHCP server sends.

2.2.2 To set the time zone in the netconf file

For further information on using the `netconf` settings, see the Amino *Set-Top Box Configuration Guide*.

If you want to set the set-top box’s time zone statically, you can do it with the following setting in the `netconf` file:

```
TIME_ZONE
```

Note: If the set-top box receives a time zone value from the DHCP server, it uses this value in preference to the one in the `netconf` file.

2.3 Setting up the multicast server

The following instructions describe the minimum steps that you need to complete to set up a working multicast server that provides software images for a single type of set-top box (for example, as part of a trial system).

The instructions that follow assume that you have already installed, configured and started your DHCP server, and that you have copied the files from your software release to the example locations.

1. Navigate to the `/etc` directory, and open the following file in a text editor:

   `mcastbootd.conf`

2. Add the `Image` and `Filesystem` sections for each type of set-top box. Note that each section must be named uniquely. The software images for each set-top box type must be unique (they can either be stored in separate directories or named uniquely by renaming the image files you have created).

3. In the `[Image bootstrap.signed]` section, edit the following settings to specify the IP address and port for multicasting the bootstrap image.

   ```
   MulticastIPAddress=<mcast address>
   MulticastUDPPort=<port number>
   ```

   For example:

   ```
   MulticastIPAddress=239.255.200.2
   MulticastUDPPort=11111
   ```

2.3.1 Example Image sections

This contains two `Image` sections that specify bootstrap images for two platforms. Note that the images are named uniquely - `bootstrap.110` and `bootstrap.130`:

```
[Image bootstrap.110]
MulticastIPAddress=239.255.230.52        <-This is an example value.
MulticastUDPPort=11111
FileName=bootstrap.110
Description=Linux bootstrap image
ImageType=1
SerialNumber=1
PacketSize=1456
CycleTime=0

[Image bootstrap.130]
MulticastIPAddress=239.255.230.90       <-This is an example value.
MulticastUDPPort=11111
FileName=bootstrap.130
Description=Linux bootstrap image
ImageType=1
SerialNumber=1
PacketSize=1456
CycleTime=0
```

These must be the same values as specified in the DHCP server configuration file's `mboot` class.

4. In the same section, check that the `FileName` setting lists the directory that you will be copying signed bootstrap images to and has the correct file name.

   `Filename=bootstrap.signed`

   This is the bootstap file that will be broadcast by the server.

   If you have used the Amino recommended settings, this is:

   `/usr/local/amino/images/bootstrap.signed`

5. In the `[Filesystem mc2]` section, edit the following settings to specify the IP address and port for multicasting the bootstrap image.

   ```
   MulticastIPAddress=<mcast address>
   MulticastUDPPort=<port number>
   ```

   For example:

   ```
   MulticastIPAddress=239.255.200.3
   MulticastUDPPort=11111
   ```

   These must be the same values as specified in the DHCP server configuration file's `upgrd` class.

2.3.2 Example Filesystem sections

For transmitting software upgrade images for the same platforms as in the `Image` section examples above. Again the upgrade images are named uniquely - `mc110` and `mc130`:

```
[Filesystem mc110]
MulticastIPAddress=239.255.230.53       <-This is an example value.
MulticastUDPPort=11111
ImageName=mc110
Description=upgrade filesystem
SerialNumber=8
DirsPerCycle=128
DataRate=256

[Filesystem mc130]
MulticastIPAddress=239.255.230.91       <-This is an example value.
MulticastUDPPort=11111
ImageName=mc130
Description=upgrade filesystem
SerialNumber=5
DirsPerCycle=128
DataRate=256
```

Important: The multicast server configuration file must contain a `Filesystem` section called `mc2`. If this section is removed, the `signupgradeimage` script cannot create new software upgrade images successfully. If you want to name sections appropriately for each set-top box type, then you should leave this section in the file as a dummy section, and add new sections for each of the set-top box types that you want to support. If there is no software upgrade image at the location specified in this dummy section, the multicast server will generate errors on start-up, but will still run.

Example [`Filesystem mc2`] section

(this can be a dummy section, as described above):

```
[Filesystem mc2]
MulticastIPAddress=239.255.230.100       <-This is an example value.
MulticastUDPPort=11111
ImageName=mc2
Description=upgrade filesystem
SerialNumber=73
DirsPerCycle=128
DataRate=256
```

6. Check that the `ImageName` setting lists the directory that you will be copying signed software upgrade images to and has the correct name for the image.

   ```
   ImageName=mc2
   ```

   This is the upgrade image that will be broadcast by the server.

7. If you are setting up the multicast server to transmit software for multiple set-top box types, you will need to add separate `[Image <filename>]` and `[Filesystem <imagename>]` sections for each of the types of set-top box. The `<imagename>` and `<filename>` must be unique, and must match the name of the bootstrap and upgrade images for that type of set-top box. (To set this up, you will also need to rename the software images you create, so that the image name for each type of set-top box is unique.) This is not necessary if you have only configured the DHCP server for a single type of set-top box.

The multicast server is now set up. You should now create software images and copy them to the directory specified in the configuration file before you start the multicast server.

2.3.3 DHCP and multicast server matching values

For each set-top box type set up in the DHCP server, the multicast values that the configuration specifies must match the values that the multicast server uses for transmitting software images. The following table shows the equivalent settings in the example DHCP server and multicast server configuration files.

Note: The multicast server values shown are example values.

DHCP server configuration	Multicast server configuration
`class "AmiNET110 mboot"` `option AMINO.address 239.255.230.52;`	`[Image bootstrap.110]` `MulticastIPAddress=239.255.230.52`

`option AMINO.port 11111;`
`class "AmiNET110 upgrd"`
`option AMINO.address 239.255.230.53;`
	`MulticastUDPPort=11111`
	`[Filesystem mc110]`
	`MulticastIPAddress=239.255.230.53`

`option AMINO.port 11111;`	`MulticastUDPPort=11111`
`class "AmiNET110 fisys"`	Example configuration file contains no settings for retrieving upgrade image in `fisys` state, but it could use the same values as in the `[Filesystem mc110]` section (e.g. if using deployment index for automatic upgrades).

`class "AmiNET130 mboot"`
`option AMINO.address 239.255.230.90;`
`option AMINO.port 11111;`
	`[Image bootstrap.130]`
	`MulticastIPAddress=239.255.230.90`
	`MulticastUDPPort=11111`

`class "AmiNET130 upgrd"`
`option AMINO.address 239.255.230.91;`
`option AMINO.port 11111;`
	`[Filesystem mc130]`
	`MulticastIPAddress=239.255.230.91`
	`MulticastUDPPort=11111`

`class "AmiNET500 mboot"`
`option AMINO.address 239.255.230.80;`
`option AMINO.port 11111;`
	`[Image bootstrap.500]`
	`MulticastIPAddress=239.255.230.80`
	`MulticastUDPPort=11111`

DHCP server configuration	Multicast server configuration
class "AmiNET500 upgrd" option AMINO.address 239.255.230.81; option AMINO.port 11111;	[Filesystem mc500] MulticastIPAddress=239.255.230.81 MulticastUDPPort=11111

2.4 Software images

The set-top box NAND flash memory holds two types of signed software image:

- **Bootstrap image**

  The bootstrap image contains the software needed to manage the download of a multicast upgrade image. It is a combined Linux kernel and RAM disk image with an RSA digital signature.

  A signed bootstrap image is generated as a file called `bootstrap.signed`.

- **Upgrade image**

  The upgrade image is a file that contains a complete software upgrade in the form of a file system image with an RSA digital signature.

  A signed upgrade image is normally generated as a file called `mc2.mcfs`.

2.4.1 Creating a signed bootstrap image

The following steps detail how to create a signed bootstrap image. This is generated as a file called `bootstrap.signed`.

The steps outlined here assume that you have set up a multicast server and you have logged in as `root`. The Amino software release normally contains the engineering customer key that you will need for creating the bootstrap image. If you are using your own keys, then you will need to know the pass phrase for your key.

1. Set the `CUSTOMER_KEY` environment variable by entering a command in the following format:

   ```
   export CUSTOMER_KEY=/usr/local/amino/releases/<release_name>/utils/
   keys/amino/KEY.private
   ```

2. Navigate to the `bootstrap` directory:

   ```
   cd /usr/local/amino/releases/<release_name>/bootstrap
   ```

3. Run the `signbootstrap` script by entering the following command:

   ```
   ./signbootstrap
   ```

   You will be prompted to enter the password for the key. For the Amino engineering key, the password is `markskey`

4. The script generates a `bootstrap.signed` file in the `bootstrap` directory. Copy this file to the `/usr/local/amino/images` directory specified by the `FileName` setting in the `[Image <image_name>]` section of the multicast server configuration file (`mcastbootd.conf`).

   For example:

   ```
   cp bootstrap.signed /usr/local/amino/images/
   ```

When the multicast server is started, the new image will be multicast automatically and the set-top box will download it next time it needs to upgrade its software. If the multicast server is already running, you will need to restart it in order for it to transmit the new image.

2.4.2 Creating a signed upgrade image

The following steps detail how to use the `signupgradeimage` script to create a signed software upgrade image. This image is normally called `mc2.mcfs`.

Warning: `mcastbootd.conf` must already exist, and be populated, before you use `signupgradeimage`, otherwise the `signupgradeimage` script will fail saying that the filesystem directory `mc2` does not exist.

The steps outlined here assume that you have set up a multicast system and you have logged in as `root`. The Amino software release contains the engineering customer key that you will need for creating the signed software upgrade image. If you are using your own keys, then you will need to know the pass phrase for your customer key.

1. Configure the files that will form the release, for example, by changing the default settings in the file settings or changing the splash image, and add any extra files to the `flashcontents` file. If you are setting up a system to familiarise yourself with the components, then you will not need to perform any configuration at this stage.

   The `flashcontents` file lists the files that will be included in a software upgrade image, and defines permissions for the file.

   Example flashcontents file

   This shows a small part of a typical `flashcontents` file.

```
# R == file must exist
#      must be read-only
#      must match the checksum in listfile.sig
# W == file may exist
#      must not be executable
#      we don't care about the checksum
# E == file may exist
#      if it does exist:
#      it must be read-only
#     must match the checksum in listfile.sig

R AMINET.img
R libm223.so
R libc223.so
R xfresco
R xfresco.amem
R mkfs.ext2
R fsck.ext2
R tune2fs
R mkfs.xfs
R font_opt.bin
R texttvd
R ttsub
R bbsdk.cfg
R irb_keys.txt
W noformat
W !poweron
W cookies.txt
W history.txt
```

2. Set the `CUSTOMER_KEY` environment variable by entering a command in the following format:

```
export CUSTOMER_KEY=/usr/local/amino/releases/<release_name>/utils/
  keys/amino/KEY.private
```

3. Navigate to the `upgradeimage` directory:

   ```
   cd /usr/local/amino/releases/<release_name>/upgradeimage
   ```

4. Run the `signupgradeimage` script by entering a command in the following format:

   ```
   ./signupgradeimage [-hvz] [-s <pkt-size] [<DI>]
   ```

Argument	Usage
`-h`	Displays help text.
`-v`	Displays version.
`-z`	Compresses the image that is produced (this is only needed for Opera releases, but can also be accepted by other browsers).
`-s`	Sets the size of the multicast image packets to `<pkt-size>` (bytes). This is useful for preventing packet fragmentation in some networks. The default packet size is 4096 bytes.
`<DI>`	Sets the deployment index for the image that is created. The default deployment index is `0` (zero). If you are not controlling upgrades with a deployment index (from example, in trial installations), then you can omit this argument. Otherwise, `<DI>` must be the same or greater than the software currently on the set-top boxes that you want to upgrade with this upgrade image. The exception is for HTTP upgrades where the deployment index must always be greater than the previous deployment index (not the same). You are normally recommended to increment the deployment index by `1` for each upgrade image.
`-n <name>`	Alternative `mcfs` file `<name>` without the `mcfs` extension provided the `mcfs` file is matched within `mcastbootd.conf`.

   For example (recommended for a trial installation):

   ```
   ./signupgradeimage -h
   ```

   or

   ```
   ./signupgradeimage -s 2048
   ```

   or

   ```
   ./signupgradeimage
   ```

   You will be prompted to enter the pass phrase for the key. For the Amino engineering key, the password is `markskey`.

5. The script generates an upgrade image in the current directory. It is normally called `mc2.mcfs`. Copy this file to the `images` directory specified by the `ImageName` setting in the `[Filesystem mc2]` section of the multicast server configuration file (`mcastbootd.conf`).

   For example:

   ```
   cp mc2.mcfs /usr/local/amino/images/
   ```

   If you have multiple `[Filesystem <filename>]` sections in the multicast server configuration file, you will need to change the name of the file to the `<filename>` for the appropriate section, and then copy the file, as above.

When the multicast server is started, the new image will be multicast automatically and the set-top box will download it next time it needs to upgrade its software. Alternatively, you can force the set-top box to upgrade to a specified version. If the multicast server is already running, you will need to restart it in order for it to transmit the new image.

2.4.3 Bootstrap/mc2 packet sizes

When signing a bootstrap/mc2 image, for optimum performance the maximum trasmission unit (MTU– the size (in bytes) of the largest protocol data unit that it can pass onwards) of each image should be equal to the maximum MTU size allowable on the network, allowing for ethernet, IP, UDP or any other overheads.

`Mcastbootd` also adds a 16-byte header to each packet, so this must be taken into account in any calculations.

For example, a DSL PPPoE link might have a frame size of 1518 bytes (say). After headers are taken into account, the maximum payload might be 1492 bytes.

Assuming `mcastbootd` adds a 16-byte header, the maximum allowable packet size is therefore 1476 byes.

For the bootstrap, this is configured in `mcastbootd.conf` with the setting:

```
PacketSize=1476
```

The default value is 1456 bytes, which would be acceptable on this network, but sub-optimal.

For the `mc2` image, the packet size is defined when signing the image with the `signupgradeimage` script. The `-s` option defines the packet size. The default value is 4096 bytes, which will result in fragmentation on this network.

```
signupgradeimage -s 1476
```

It is worth checking with a packet capture tool such as wireshark when running `mcastbootd` on an end-to-end connection that no packet fragmentation is occurring.

2.5 Starting the multicast server

These instructions assume you have set up the DHCP and multicast servers, and that you have copied signed bootstrap and software upgrade images to the location specified in the multicast server configuration file.

1. Navigate to the directory in which software images are located.

   For example:

   ```
   cd /usr/local/amino/images
   ```

2. Enter the following command:

   ```
   mcastbootd
   ```

   ```
   to start the Multicast server in normal mode
   ```

   ```
   mcastbootd -D
   ```

   starts the multicast server in debug mode, and details of its operation are echoed to the console. This is useful to monitor its operation in a trial installation.

The multicast server continuously transmits the software images as a carousel, and when a set-top box needs to install a bootstrap or upgrade image, it joins the carousel and retrieves and installs the required files.

2.6 Amino vendor options in DHCP

The Dynamic Host Configuration Protocol (DHCP) is a protocol used by networked computers (clients) to obtain IP addresses and other parameters such as the default gateway, subnet mask, and IP addresses of DNS servers from a DHCP server. It facilitates access to a network because these settings would otherwise have to be made manually for the client to participate in the network.

In the `dhcpd` configuration file the system integrator can customise the STB, for example, setting the browser to a home page, by adding `option AMINO.homepage "http://www.aminocom.com"` in the fisys class for the particular STB, and similarly for other options.

You will find the following code in the `dhcpd.conf` file:

```
option space AMINO;

option AMINO.address             code  1 = ip-address;
option AMINO.port                code  2 = integer 16;
option AMINO.product             code  3 = text;
option AMINO.option              code  4 = text;
option AMINO.version             code  5 = text;
option AMINO.middleware          code  6 = ip-address;
option AMINO.mw_port             code  7 = integer 16;
option AMINO.homepage            code  8 = text;
option AMINO.dindex              code  9 = integer 32;
option AMINO.dindex_min          code 10 = integer 32;
option AMINO.dindex_page         code 11 = text;
option AMINO.STBrc-mcast-address code 12 = ip-address;
option AMINO.STBrc-mcast-port    code 13 = integer 16;
option AMINO.STBrc-unicast-port  code 14 = integer 16;
option AMINO.local-config        code 15 = text;
option AMINO.timezone            code 16 = text;
option AMINO.middleware2         code 17 = ip-address;
option AMINO.mw_args             code 18 = text;
option AMINO.test_host           code 19 = ip-address;
option AMINO.test_dir            code 20 = text;
option AMINO.recovery_mode       code 21 = integer 8;
option AMINO.mirimon_args        code 22 = text;
```

Standard option 43 Vendor Specific is used to provide the following Amino specific options as shown above:

TAG	NAME	TYPE	MEANING
1	address	ip-address	The multicast FS address, supplied as a 4-byte value.
2	port	2 octet integer	The port number for the group – as a `short`.
3	product	string	The product identifier, sent from the client to the server.
4	option	string	Option string, sent from the client to the server.
5	version	string	Version number sent from the client to the server.
6	middleware	ip-address	The middleware multicast address, supplied as 4-byte value.
7	mw_port	2 octet integer	The middleware port to use on that group (`short`).
8	homepage	string	The browser homepage (`string`).
9	dindex	integer	Deployment index (`int`).
10	dindex_min	integer	Deployment index minimum (`int`): Triggers an automatic upgrade if the deployment index on the box is less than this. If there is no minimum, never automatically update. .

TAG	NAME	TYPE	MEANING
11	dindex_page	string	Deployment index upgrade page (`string`) – where to get the upgrade from.
12	STBrc_mcast_address	ip-address	`STBremoteconf` multicast group (4-byte value).
13	STBrc_mcast_port	2 octet integer	`STBremoteconf` multicast port (`short`).
14	STBrc_ucast_port	2 octet integer	`STBremoteconf` unicast port (`short`)
15	local_config	string	Local config URL (`string`) – location of a config tarball that should be downloaded at boot
16	timezone	string	Timezone string (Either fully specified e.g. "`GMT0BST-1,M3.5.0/01:00,M10.5.0/02:00`" or zoneinfo-style e.g. "`Europe/London`")
17	middleware2	ip-address	The alternative middleware multicast address, supplied as 4-byte value.
18	mw_args	string	CLI arguments string to be passed to the middleware.
19	AMINO INTERNAL		TEST host machine IP-address.
20	AMINO INTERNAL		TEST directory path string.
21	recovery-mode	1 octet integer	Recovery mode. Used by IntActOS to instruct it to store a bootstrap as a recovery image (see `saverecov` and `loadrecov` in `sm- filmod`). Used by bootstraps to determine how/whether to start the recovery UI. 1. Boot on-board recovery bootstrap if present. If not present then run `mboot` and store downloaded bootstrap as on-board recovery bootstrap. 2. Ignore any on-board recovery bootstrap and perform `mboot`. Do not store downloaded bootstrap. 3. Ignore any on-board recovery bootstrap and perform `mboot`. Store newly downloaded bootstrap as recovery image. (**WARNING**: restrict use as it will install recovery image on every reboot.)
22	mw_args	string	Mirimon CLI arguments string

2.6.1 Standard DHCP options supported

The supported options are defined in the following table (options not listed here are ignored by the STB):

The options shown in **BOLD** are requested by the STB in option 55 - Parameter List.

TAG	NAME	MEANING	IntActOS support	Linux support	Used by STB
1	**Subnet Mask**	Subnet Mask Value	Y	Y	Y

TAG	NAME	MEANING	IntActOS support	Linux support	Used by STB
2	**Time Offset**	Time Offset in seconds from UTC	N	Y	Y
3	**Router**	Router addresses	N	Y	Y
6	**Domain Server**	DNS Server Addresses	N	Y	Y
12	**Hostname**	Hostname string	N	Y	Y
15	**Domain Name**	The DNS domain name of the client	N	Y	Y
17	**Root Path**	Path name for root disk	N	Y	Y
23	**Default IP TTL**	Default IP Time To Live	N	Y	Y
28	**Broadcast Address**	Broadcast Address	N	Y	Y
29	**Mask Discovery**	Perform Mask Discovery	N	Y	Y
31	**Router Discovery**	Perform Router Discovery	N	Y	Y
33	**Static Route**	Static Routing Table	N	Y	Y
40	**NIS Domain**	NIS Domain Name	N	Y	Y
41	**NIS Servers**	NIS Server Addresses	N	Y	Y
42	**NTP Servers**	NTP Server Addresses	N	Y	Y
43	**Vendor Specific**	Vendor Specific Information	N	Y	Y
51	Address Time	IP Address Lease Time	N	Y	Y
53	DHCP Msg Type	DHCP Message Type	N	Y	Y
54	DHCP Server ID	DHCP Server Identification	Y	Y	Y
55	Parameter List	Parameter Request List	N	Y	Y
57	DHCP Max Msg Size	DHCP Maximum Message Size	N	Y	Y
58	Renewal Time	DHCP Renewal (T1) Time	N	Y	Y
59	Rebinding Time	DHCP Rebinding (T2) Time	N	Y	Y
60	Class Id	Vendor Class Identifier	Y	Y	Y
61	Client Id	Vendor Client Identifier	Y	Y	Y
77	User-Class	User Class Information	N	Y	N

2.6.2 Class IDs

Standard option 60 - Class ID is included by the STB in all requests it sends. The value that the STB includes for this option depends on whether the request is from the Linux or IntActOS level.

2.6.2.1 Linux

For the Linux images, the value of the string sent is based on the following format:

```
Amino<product><type>
```

where `<product>` is the value of an internal "product" environment variable, for example, AMINET110 and `<type>` is `upgrd` for bootstraps and `fisys` for the main images.

2.6.2.2 IntActOS - multicast versions

For IntActOS multicast versions, the value of the string sent is based on the following format:

```
insecure<product>mboot<romver>
```

or

```
amino<product>mboot<romver>
```

where `<product>` is the value of an internal “platform type” environment variable, for example, AMINET110 and `<romver>` is the version number of the BootROM.

For example:

```
aminoAMINET130mboot1.34
```

2.6.2.3 IntActOS - TFTP versions

For IntActOS TFTP versions, the value of the string sent is:

```
amino
```

or

```
insecure
```

2.6.3 Client IDs

Standard option 61 - Client ID is included by the STB in all requests it sends. The value sent is the Ethernet MAC address of the STB.

2.7 Using symbolic links to reduce multicast server configuration when adding new images

If you are implementing a system in which you will be updating the software image that a set-top box needs to download (for example, to provide images that use new software versions), it can be useful to put in place an image directory structure that uses symbolic links, so that it becomes unnecessary to edit the multicast server configuration file for every new software image. The following instructions outline how to put this in place.

The steps that follow assume you have already set up the multicast server as detailed in "Setting up the multicast server" on page 16.

1. Create a directory structure with a separate directory for each type of set-top box that you are supporting, each with a subdirectory for the current version.

   For example, if you are supporting the AmiNET110 and AmiNET500, and you are currently using software version 0.14.8 for both, create the following directories:

   ```
   <images>/A110/0.14.8
   <images>/A500/0.14.8
   ```

2. Create the bootstrap and upgrade images for each software version, and copy them into the appropriate directories. Note that there is no need for the images for each set-top box type to have different names from each other, as they are already uniquely identified by their directory location. The configuration example below assumes both bootstrap images are called `bootstrap.signed`, and both upgrade images are called `mc2`.
3. Create symbolic links called `current` for each of these directories.

   Example:

   ```
   ln -s <images>/A110/0.14.8 <images>/A110/current
   ln -s <images>/A500/0.14.8 <images>/A500/current
   ```

4. Edit the multicast server configuration file to include sections for each image, using the same multicast IP addresses and ports as specified in the DHCP server configuration. Instead of giving the image location directory by version number, use the `current` symbolic links that you just created. Note that you'll also need to keep the original `mc2` section with dummy settings.

   Example configuration file:

```
# Configuration file: Wed Sept 19 10:25:51 2007

[Server]
LogLevel=4
MulticastTTL=5
ImageDir=/usr/local/amino/images

[Image A110Bootstrap]
MulticastIPAddress=225.50.50.50          <-This is an example value.
MulticastUDPPort=11111
FileName=110/current/bootstrap.signed
Description=Linux bootstrap image
ImageType=1
SerialNumber=1
PacketSize=1456
CycleTime=0

[Image A500Bootstrap]
MulticastIPAddress=225.50.50.52          <-This is an example value.
MulticastUDPPort=11111
FileName=A500/current/bootstrap.signed
Description=Linux bootstrap image
ImageType=1
SerialNumber=1
PacketSize=1456
CycleTime=0

[Filesystem mc2]
MulticastIPAddress=225.50.50.45         <-This is an example value.
MulticastUDPPort=11111
ImageName=mc2
Description=DUMMY upgrade filesystem
SerialNumber=149
DirsPerCycle=128
DataRate=512

[Filesystem A110Upgrade]
MulticastIPAddress=225.50.50.51         <-This is an example value.
MulticastUDPPort=11111
ImageName=A110/current/mc2
Description=upgrade filesystem
SerialNumber=6
DirsPerCycle=128
DataRate=512
```

```
[Filesystem A500Upgrade]
MulticastIPAddress=225.50.50.53          <-This is an example value.
MulticastUDPPort=11111
ImageName=A500/current/mc2
Description=upgrade filesystem
SerialNumber=6
DirsPerCycle=128
DataRate=512
```

5. Save your changes and restart the multicast server.

Next time you need to add a new software image for the platforms you have set up:

1. Create a subdirectory with the new version number.

   For example, if you're adding images for version 0.14.8, for the AmiNET110, create the following directory:

   `<images>/A110/0.14.8`

2. Change the `current` symbolic link to this new directory instead of the `0.14.0` one:

   `ln -s <images>/A110/0.14.8 <images>/bootstrap/A110/current`

3. The multicast server configuration sections already point to the `current` directory, so all you need to do to start transmitting the new images is restart the multicast server.

Chapter 3—Upgrading the STB

The Amino multicast upgrade system offers various means of instructing set-top boxes to replace their software images.

3.1 Upgrade mechanisms

- **STBremoteconf** – remote configuration tool that enables you to send commands such as changing basic configuration settings or initiating software upgrade.
- **Management pages** – local configuration pages, using an Amino IR keyboard and television display to change configuration settings and simple commands such as rebooting and initiating software upgrades.
- **DHCP server configuration** – DHCP server configuration specifies where set-top boxes “go” to retrieve software images transmitted by the multicast server, it can also be configured to force upgrades by specifying minimum deployment index numbers for software versions that the set-top box must be running. DHCP server and Multicast server together also enable the set-top box to connect and retrieve new software images when required (for example, if the software on the set-top box is corrupted).
- The **JMACX API** offers functions for controlling a range of set-top box operational areas, including software upgrades. This manual does not cover the use of the JMACX functions, please see the Amino *JMACX API Specification* for more information.

As detailed in the table below, the various upgrade mechanisms are each suited to particular circumstances and contexts. For example, making images available by defining upgrade groups and ports in the DHCP server configuration enables automatic upgrades when the set-top box requires them, but `STBremoteconf` can be used to force an upgrade at a required time.

In the table below, **reflash** refers to deleting all the contents of the NAND Flash and completely replacing the main software image using the bootstrap process. All configuration options, except those stored in NOR Flash, will be replaced with the configuration contained in the new software image, **upgrade** refers to replacing just the main software image. In this case, some of the configuration files, for example, the user settings, may be preserved. This can be controlled when the software image is built.

	STBremoteconf	**Management pages**	**DHCP server configuration**
Contexts			
Remote	Y		Y
Local		Y	
Single STB	Y	Y	Y
Large deployment	Y	Y	Y

	STBremoteconf	Management pages	DHCP server configuration
Test settings	Y[a]		Y
Automatic			Y
Manual	Y	Y	
Operation			
upgrade software	Y	Y	Y
reflash	Y	Y	
upgrade to min. DI number			Y
upgrade to min. version number	Y		

a. Not recommended: requires all STBs to be powered.

3.1.1 Using STBremoteconf to upgrade software

`STBremoteconf` provides a mechanism for remotely configuring and controlling a local network of AmiNET set-top boxes. It can be run on a Linux/Unix command line by sending commands individually to one or more set-top boxes, or commands can be collected into a script to save time. Commands are signed when they are created, so that the set-top box can confirm that the commands have been sent by an authorised source.

Example functions include rebooting, modifying output modes or updating software, as well as viewing information about the set-top box.

The `UPGRADEMCAST` and `UPGRADEMCAST_VER` commands enable you to instruct a set-top box to upgrade. See the Amino *Set-top Box Configuration Guide* for further details.

3.1.2 Using the Management pages to upgrade software

Set-top boxes with built-in browsers include HTML configuration pages, displayed on the television screen, to enable local access and editing of configuration areas. The pages are divided into an administrator area (**Management** pages) and user area (**Preferences** pages).

The set-top box **Preferences** pages are a simple interface that enables users to change basic set-top box configuration areas such as language, subtitle and keyboard set-up. The pages are accessed locally via an Amino InfraRed (IR) remote control (or keyboard). The **Preferences** pages are designed for use by end-users.

The set-top box **Management** pages are a simple interface that enable administrators to change set-top box configuration areas such as networking, channel list and browser set-up, as well as initiate basic operations such as rebooting and updating software. **Management** pages allow access to advanced configuration functions and are designed for use by administrators rather than end-users. The pages are password-protected and accessed locally via an Amino IR keyboard. Limited functionality is also supported via the remote control.

You can instruct a set-top box to upgrade or reflash from the **Update Software** Management page. See the Amino *Set-top Box Configuration Guide* for further details.

3.1.3 Using the DHCP server to upgrade software

The DHCP server and multicast server enable the set-top box to retrieve software upgrades when it needs them. You can also configure the DHCP server so that the set-top box automatically upgrades to a specific software version, even if its current software is running normally. See "Automatic upgrading" on page 43 for details of how to enable automatic upgrades.

3.2 Upgrading the software

This section describes a laboratory-based upgrade process and is a sub-set of the instructions found in Chapter 2, "Installing the software" .

Note: The following instructions assume the use of the Debian operating system.

Note: The commands in this section require you to be logged on as a superuser, or to use the `sudo` command.

1. Copy the required `.tgz` file to the `/usr/local/amino/releases` directory.
2. Navigate to the releases directory `/usr/local/amino/releases` and unpack the Amino software release using:

   ```
   tar zxvf <archive_name>
   ```
3. Copy the multicast server binary to a directory in the system path, using:

   ```
   cp <release_name>/server/mcastbootd /usr/local/bin
   ```
4. Copy the contents of the `utils` directory to the same location, using:

   ```
   cp <release-name>/utils/* /usr/local/bin
   ```
5. Copy the multicast server configuration file to the `/etc` directory, using:

   ```
   cp <release_name>/server/mcastbootd.conf /etc
   ```

3.2.1 Create a signed bootstrap image

1. Set the `CUSTOMER_KEY` environment variable by entering a command in the following format:

   ```
   export CUSTOMER_KEY=/usr/local/amino/releases/<release_name>/utils/
   keys/amino/KEY.private
   ```
2. Navigate to the `bootstrap` directory:

   ```
   cd /usr/local/amino/releases/<release_name>/bootstrap
   ```
3. Run the `signbootstrap` script by entering the following command:

   ```
   ./signbootstrap
   ```

   You will be prompted to enter the password for the key. For the Amino engineering key, the password is `markskey`
4. The script generates a `bootstrap.signed` file in the `bootstrap` directory. Copy this file to the `/usr/local/amino/images` directory specified by the `FileName` setting in the `[Image <image_name>]` section of the multicast server configuration file (`mcastbootd.conf`).

   For example:

   ```
   cp bootstrap.signed /usr/local/amino/images/
   ```

When the multicast server is started, the new image will be multicast automatically and the set-top box will download it next time it needs to upgrade its software. If the multicast server is already running, you will need to restart it in order for it to transmit the new image.

3.2.2 Create a signed upgrade image

The following steps detail how to use the `signupgradeimage` script to create a signed software upgrade image. This image is normally called `mc2.mcfs`.

1. Configure the files that will form the release, for example, by changing the default settings in the file settings or changing the splash image, and add any extra files to the `flashcontents` file.
2. Set the `CUSTOMER_KEY` environment variable by entering a command in the following format:

```
export CUSTOMER_KEY=/usr/local/amino/releases/<release_name>/utils/
  keys/amino/KEY.private
```

3. Navigate to the `upgradeimage` directory:

   ```
   cd /usr/local/amino/releases/<release_name>/upgradeimage
   ```

4. Run the `signupgradeimage` script by using:

   ```
   ./signupgradeimage
   ```

   You will be prompted to enter the pass phrase for the key. For the Amino engineering key, the password is `markskey`.

5. The script generates `mc2.mcfs` in the current directory. Copy this file to the `images` directory specified by the `ImageName` setting in the `[Filesystem mc2]` section of the multicast server configuration file (`mcastbootd.conf`).

   For example:

   ```
   cp mc2.mcfs /usr/local/amino/images/
   ```

   If you have multiple `[Filesystem <filename>]` sections in the multicast server configuration file, you will need to change the name of the file to the `<filename>` for the appropriate section, and then copy the file, as above.

When the multicast server is started, the new image will be multicast automatically and the set-top box will download it next time it needs to upgrade its software. If the multicast server is already running, you will need to restart it in order for it to transmit the new image.

3.2.3 Start the multicast server

1. Navigate to the directory in which software images are located.

   For example:

   ```
   cd /usr/local/amino/images
   ```

2. Enter the following command:

   ```
   mcastbootd
   ```

   ```
   to start the Multicast server in normal mode
   ```

   ```
   mcastbootd -D
   ```

   starts the multicast server in debug mode, and details of its operation are echoed to the console. This is useful to monitor its operation in a trial installation.

The multicast server continuously transmits the software images as a carousel, and when a set-top box needs to install a bootstrap or upgrade image, it joins the carousel and retrieves and installs the required files.

3.3 About set-top box operation

The sections that follow provide detail about the set-top box operation states, including how set-top box operation is affected if one of the actions that the set-top box executes is not successful. Information about operational errors is indicated by flashing codes used by the LED on the set-top box. See Appendix D, ”Troubleshooting“ for a list of these flashing codes.

Set-top box operation is driven by a defined set of states. How a set-top box operates depends on its current state. The state a set-top box is in depends on the existence of a current software image in its NAND Flash, or whether it can load that software successfully. Each time a set-top box enters a new

state, it sends a DHCP request that includes an appropriate string to identify its current state, as defined by Amino vendor extensions added to the DHCP server's configuration file.

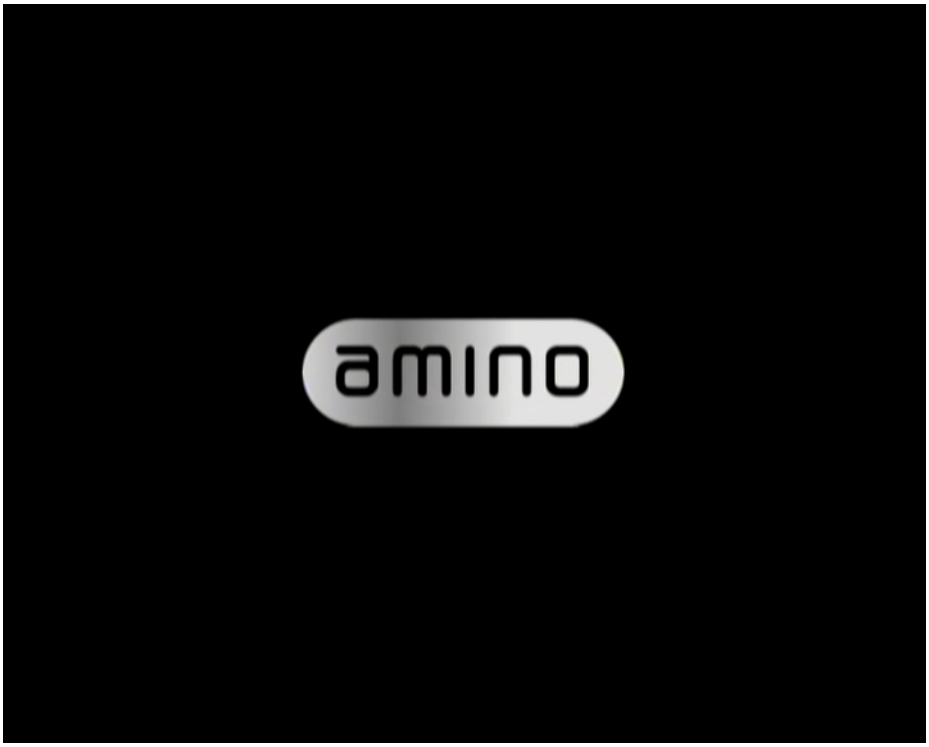


Figure 3.1 Normal operation splash screen. Displayed when powering up.

Figure 3.2 Splash screen for builds later than 0.15.3/0.10.22 on the ST platform, or 0.16.0 for the TI platform. Displayed when powering up.

When a set-top box is powered, it attempts to execute software from its NAND Flash. If this is successful, the set-top box enters its normal running state. See Filesystem (fisys) state, otherwise, the set-top box enters its bootstrap state.

The operation is summarised in Figure 3.3.

3.3.1 Static DHCP settings

The descriptions that follow outline operation in cases where the set-top box relies on the DHCP server for its network and other settings. Where these are defined statically in the set-top box's configuration, they can either be used as a fall-back – if communication with the DHCP server fails – or they can be used in place of DHCP communication. How these static settings are used depends on configuration.

Note: Operation when the NAND Flash does not contain valid software is enabled by the Bootloader stored permanently in the NOR Flash.

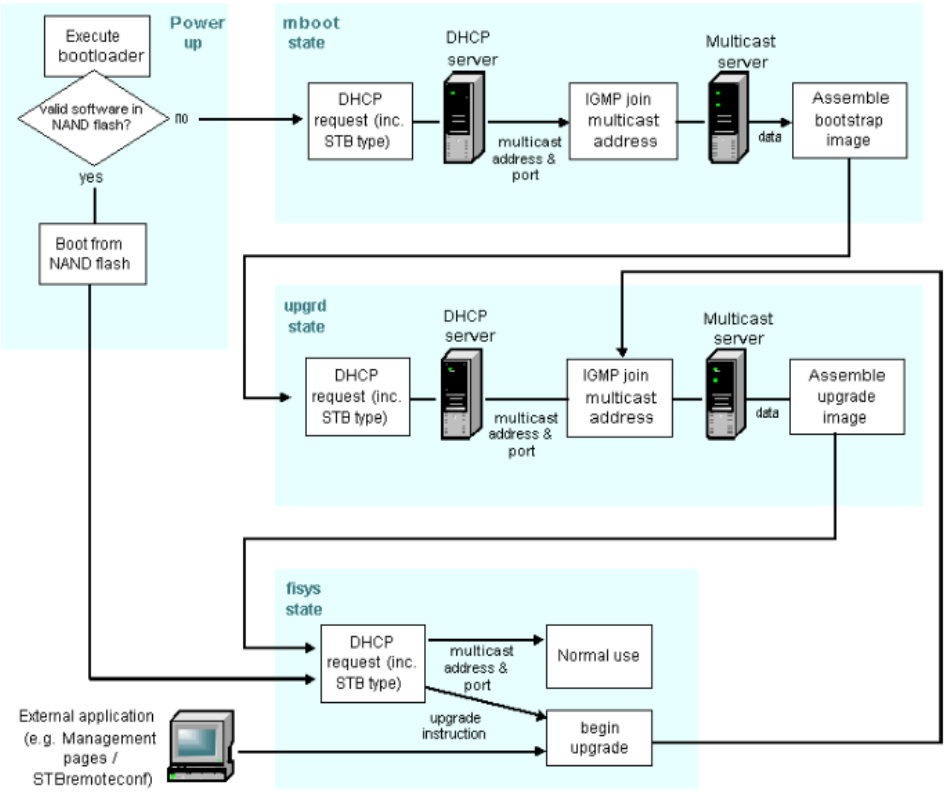


Figure 3.3 Dataflow diagram

3.4 Operational states

The following operational states are defined.

3.4.1 Bootstrap (mboot) state

When the set-top box is first turned on (not from standby), it inspects its NAND Flash for valid software. If no valid software is found, the set-top box enters the multicast bootstrap (`mboot`) state. In this state, it can securely download a bootstrap image from the multicast server.

When a set-top box is in bootstrap state, it includes `mboot` state in the message it uses to communicate with a DHCP server.

3.4.1.1 Operation in the bootstrap (mboot) state

During bootstrap state, the default boot logo is displayed on the screen. If the logo has not been customised, it will be similar to Figure 3.4:

Figure 3.4 Default boot logo

The operational sequence on entering this state is as follows:

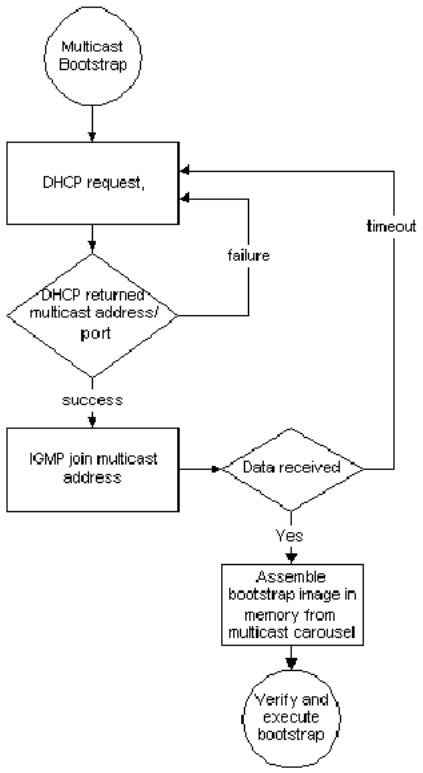


Figure 3.5 Operational sequence (bootstrap state)

If there is a failure during this state, a sequence of flashes of the LED on the set-top box indicates the reason.

3.4.1.2 Send DHCP request

The set-top box sends a message to the DHCP server, including the current `mboot` state and a string identifying the hardware platform. The DHCP server compares the state and the platform string against classes defined in its configuration file. If it cannot find such a class, then it will be unable to provide the mulitcast information that the set-top box needs, but will supply the IP address.

3.4.1.3 DHCP response

When the DHCP server identifies an appropriate class, it responds to the set-top box with network settings and the vendor extensions defined in that class. These vendor extensions are normally an IP address and port for retrieving a bootstrap image that is being transmitted by the multicast server.

If the set-top box does not receive a response from the DHCP server, it retries sending the DHCP request a number of times (the number of retries is specified in the code). If this is still not successful, it either uses values from its configuration (if these have been set) or reboots itself, returning to the initial powered state.

3.4.1.4 Download image

The set-top box connects to the multicast carousel with the details it has received and downloads the packets for a bootstrap image. If the set-top box does not manage to download an image from the multicast server within a defined timeout, it reboots itself, returning to the initial powered state.

3.4.1.5 Assemble, verify and execute image

The set-top box verifies the new bootstrap image by using the public customer key it holds to check the digital signature that the image is signed with. If the software is successfully verified, the set-top box executes it and stores it in its NAND Flash.

If the set-top box is unable to verify or execute the software it has downloaded, the software is reset and the set-top box either reboots itself or re-enters the bootstrap state, depending on what stage the process fails at. The debug output will contains details of why the failure occurred.

3.4.2 Upgrade (upgrd) state

Once the set-top box has found, downloaded and installed a valid bootstrap image, it enters the (`upgrd`) upgrade state. The set-top box will also enter this state in response to a manual upgrade request (for example, one initiated via `STBremoteconf`). In this state, all non-essential applications are closed, and the set-top box can download and execute a new software upgrade image.

When a set-top box is in upgrade state, it includes `upgrd` state in the message it uses to communicate with a DHCP server.

3.4.2.1 Operation in the upgrade (upgrd) state

In the upgrade state, the loading screen is displayed, which contains just the Amino logo.

Figure 3.6 Upgrading logo

For builds later than 0.15.3/0.10.22 on the ST platform or 0.16.0 for the TI platform the following screen is displayed while upgrading.

Figure 3.7 Upgrading logo for recent builds

The operational sequence on entering this state is as follows:

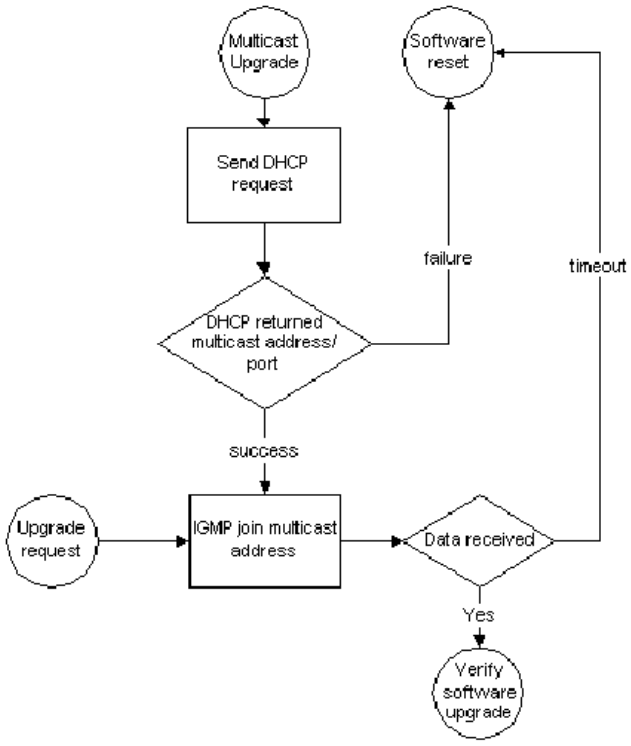


Figure 3.8 Operational sequence (upgrade state)

If there is a failure during this state, a sequence of flashes of the LED on the set-top box indicates the reason. See Appendix D, ”Troubleshooting“ for a list of these flashing codes.

3.4.2.2 Send DHCP request

Once the set-top box has successfully loaded a bootstrap image, it enters the upgrade state.The set-top box also enters this state if a manual upgrade request is initiated (for example, via `STBremoteconf`). The set-top box sends a message to the DHCP server, including `upgrd` state and a string identifying the hardware platform.

3.4.2.3 DHCP response

The DHCP server compares the state and the platform string against classes defined in its configuration file. When the DHCP server identifies an appropriate class, it responds to the set-top box with network settings and the vendor extensions defined in that class. These vendor extensions are normally an IP address and port for retrieving an upgrade image that is being transmitted by the multicast server. If it does not find a class, then it cannot provide the information that the set-top box needs.

If the set-top box does not receive a response from the DHCP server after a number of retries, it tries to move to the next stage using the settings in its configuration (that is, using null values for the multicast IP address and port, if these have not been set in the configuration – so the download will not succeed).

3.4.2.4 Download image

The set-top box connects to the multicast carousel with the details it has received and downloads the packets for an upgrade image. If it fails to download the image within a defined timeout, the set-top box reboots, and returns to the initial powered state.

3.4.2.5 Assemble, verify and execute image

The set-top box verifies the new image by using the public customer key it holds to check the digital signature that the image is signed with. It also compares the deployment index (DI) of the software against its current index. If the software is successfully verified, the set-top box stores it in its NAND Flash and executes it.

If the software is not successfully verified and executed, the set-top box resets itself. How it behaves at this stage depends on what stage of the verification process it failed at: it can either reboot or re-enter the upgrade state.

Note: At this point the NAND Flash could potentially be corrupted by a power outage or other failure. If this happens, the set-top box will detect it and return to the bootstrap state.

3.4.3 Filesystem (fisys) state

Once the set-top box has found and executed a valid software image from the NAND Flash, it enters the filesystem (`fisys`) state. In this state, the set-top box is running and available for normal use. An administrator can initiate a software upgrade using one of the management tools (for example, `STBremoteconf`).

3.4.3.1 Operation in the filesystem (fisys) state

In the filesystem state, normal operation is enabled. When the set-top box first enters this state, it shows the page set as the default home page. For example, for a standard Amino build with the ANT Fresco browser, a screen similar to the following is displayed:

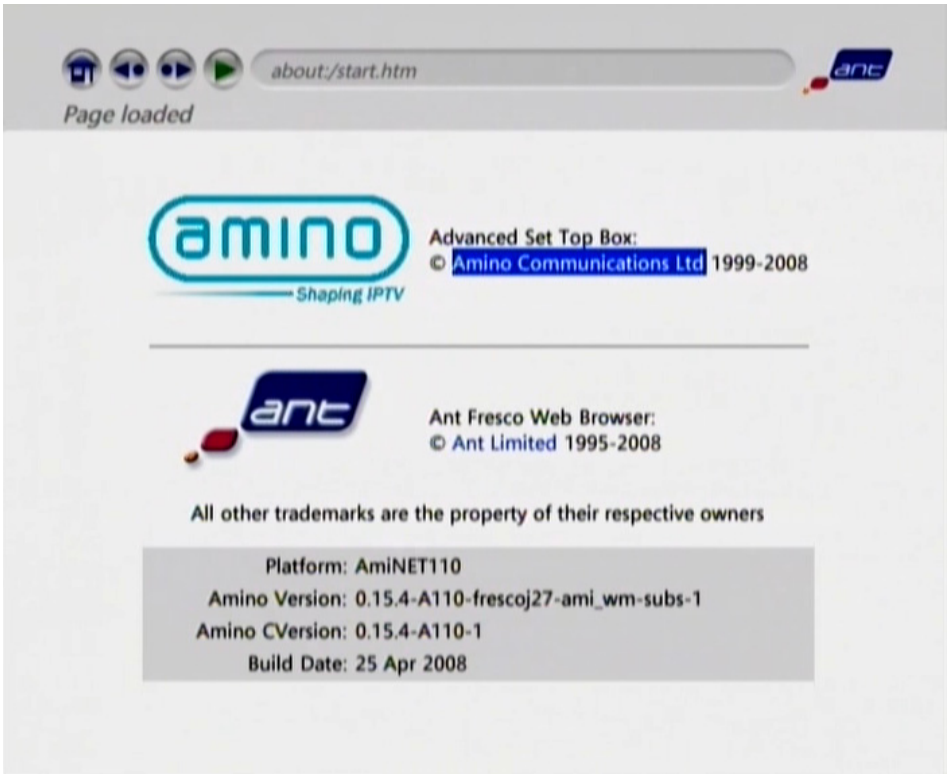


Figure 3.9 ANT Fresco browser screen

The operational sequence on entering this state is as follows. If there is a failure during this state, a sequence of flashes of the LED on the set-top box indicates the reason. Refer to Appendix A, ”LED flashing codes“ for details of these flashing codes.

3.4.3.2 Send DHCP request

When the set-top box successfully loads both a bootstrap and software upgrade image, it enters the filesystem state. The set-top box sends a message to the DHCP server, including `fisys` state and a string identifying the hardware platform. The DHCP server compares the state and the platform string against classes defined in its configuration file. If it does not find a class, then it is unable to supply the information that the set-top box needs.

3.4.3.3 DHCP response

When the DHCP server identifies an appropriate class, it responds to the set-top box, with network settings and any vendor extensions defined in that class. For the filesystem state, vendor extensions are optional, but they can include an IP address and port for retrieving an upgrade image that is being transmitted by the Multicast server as well as other options such as middleware server details and minimum software deployment index (DI) that the set-top box must be running.

If the set-top box does not receive a response from the DHCP server after a number of retries, it reboots, returning to its initial powered state.

3.4.3.4 Initiate upgrade or normal use

If the minimum deployment index supplied is greater than the set-top box’s current index, the set-top box enters the upgrade state, to download a new software version. The set-top box must have been supplied with multicast IP address and port in order for this upgrade to be successful. If the upgrade fails, then the set-top box returns to normal operation, using its current software version.

If the set-top box receives an upgrade instruction from an external application, the set-top box enters the upgrade state, using the multicast IP address and port supplied in the upgrade instruction. Again, if the upgrade fails, the set-top box returns to normal operation without upgrading its software.

Otherwise, the set-top box is available for normal operation.

When a set-top box is in filesystem state, it includes `fisys` state in the message it uses to communicate with a DHCP server.

3.5 Set-top box configuration

The multicast system offers a range of tools for configuring set-top boxes, depending on factors such as whether you are testing new settings or rolling out configuration for a software upgrade and whether you are configuring the box remotely or locally. For an overview of configuration methods, see the Amino *Set-top box Configuration Guide*.

3.6 Automatic upgrading

Automatic upgrading to a specified minimum deployment index is set up in the DHCP server. The following steps outline how to implement this system:

Note: These steps assume you have already set up a multicast upgrade system and the software image is in the location specified in the multicast server's configuration.

1. Open the DHCP server configuration file (`dhcpd.conf`).
2. Find the Amino options (under `option space AMINO;`) and ensure that the following line is included (add it if it is not there already):

   ```
   option AMINO.dindex_min code 10=integer 32;
   ```

3. Find the `fisys` class for the set-top box type that you want to force to upgrade, and under `vendor-option-space AMINO;` add a line in the following format:

   ```
   option AMINO.dindex_min <minimum_DI>;
   ```

   For example, if you want set-top boxes to be running software with the minimum deployment index of 5, the line should be as follows:

   ```
   option AMINO.dindex_min 5;
   ```

4. Also ensure that the following lines are present, and specify the correct multicast IP address and port:

   ```
   option AMINO.address <IPaddress>;
   option AMINO.port <port_number>;
   ```

   The `fisys` class you are editing should now look similar to the following:

   ```
   class "110 normal"
   {
   match if (substring( option vendor-encapsulated-options, 2,
         9)="aminet110") and (substring( option vendor-encapsulated-
         options,13, 5)="fisys");

   vendor-option-space AMINO;
   option AMINO.dindex_min 5;
   option AMINO.address 123.4.5.67;
   option AMINO.port 11111;
   }
   ```

5. Save your changes and restart the DHCP server. Also ensure the multicast server is running.

The system is now set up. Next time a set-top box of the configured type boots, it will compare its current deployment index with the one you specified, and will upgrade to the specified software image if its current deployment index is lower than the one in the DHCP server configuration.

3.6.1 To test automatic upgrading

Once you have implemented automatic upgrading using a specified minimum deployment index to force upgrades, you can test the system. The following steps assume that Telnet (or other remote log-in client) is enabled in the set-top box.

1 Check the deployment index number of set-top box you are going to test with. You can do this from the command line by Telnetting to the set-top box and entering the following command (the current number must be lower than the one you want to upgrade to):

```
libconfig-get NORFLASH.DI
```

2. Power up (or reboot) the set-top box.

   The loading screen is displayed (the default image has **Loading** followed by **Upgrading, do not unplug!**

3. When the upgrade is finished, you can check that the set-top box is now running the new software version and you can check the deployment index, as in step 1 (it should now have the deployment index specified in the DHCP server configuration file).

3.6.2 Setting set-top boxes with static network and multicast upgrade settings

Some DHCP servers do not support the addition of the vendor extensions that this multicast upgrade system uses to tell set-top boxes the IP address and port to use for booting and retrieving software images (for example, `AMINO.address` and `AMINO.port`). In these cases, the DHCP server can be used just to assign an IP address to the set-top box, and you can configure the addresses and ports to use for software upgrades as static values in the set-top box's configuration.

In other installations, it may be useful to set all network settings statically, rather than retrieving them from the DHCP server.

In both cases, the static values are normally defined in the NOR Flash, which you can edit in the following ways:

- `libconfig` commands in upgrade script (recommended method)
- `libconfig` commands via remote log-in tool (for example, Telnet)
- via the Management pages (Network Configuration page)
- JMACX calls in HTML page (not generally recommended, for security reasons)

In some cases, the settings may be defined in the `netconf` file. See the Amino *Set-Top Box Configuration Guide* for more information.

The recommended way to set static multicast upgrade and network values for multiple set-top boxes is to add `libconfig` commands to the upgrade script that is included as part of a software image. The script executes automatically when the new software image is loaded to the set-top box, and is then deleted.

3.6.3 Network and multicast upgrade configuration settings

To set static network and multicast upgrade settings, you need to edit settings in the NOR Flash. These settings can be accessed via various configuration tools (for example, `libconfig` commands in an `upgrade.sh` script or over Telnet, JMACX calls or through Management pages). For details of how to

use the tools available to change settings, see the Amino *JMACX API Specification* and the Amino *Set-Top Box Configuration Guide*.

The NOR Flash settings that configure static network and multicast upgrade values are listed in the following sections.

Note: In some cases, these settings may be defined in the `netconf` file. See the Amino *Set-Top Box Configuration Guide* for more information.

3.6.3.1 To set static multicast upgrade values

```
MULTICAST_BOOTSTRAP_GROUP
MULTICAST_BOOTSTRAP_PORT
MULTICAST_UPGRADE_GROUP
MULTICAST_UPGRADE_PORT
MULTICAST_FILESYSTEM_GROUP
MULTICAST_FILESYSTEM_PORT
```

Note: The bootstrap, upgrade and filesystem groups must be set with a valid multicast addresses in order for multicast upgrades to work. If an address is invalid, the set-top box tries to retrieve an IP address via DHCP, if this is not successful – or if the address that the set-top box retrieves via DHCP is also invalid – the set-top box reboots.

3.6.3.2 To set a static IP address

```
IPADDR
```

Other network settings

```
NETMASK
GATEWAY
DNS
TIME_SERVER
```

Required vs. optional settings

The list of values that must be defined statically depends on what functionality your DHCP server supports. The following guidelines apply:

- **Required multicast upgrade settings**

  If the DHCP server is being used to provide network settings, but not the multicast upgrade settings (that is, the DHCP server does not support the Amino vendor extensions), the bootstrap and upgrade settings are required. The filesystem settings are normally optional, although this will depend on the functionality you want to enable in the system.
- **Required network settings**

  If the DHCP server is not being used to provide network settings, then most of the values listed are required, only `DNS` and `TIME_SERVER` are optional.

Example upgrade.sh script to set static multicast upgrade values

The recommended way to set static network and multicast upgrade values for multiple set-top boxes is to add `libconfig` commands to the upgrade script that is added to the image components used to create a software upgrade image. When the new upgrade image is loaded to the set-top box, the script executes automatically, and is then deleted.

Important: The software image containing this upgrade script must be installed on the set-top boxes before they are installed on a network that requires the static settings. This is because the set-top box requires these network settings in order to retrieve the new software image.

The following script uses `libconfig` commands to set values for some of the multicast upgrade settings (bootstrap and upgrade ports and IP addresses) in the NOR Flash:

```
#!/bin/sh
libconfig-set NOFLASH.MULTICAST_BOOTSTRAP_GROUP 239.255.1.1
libconfig-set NORFLASH.MULTICAST_BOOTSTRAP_PORT 11111
libconfig-set NOFLASH.MULTICAST_UPGRADE_GROUP 239.255.1.2
libconfig-set NORFLASH.MULTICAST_UPGRADE_PORT 11111
```

3.7 Automating multicast server start-up

By default, the multicast server does not start when you start the server it is installed on. You can enable automatic start-up by creating an `mcastbootd` script in `/etc/init.d`, and running the following command to start the service:`chkconfig mcastbootd on`. The multicast server will be started automatically on the next reboot.

The following example start script works with Redhat Linux.

```
#!/bin/bash
#
# chkconfig: 2345 55 25
# description: Amino Multicast Daemon

# processname: mcastbootd
# config: /etc/mcastbootd.conf

# Source function library.
[ -f /etc/rc.d/init.d/functions ] || exit 0
. /etc/rc.d/init.d/functions

# getting the Mcastbootd options
. /etc/sysconfig/mcastbootd

RETVAL=0
start() {
    echo -n $"Starting Mcastbootd server services: "
    mcastbootd $MCASTBOOTD_ARGS
    RETVAL=$?
    echo
    [ $RETVAL -eq 0 ] && touch /var/lock/subsys/mcastbootd
    return $RETVAL
}

stop() {
    echo -n $"Stopping Mcastbootd server services: "
    killproc mcastbootd
    RETVAL=$?
    [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/mcastbootd echo
    return $RETVAL
}

# See how we were called.
case "$1" in
    start)
       start ;;
```

```
    stop)
       stop ;;
    status)
       status mcastbootd ;;
    restart|reload)
       stop start ;;

    condrestart)
       if [ -f /var/lock/subsys/mcastbootd ]; then
       stop
       start
       fi
       ;;
    *)
       echo $"Usage: $0 {start|stop|status|restart|reload|condrestart}"
       exit 1

esac
exit $RETVAL
```

3.8 Using the deployment index for automatic upgrade

In normal operation, the set-top box downloads new software from the multicast server when there is a problem with its current software or when an upgrade command instructs it to (for example, from `STBremoteconf`, the Management pages or a JMACX call). Using the deployment index, it is also possible to configure the multicast upgrade system to force set-top boxes to upgrade to a particular software version (as identified by the software's deployment index).

3.8.1 How it works

The set-top box stores the deployment index of its current software version in the NOR Flash.

A new deployment index is specified when a new signed software upgrade image is created (with the `signupgradeimage` script). The deployment index is an integer between `0` and `65535`.

When the set-top box receives the new software version, it checks the deployment index against the number in its NOR Flash, and accepts the software upgrade if the new number is the same or greater, and changes the deployment index in its NOR Flash accordingly. Otherwise it rejects it.

Note: For trial versions and for early stages of a project, we recommend you use a deployment index of `0` (zero) all the time. This means that you do not need to specify the deployment index when you create a new upgrade image. Once the deployment is live, the service provider should increase the deployment index by 1 when asked to produce a new version by the software provider (for example, the CA or middleware vendor).

3.8.2 Deployment indices and software versioning

The deployment index mechanism is completely separate from software version numbers. This means that if necessary a service provider can downgrade from a current software version to an older one, as long as they create a software upgrade image with a higher deployment index than the current software version has.

3.8.3 Migrating from pre-DI to DI-capable software

All set-top boxes are manufactured with an initial deployment index of `0` (zero) in their NOR Flash. All you need to do to start using the deployment index mechanism in a system that currently does not implement this control, is to increase the deployment index to `1` in the next software upgrade image that you create.

3.8.4 Setting up automatic upgrade to minimum deployment index

The deployment index mechanism can be used to force set-top boxes to upgrade to a specified minimum deployment index, using Amino DHCP vendor extensions. The minimum index number is added to the DHCP configuration. When the set-top box boots up and contacts the DHCP server, the DHCP server includes this deployment index in its response, along with the multicast upgrade address and port for the upgrade image. The set-top box checks this index against the deployment index in its NOR Flash, and upgrades immediately if its index is lower than the specified value.

3.8.4.1 Resetting the deployment index

For security reasons Amino do not provide a method of resetting the DI remotely on an STB. However, it is possible to retrieve the current DI through telnet.

Telnet to an STB, and run this command:

```
libconfig-get NORFLASH.DI
```

If, having obtained the DI, the software download is configured with a DI higher than the current value and the STBs all reflashed, they will all be on the same, new, DI.

3.8.5 To implement automatic upgrading

Automatic upgrading to a specified minimum deployment index is set up in the DHCP server. The following steps outline how to implement this system.

Note: The se steps assume you have already set up a multicast upgrade system and the software image is in the location specified in the multicast server's configuration.

1. Open the DHCP server configuration file (`dhcpd.conf`).
2. Find the Amino options (under `option space AMINO;`) and ensure that the following line is included (add it if it is not there already):

   ```
   option AMINO.dindex_min code 10=integer 32;
   ```

3. Find the `fisys` class for the set-top box type that you want to force to upgrade, and under `vendor-option-space AMINO;` add a line in the following format:

   ```
   option AMINO.dindex_min <minimum_DI>;
   ```

   For example, if you want set-top boxes to be running software with the minimum deployment index of 5, the line should be as follows:

   ```
   option AMINO.dindex_min 5;
   ```

4. Also ensure that the following lines are present, and specify the correct multicast IP address and port:

   ```
   option AMINO.address <IPaddress>;
   option AMINO.port <port_number>;
   ```

   The `fisys` class you are editing should now look similar to the following:

   ```
   class "110 normal"
   {
   ```

```
match if (substring( option vendor-encapsulated-options, 2,
      9)=”aminet110”) and (substring( option vendor-encapsulated-
      options,13, 5)=”fisys”);

vendor-option-space AMINO;
option AMINO.dindex_min 5;
option AMINO.address 123.4.5.67;
option AMINO.port 11111;
}
```

5. Save your changes and restart the DHCP server. Also ensure the multicast server is running.

The system is now set up. Next time a set-top box of the configured type boots, it will compare its current deployment index with the one you specified, and will upgrade to the specified software image if its current deployment index is lower than the one in the DHCP server configuration.

Chapter 4—STB Security Features

The software on a set-top box is protected by a combination of security measures:

- Security keys
- Deployment Index (DI)
- Password-protected access to management tools

4.1 Security keys

Software is protected by cryptographic public-private keys and digital signatures. This protection extends from initial booting through to uploading of new software images. Images and commands are digitally signed before they are sent to a set-top box, and the set-top box checks this signature before it loads the new software or completes the operation requested.

Amino STBs use 1024-bit RSA signing to verify code and commands. Each STB has three keys in a heirarchy, a master key, a customer key and a configuration or STBRC key. For each "key" in a software distribution there are in fact two keys, a public key (used by the set-top box to verify images and commands), which is stored on the STB and a private key that is kept secret, either here at Amino, at the customer (that is, the telephone or broadband service provider), or at our CA partner NDS in Israel. The keys used can be either the Amino engineering keys – used for trial purposes – or customer specific keys, depending on customer requirements.

4.1.1 Hierarchy

The use of a series of different keys allows us to achieve the above aims. The hierarchy used is shown in Figure 4.10.

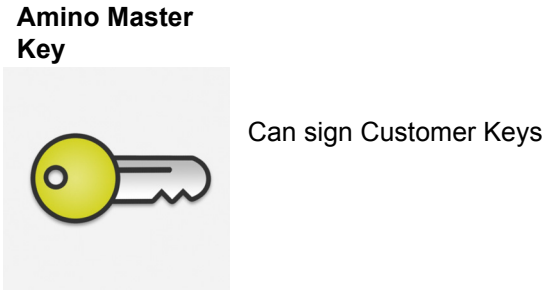


Figure 4.10 Key hierarchy

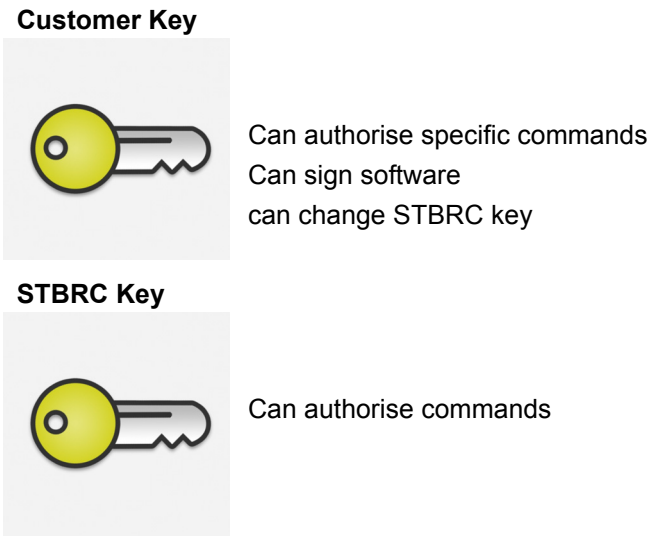


Figure 4.10 Key hierarchy

The final implementation makes use of three 1024-bit RSA keys. The private part of each is encrypted using with the Blowfish cipher using a secret passphrase. The individual keys are described in more detail below.

The public part of each key is stored in the NOR Flash.

4.1.1.1 Amino Master key

This key is the most secure. It is used to sign customer keys and the code in the BootRom.

The Amino Master private key is kept in a secure environment and only two senior, long-term, employees know the passphrase.

The public part of the key is programmed into each individual STB at time of manufacture and is not intended to be changed.

The Master key is used exclusively by Amino to validate the customer key. It is held securely and is only available to Amino.

The Master public key is stored in the lockable (OTP) region of the NOR Flash ROM, it is only used infrequently. At startup it is used to verify that the Customer public key has been signed with the Master private key and that the IntActOS code has been signed with the Master private key. It is also used sometimes to verify bootstraps and software downloads before executing them.

4.1.1.2 Customer key

This key is used to sign software images and specific remote commands.

A different customer key is generated for each customer by Amino and the private part is kept off-line, generally by the customer.

For generic Amino STBs, a default customer key is stored on the STB at time of manufacture. This key is included with all Amino SDKs and is used in-house for development. It is also used for all demonstration and evaluation STBs. In the event of a particular site moving to a full deployment the customer key in all of their STBs must be replaced with a customer specific one. For larger deployments, Amino manufacture with customer public keys in place.

The use of a single default key for new boxes ensures that the Amino master key is kept secure, as it is not needed for non-customer specific STBs (for example, demonstration and evaluation units). This also minimises the number of times that the Amino key is used, which always improves security. It also means that only customers at the point of deployment will need to be concerned with key security.

Once verified by the Master key, the Customer key is used to verify the STBRC key at startup time. It is also used to verify bootstraps and software downloads before executing them. The Customer keys is used for this verification and if it should fail then the Master key is tried, only when that also fails is the software rejected.

The customer key is used to verify that bootstrap and upgrade images have been signed by a trusted source.

The Amino engineering customer key (`KEY.private`) is normally provided with a release.

The password for this Amino engineering customer key is `markskey`.

4.1.1.3 STBRC key

The `STBRC` key is verified by the Customer key at startup and is only used to verify remote commands set to the STB by the `STBremoteconf` utility. Again, there is a heirarchy of keys so if a remote command does not verify with the `STBRC` key, then the Customer and Master keys are tried in that order.

This key is used for remote control of the STB. This is the key that is normally used when signing `STBremoteconf` messages and as such it may be scripted and will need to be stored on-line (on servers and given to field engineers).

As a result this is the key that is most likely to be compromised. Therefore it is not used to sign software which may run on the STB.

In the event of this key becoming compromised a customer can change it by generating a new key and then using `STBremoteconf` to upload it (in a message signed with their customer key).

This should be the only key that is vulnerable to loss since it is the only key which is kept online.

The STBRC key is used to verify the validity of remote management commands sent to the set-top box.

The Amino engineering configuration key (`STBrc-KEY.private`) is normally provided with a release.

The password for this Amino engineering configuration key is `stbrckey`.

4.1.2 Consequences of key compromises

This section discusses the consequences of any particular key becoming lost or compromised.

4.1.2.1 Amino Master key

The consequences of losing this key are serious – Amino would be unable to sign new customer keys. The key would immediately need to change and all newly manufactured boxes would need to be modified to use new keys.

Software could be written and signed with each customer's key that modifies the Amino Master Key stored in their STBs. This can only be done if customers have their keys available and the theft of the Amino Master Key is discovered before boxes are maliciously attacked.

This key clearly needs protecting at all costs and as such is never stored on-line. A duplicate of the key is stored offsite in a high-security environment.

4.1.2.2 Customer key

If this key is compromised then an attacker can run arbitrary software on any of the customer's STBs. They cannot run the software on other sites unless they also managed to steal the Customer key changing message that was signed by Amino.

Clearly all customers need to be aware of the serious consequences of the loss of this key and must be fully aware of their responsibility to keep it securely. Provided that they act quickly, loss of this key can normally be recovered without material loss but this cannot be guaranteed.

4.1.2.3 STBRC key

If this key is compromised then an attacker can issue commands to STBs. This could cause, amongst other things:

- Reboots.
- Displaying particular web pages.
- Transfer of files to the STB which will allow arbitrary code to be run (although this is a complex procedure).

4.2 Distributing software

This key hierarchy influences the method for distribution of software from Amino. The standard procedure is:

1. Amino create and test a new software build in-house.
2. Amino securely encrypts the software and transfers it to a customer.
3. The customer decrypts and modifies the software as required by their site. This might involve changing default addresses, for example.
4. The customer signs the image offline with their customer key.
5. The software image is multicasted to the customer's STBs.
6. The STBs check the validity of the signature on the image and accept it if they are satisfied.

Some smaller customers may choose not to maintain responsibility for their own keys. In that event Amino will generate and store the customer's key for them:

1. Amino create and test a new software build in-house.
2. Amino signs the image offline with the customer key.
3. Amino transfer the image to the customer (since this image is already signed it cannot be tampered with *en route*).
4. The software image is multicasted to the STBs.
5. The STBs check the validity of the signature on the image and accept it if they are satisfied.

4.3 Deployment Index (DI)

The Amino multicast upgrade system uses a facility called "Deployment Index". This was added at the request of Conditional Access vendors as a mechanism to prevent older software releases being loaded onto an Amino STB.

4.3.1 How it works

Stored in the STB's NOR Flash there is a DI number (starting at 0). When the STB downloads a new software image it compares the DI number in NOR Flash with the DI number in the software image. If the DI number in the software image is the same *or greater* than the DI number in NOR Flash, it accepts the upgrade. Otherwise, it rejects it.

So, when a customer receives a software update from Amino with a security fix they should increase the DI number by 1 to prevent anyone from loading older (lower DI numbered) images back onto the box.

The DI number is completely separate from software version numbers. Indeed, should a service provider wish to downgrade from software version 0.15.6 to 0.15.4 they can do this. All they have to do is generate the multicast download image of 0.15.4 again, but with a new (higher) DI number. The STBs will then accept this software and downgrade from 0.15.6 to 0.15.4

4.3.2 How to use it

The `signupgradeimage` script within the release takes a parameter that is the DI number to put into that image. This is a number in the range `0` to `65535`.

for example:

```
./signupgradeimage 1
```

For development work we recommend just using a Deployment Index of 1 all the time. In customer deployments, the service provider must increase the DI by 1 for a new release when advised by the service provider (that is, CA or middleware vendor) due to a security issue being fixed in the new release.

4.4 Passwords

Access to management functionality is assigned to password-controlled users.

The following table outlines the main password-controlled user levels:

User name	Description	Default password
`root`	Used for Telnet or serial console access on development builds.	`root2root`
`mngread`	Used for access to the set-top box management pages.	`leaves`
`mngwrite`	Used to write back changes from the set-top box management pages.	`snake`

Warning:

Amino strongly recommends that you change the default passwords when deploying our set-top boxes. See below for information on changing these default passwords.

4.4.1 Changing set-top box passwords

It is possible to customise the passwords used to access to a set-top box by editing the `shadow` file supplied with a software release, and then generating a new software upgrade image that includes the updated shadow file. In some builds it is also possible to customise passwords once software is installed on a set-top box, using `STBremoteconf`.

4.4.2 Shadow file

The `shadow` file defines the passwords for set-top box management users. Each line in the file specifies password information for a user, beginning with the user name.

4.4.3 To change passwords in the shadow file

Note: The following steps assume you have already set up a multicast upgrade system and the software release is in the location specified in the set-up instructions.

1. At a Linux command line, generate a new password hash with a command in the following format:

   ```
   perl -e 'print crypt("<password>", <salt>)'
   ```

(<password> is the new password you want to set, and <salt> is any two alphabetical characters.)

For example:

```
perl -e 'print crypt("admin", Ap)'
```

The command returns a hash for the password admin. In general, a hash function takes a string of any length as input and produces a fixed length string as output.

Result:

```
ApC4fVgseQ63w
```

2. Open the shadow file. If you used the recommended settings when you set up the multicast upgrade system, you will find the file in the following location:

   ```
   /usr/local/amino/releases/<release_name>/upgradeimage/imagecomponents
   ```

3. The shadow file has a line for each password that can be changed. Find the line for the password you want to change, and replace the current password hash with your new one. The password hash is the string between colons after the user name.

   For example, if you want to change the password for root, find the line that looks similar to the following:

   ```
   root:kUCfiRdXdzJSM:10596:0:99999:7:-1:-1:0
   ```

   Replace the password hash string, so that the line now looks similar to this:

   ```
   root:ApC4fVgseQ63w:10596:0:99999:7:-1:-1:0
   ```

   Repeat these steps for any other passwords you want to change, then save your changes.

Once you have changed the passwords as required, you will need to create a new software upgrade image that includes the edited shadow file, and save it to your set-top boxes.

Appendix A—LED flashing codes

The LED on the front of AmiNET set-top boxes indicates operational information such as errors during booting. A sequence of short and long flashes communicates the general type of information and the detail of the error.

Each LED flashing code is constructed from a sequence of flashes and pauses in the following order:

```
<major number of flashes><short pause><minor number of flashes><long
pause> <repeat>
```

A.1 Major error groups

The flashing codes indicate two main groups of errors – information and fatal errors.

Information errors: Information errors are indicated by shorter flashes. These errors generally require a reboot of the set-top box, but there are exceptions - for example if the set-top box can't execute a command sent by STBremoteconf, it continues normal operation rather than rebooting.

Fatal errors: Fatal errors are indicated by longer flashes. When the set-top box detects these errors, it flashes the LED error in an endless loop. Rebooting the set-top box in these cases may cause it to stop responding permanently, particularly after a NOR flash write failure. For this reason, the set-top box should not be rebooted until all diagnostics have been completed.

The following table shows the number of flashes that indicates the major groups of information errors.

Major no. of flashes	Error type
1	Not full functionality but the box will start-up fully.
2	Code upgrade failure.
3	Security/cryptographic issue (boot loader).
4	Security/cryptographic issue (OS).
5	Miscellaneous errors.
6	Reserved for future use.
7	Reserved for future use.
8	**Cryptographic failure (fatal).**
9	**Hardware failure (fatal).**

List of LED flashing codes

Major	Minor	Error	Set-top box behaviour	Reboot
1	**Not full functionality but the box will start-up fully**			
1	1	No network connection.	Repeats three times. STB continues booting.	No
1	2	DHCP look-up failure.	Repeats three times. STB continues booting.	No
1	3	I2C communication failure.	Repeats three times. STB continues booting.	No
2	**Code upgrade failure**			
2	1	TFTP download failure.	`poweron` file is not saved. Reboots.	Yes
2	2	BOOTP server not found.	`poweron` file is not saved. STB reboots.	Yes
2	3	Multicast group not found.	Retries three times. If still unsuccessful, STB reboots.	Yes
2	4	Multicast download timeout.	Retries three times. If still unsuccessful, STB reboots.	Yes
2	5	`product` file missing from running software.	Upgrade system is broken, uses reflash and then reboots.	Yes
2	6	`bin/products` file missing from downloaded software upgrade image.	STB reboots.	Yes
2	7	`bin/products` file does not match set-top box product (i.e. the software upgrade image is not suitable for this platform).	STB reboots.	Yes
2	8	Failed to load new NOR flash driver.	STB reboots.	Yes
2	9	An `EXT2SEC` upgrade did not find an EXT ROM.	No ROM upgrade takes place. STB reboots.	Yes
3	**Security/cryptographic issue (boot loader)**			
3	1	Verification of download image failed.	Deletes the image if saved in the flash. Retries three times. If still unsuccessful, STB reboots.	Yes
3	2	Signature check of `listfile.sig` failed.	Deletes the image if saved in the flash. Retries three times. If still unsuccessful, STB reboots.	Yes
3	3	`listfile.sig` is missing.	Erases the flash (which must be assumed to be unsafe) and reboots.	Yes

Major	Minor	Error	Set-top box behaviour	Reboot
3	4	Missing file in the NAND flash (according to `listfile.sig`).	Erases the flash (which must be assumed to be unsafe) and reboots.	Yes
3	5	File check against `listfile.sig` failed.	Erases the flash (which must be assumed to be unsafe) and reboots.	Yes
3	6	File found in the NAND flash is not in `listfile.sig`.	Erases the flash (which must be assumed to be unsafe) and reboots.	Yes
3	7	Unable to decode the customer public key on the STB.	STB reboots.	Yes
4	**Security/cryptographic issue (OS)**			
4	1	Verification of the download image failed.	Deletes the image if saved in the flash. Retries three times. If still unsuccessful, reboots.	Yes
4	2	Signature check of `listfile.sig` failed.	Deletes the image if saved in the flash. Retries 3 times. If still unsuccessful, reboots.	Yes
4	3	`listfile.sig` is missing or corrupted.	Erases the flash (assumed to be unsafe) and reboots.	Yes
4	4	Software upgrade image DI (deployment index) is less than set-top box DI.	Deletes the image if saved in the flash. Retries three times. If still unsuccessful, the STB reboots.	Yes
4	5	Set DI failed.	Deletes the code contents of the flash, then reboots.	Yes
4	6	`STBremoteconf` key not found.	Ignores command.	No
4	7	`STBremoteconf` key failed to verify.	Ignores command.	No
4	8	MD5 sum failure.	Rejects software upgrade image.	No
4	9	Number of entries in `listfile.sig` doesn't match files downloaded.	Rejects software upgrade image.	No
4	10	No DI found.	STB reboots.	Yes
5	**Miscellaneous errors**			
5	1	NAND flash corruption.	Reformat the NAND flash and reboot.	Yes
5	2	PID mismatch when upgrading the ROM.	Reboots.	Yes
5	3	Failed to remove files on upgrade.	Reboots and retries.	Yes

Major	Minor	Error	Set-top box behaviour	Reboot
5	4	Failed to set execute/start address in the NAND flash.	Clears the NAND flash.	Yes
5	5	Kernel crash / no root FS.	Reboots (cannot clear the NAND flash).	Yes
5	6	Invalid type of GIF file saved on NAND flash.	Deletes the file.	Yes
5	7	Failed to write files to NAND flash.	Clears the NAND flash	Yes
5	8	Failed to gunzip the boot ROM image.	Rejects image and reboots.	Yes
6	**Reserved for future use**			
7	**Reserved for future use**			
8	**Cryptographic failure (fatal)**			
8	1	No master key found.	Flashes in endless loop	No
8	2	No customer key found.	Flashes in endless loop	No
8	3	Signature check of customer key failed.	Flashes in endless loop	No
8	6	Verification of IntactOS code failed.	Flashes in endless loop	No
8	7	Key update during bootstrap failed.	Flashes and then continues the bootstrap.	
9	**Hardware failure (fatal)**			
9	1	Page write to ROM failed.	Keeps flashing code in endless loop.	No
9	2	NAND flash hardware / format failure.	Keeps flashing code in endless loop.	No
9	3	SDRAM access failure.	Keeps flashing code in endless loop.	No
9	4	Communication to Ethernet chip failed.	Keeps flashing code in endless loop.	No
9	5	STB has overheated.	Keeps flashing code in endless loop.	

A.2 Ethernet monitoring

The physical network connection at the back of the STB contains two LEDs. The Ethernet LEDs are set up as follows:

- Green - on - 100Mbit or 10Mbit link detected (so off = no link)
- Amber - on - Tx or Rx packet.. Data activity.

The speed at which the Amber LED flashes depends on the configuration of the ethernet chip and the network activity.

It should be noted however that for the AmiNET125 the position is reversed:

- Amber - on - 100Mbit or 10Mbit link detected (so off = no link)
- Green - on - Tx or Rx packet.. Data activity.

Appendix B—DHCP Configuration

This Appendix contains a full copy of the unmodified DHCP file. This example configuration assumes that you are creating a DHCP file for an AmiNET 130 and that the IP address of the multicast server is `239.255.225.100`.

Should you wish to add a different model of set-top box, simply add the configuration details for the set-top box type that you wish to add.

Note: Some of the line lengths in this file have been edited for the sake of appearance.

Note: Bold text indicates where the file contents have been edited.

Note: Grey text indicates where the contents of the DGHCP file are not strictly necessary for the file to function correctly. If you are editing the file for one particular type of STB, it may be worth removing the unnecessary grey text to make the file less unwieldy.

```
#########################################################################
# Amino Communications Sample dhcpd.conf file                           #
#                                                                       #
# Note, this is not a complete dhcpd.conf file. It just covers the various#
# settings required for Amino AmiNet STB multicast bootstraping         #
# it may require some changes for deployment on your network.           #
#                                                                       #
# 29.01.04 - Created Paul Rae                                           #
# 20.08.04 - Aminet500 and Aminet110H added by Richard Warren           #
# 03.10.05 - AMINO.local-config added by Steve McIntyre                 #
# 29.03.06 - Aminet124 and mboot bootrom versions added by Rob Thornburrow#
#                                                                       #
#########################################################################

#########################################################################
# Misc dhcp options                                                     #
#########################################################################
        allow bootp;
        ddns-update-style ad-hoc;
        filename="AMINET.txt";


#########################################################################
# Extra Options for AMINO option space (used for multicast)             #
#########################################################################
        option space AMINO;
        option AMINO.address            code  1 = ip-address;
        option AMINO.port               code  2 = integer 16;
        option AMINO.product            code  3 = text;
        option AMINO.option             code  4 = text;
```

```
        option AMINO.version             code  5 = text;
        option AMINO.middleware          code  6 = ip-address;
        option AMINO.mw_port             code  7 = integer 16;
        option AMINO.homepage            code  8 = text;
        option AMINO.dindex              code  9 = integer 32;
        option AMINO.dindex_min          code 10 = integer 32;
        option AMINO.dindex_page         code 11 = text;
        option AMINO.STBrc-mcast-address code 12 = ip-address;
        option AMINO.STBrc-mcast-port    code 13 = integer 16;
        option AMINO.STBrc-unicast-port  code 14 = integer 16;
        option AMINO.local-config        code 15 = text;
        option AMINO.timezone            code 16 = text;
        option AMINO.middleware2         code 17 = ip-address;
        option AMINO.mw_args             code 18 = text;
        option AMINO.test_host           code 19 = ip-address;
        option AMINO.test_dir            code 20 = text;
        option AMINO.recovery_mode       code 21 = integer 8;
        option AMINO.mirimon_args        code 22 = text;

    ##########################################################################
    # AmiNET103 Configuration Section                                        #
    ##########################################################################
    #                                                                        #
    # class "AmiNET103 mboot" - boot state when requesting bootstrap image   #
    # class "AmiNET103 upgrd" - boot state when requesting main upgrade image#
    # class "AmiNET103 fisys" - boot state when in normal state              #
    #                                                                        #
    # The only items that may need changing are as follows:                  #
    #                                                                        #
    # option AMINO.address 225.50.50.50; - the multicast address you are     #
    # streaming on                                                           #
    # option AMINO.port 11111; - the port you are streaming on               #
    #                                                                        #
    # If you change any of these options you must also make sure you make the #
    # appropriate changes to /etc/mcastbootd.conf                            #
    #                                                                        #
    # In the mboot class, the bootrom version (e.g. 1.32) must be given in the #
    # vendor-class-identifier.  If multiple bootrom versions need to be      #
    # supported multiple match cases may be used.                            #
    #                                                                        #
    ##########################################################################



    ##########################################################################
    # Class "AmiNET103 mboot"                                                #
    # AmiNET103 - response to bootrom request for a bootstrap image          #
    ##########################################################################
    class "AmiNET103 mboot"
    {
      match if (option vendor-class-identifier="aminoAMINET103mboot<bootrom
version>") or
               ((substring( option vendor-encapsulated-options, 2,
9)="AMINET103")
                and (substring(option vendor-encapsulated-options, 13,
5)="mboot"));

      vendor-option-space AMINO;
```

```
      option AMINO.address 225.50.50.50;
      option AMINO.port 11111;
    }
    ###########################################################################

    ###########################################################################
    # Class "AmiNET103 upgrd"                                                 #
    # AmiNET103 - response to bootstrap request for a main upgrade image    #
    ###########################################################################
    class "AmiNET103 upgrd"
    {
      match if (option vendor-class-identifier="Aminoaminet103upgrd") or
               ((substring( option vendor-encapsulated-op-
tions,2,9)="aminet103")
               and (substring( option vendor-encapsulated-options,13,5)="up-
grd"));

      vendor-option-space AMINO;
      option AMINO.address 225.50.50.51;
      option AMINO.port 11111;
    }
    ###########################################################################

    ###########################################################################
    # Class "AmiNET103 fisys"                                                 #
    # AmiNET103 - response when booting in normal boot state                  #
    ###########################################################################
    class "AmiNET103 fisys"
    {
      match if (option vendor-class-identifier="Aminoaminet103fisys") or
               ((substring( option vendor-encapsulated-options, 2,
9)="aminet103")
                and (substring(option vendor-encapsulated-options, 13,
5)="fisys"));
      vendor-option-space AMINO;
      option AMINO.middleware <mcast address>;
      option AMINO.mw_port <port>;
    }


    ###########################################################################
    # AmiNET110 Configuration Section                                         #
    ###########################################################################
    #                                                                         #
    # class "AmiNET110 mboot" - boot state when requesting bootstrap image    #
    # class "AmiNET110 upgrd" - boot state when requesting main upgrade image #
    # class "AmiNET110 fisys" - boot state when in normal state               #
    #                                                                         #
    # The only items that may need changing are as follows:                   #
    #                                                                         #
    # option AMINO.address 225.50.50.50; - the multicast address you are      #
    # streaming on                                                            #
    # option AMINO.port 11111; - the port you are streaming on                #
    #                                                                         #
    # If you change any of these options you must also make sure you make the  #
    # appropriate changes to /etc/mcastbootd.conf                             #
    #                                                                         #
    # In the mboot class, the bootrom version (e.g. 1.32) must be given in the #
```

```
    # vendor-class-identifier.  If multiple bootrom versions need to be     #
    # supported multiple match cases may be used.                           #
    #########################################################################


    #########################################################################
    # Class "AmiNET110 mboot"                                               #
    # AmiNET110 - response to bootrom request for a bootstrap image         #
    #########################################################################
    class "AmiNET110 mboot"
    {
      match if (option vendor-class-identifier="aminoAMINET11xmboot<bootrom
version>") or
                ((substring(option vendor-encapsulated-options, 2,
9)="AMINET11x")
                 and (substring(option vendor-encapsulated-options, 13,
5)="mboot"));

      vendor-option-space AMINO;
      option AMINO.address 225.50.50.52;
      option AMINO.port 11111;
    }

    #########################################################################
    # Class "AmiNET110 upgrd"                                               #
    # AmiNET110 - response to bootstrap request for a main upgrade image    #
    #########################################################################
    class "AmiNET110 upgrd"
    {
      match if (option vendor-class-identifier="Aminoaminet110upgrd") or
                ((substring( option vendor-encapsulated-op-
tions,2,9)="aminet110")
                and (substring( option vendor-encapsulated-options,13,5)="up-
grd"));

      vendor-option-space AMINO;
      option AMINO.address 225.50.50.53;
      option AMINO.port 11111;
    }

    #########################################################################
    # Class "AmiNET110 fisys"                                               #
    # AmiNET110 - response when booting in normal boot state                #
    #########################################################################
    class "AmiNET110 fisys"
    {
      match if (option vendor-class-identifier="Aminoaminet110fisys") or
                ((substring( option vendor-encapsulated-options, 2,
9)="aminet110")
                 and (substring(option vendor-encapsulated-options, 13,
5)="fisys"));
      vendor-option-space AMINO;
      option AMINO.middleware <mcast address>;
      option AMINO.mw_port <port>;
    }
```

```
    ###########################################################################
    # AmiNET110H Configuration Section                                        #
    ###########################################################################
    #                                                                         #
    # class "AmiNET110H mboot" - boot state when requesting bootstrap image   #
    # class "AmiNET110H upgrd" - boot state when requesting main upgrade image#
    # class "AmiNET110H fisys" - boot state when in normal state              #
    #                                                                         #
    # The only items that may need changing are as follows:                   #
    #                                                                         #
    # option AMINO.address 225.50.50.50; - the multicast address you are      #
    # streaming on                                                            #
    # option AMINO.port 11111; - the port you are streaming on                #
    #                                                                         #
    # If you change any of these options you must also make sure you make the  #
    # appropriate changes to /etc/mcastbootd.conf                             #
    #                                                                         #
    # In the mboot class, the bootrom version (e.g. 1.32) must be given in the #
    # vendor-class-identifier.  If multiple bootrom versions need to be       #
    # supported multiple match cases may be used.                             #
    #                                                                         #
    ###########################################################################



    ###########################################################################
    # Class "AmiNET110H mboot"                                                #
    # AmiNET110H - response to bootrom request for a bootstrap image          #
    ###########################################################################
    class "AmiNET110H mboot"
    {
      match if (option vendor-class-identifier="aminoAMINET110Hmboot<bootrom
version>") or
                ((substring(option vendor-encapsulated-options, 2,
10)="AMINET110H")
                 and (substring(option vendor-encapsulated-options, 14,
5)="mboot"));

      vendor-option-space AMINO;
      option AMINO.address 225.50.50.52;
      option AMINO.port 11111;
    }

    ###########################################################################
    # Class "AmiNET110H upgrd"                                                #
    # AmiNET110H - response to bootstrap request for a main upgrade image     #
    ###########################################################################
    class "AmiNET110H upgrd"
    {
      match if (option vendor-class-identifier="Aminoaminet110hupgrd") or
                ((substring( option vendor-encapsulated-op-
tions,2,10)="aminet110h")
                 and (substring( option vendor-encapsulated-options,14,5)="up-
grd"));

      vendor-option-space AMINO;
```

```
      option AMINO.address 225.50.50.53;
      option AMINO.port 11111;
    }

    ###########################################################################
    # Class "AmiNET110h fisys"                                                #
    # AmiNET110 - response when booting in normal boot state                  #
    ###########################################################################
    class "AmiNET110H fisys"
    {
      match if (option vendor-class-identifier="Aminoaminet110hfisys") or
              ((substring( option vendor-encapsulated-options, 2,
10)="aminet110h")
               and (substring(option vendor-encapsulated-options, 14,
5)="fisys"));

      vendor-option-space AMINO;
      option AMINO.middleware <mcast address>;
      option AMINO.mw_port <port>;
    }




    ###########################################################################
    # AmiNET500 Configuration Section                                         #
    ###########################################################################
    #                                                                         #
    # class "AmiNET500 mboot" - boot state when requesting bootstrap image    #
    # class "AmiNET500 upgrd" - boot state when requesting main upgrade image #
    # class "AmiNET500 fisys" - boot state when in normal state               #
    #                                                                         #
    # The only items that may need changing are as follows:                   #
    #                                                                         #
    # option AMINO.address 225.50.50.50; - the multicast address you are      #
    # streaming on                                                            #
    # option AMINO.port 11111; - the port you are streaming on                #
    #                                                                         #
    # If you change any of these options you must also make sure you make the  #
    # appropriate changes to /etc/mcastbootd.conf                             #
    #                                                                         #
    # In the mboot class, the bootrom version (e.g. 1.32) must be given in the #
    # vendor-class-identifier.  If multiple bootrom versions need to be       #
    # supported multiple match cases may be used.                             #
    #                                                                         #
    ###########################################################################



    ###########################################################################
    # Class "AmiNET500 mboot"                                                 #
    # AmiNET500 - response to bootrom request for a bootstrap image           #
    ###########################################################################
    class "AmiNET500 mboot"
    {
      match if (option vendor-class-identifier="aminoAMINET5xxmboot<bootrom
version>") or
```

```
              ((substring(option vendor-encapsulated-options, 2,
9)="AMINET5xx")
               and (substring(option vendor-encapsulated-options, 13,
5)="mboot"));

      vendor-option-space AMINO;
      option AMINO.address 225.50.50.52;
      option AMINO.port 11111;
    }

    ###########################################################################
    # Class "AmiNET500 upgrd"                                                 #
    # AmiNET500 - response to bootstrap request for a main upgrade image      #
    ###########################################################################
    class "AmiNET500 upgrd"
    {
      match if (option vendor-class-identifier="Aminoaminet500upgrd") or
              ((substring( option vendor-encapsulated-op-
tions,2,9)="aminet500")
              and (substring( option vendor-encapsulated-options,13,5)="up-
grd"));

      vendor-option-space AMINO;
      option AMINO.address 225.50.50.53;
      option AMINO.port 11111;
    }

    ###########################################################################
    # Class "AmiNET500 fisys"                                                 #
    # AmiNET500 - response when booting in normal boot state                  #
    ###########################################################################
    class "AmiNET500 fisys"
    {
      match if (option vendor-class-identifier="Aminoaminet500fisys") or
              ((substring( option vendor-encapsulated-options, 2,
9)="aminet500")
               and (substring(option vendor-encapsulated-options, 13,
5)="fisys"));
      vendor-option-space AMINO;
      option AMINO.middleware <mcast address>;
      option AMINO.mw_port <port>;
    }



    ###########################################################################
    # AmiNET124 Configuration Section                                         #
    ###########################################################################
    #                                                                         #
    # class "AmiNET124 mboot" - boot state when requesting bootstrap image    #
    # class "AmiNET124 upgrd" - boot state when requesting main upgrade image #
    # class "AmiNET124 fisys" - boot state when in normal state               #
    #                                                                         #
    # The only items that may need changing are as follows:                   #
    #                                                                         #
    # option AMINO.address 225.50.50.50; - the multicast address you are      #
    # streaming on                                                            #
    # option AMINO.port 11111; - the port you are streaming on                #
```

```
   #                                                                      #
   # If you change any of these options you must also make sure you make the  #
   # appropriate changes to /etc/mcastbootd.conf                          #
   #                                                                      #
   # In the mboot class, the bootrom version (e.g. 1.32) must be given in the #
   # vendor-class-identifier.  If multiple bootrom versions need to be     #
   # supported multiple match cases may be used.                          #
   #                                                                      #
   ########################################################################



   ########################################################################
   # Class "AmiNET124 mboot"                                              #
   # AmiNET124 - response to bootrom request for a bootstrap image        #
   ########################################################################
   class "AmiNET124 mboot"
   {
     match if (option vendor-class-identifier="aminoAMINET124mboot<bootrom
version>") or
              ((substring(option vendor-encapsulated-options, 2,
9)="AMINET124")
               and (substring(option vendor-encapsulated-options, 13,
5)="mboot"));

     vendor-option-space AMINO;
     option AMINO.address 225.50.50.52;
     option AMINO.port 11111;
   }

   ########################################################################
   # Class "AmiNET124 upgrd"                                              #
   # AmiNET124 - response to bootstrap request for a main upgrade im-
age          #
   ########################################################################
   class "AmiNET124 upgrd"
   {
     match if (option vendor-class-identifier="Aminoaminet124upgrd") or
              ((substring( option vendor-encapsulated-op-
tions,2,9)="aminet124")
              and (substring( option vendor-encapsulated-options,13,5)="up-
grd"));

     vendor-option-space AMINO;
     option AMINO.address 225.50.50.53;
     option AMINO.port 11111;
   }

   ########################################################################
   # Class "AmiNET124 fisys"                                              #
   # AmiNET124 - response when booting in normal boot state               #
   ########################################################################
   class "AmiNET124 fisys"
   {
     match if (option vendor-class-identifier="Aminoaminet124fisys") or
              ((substring( option vendor-encapsulated-options, 2,
9)="aminet124")
```

```
                and (substring(option vendor-encapsulated-options, 13,
5)="fisys"));
      vendor-option-space AMINO;
      option AMINO.middleware <mcast address>;
      option AMINO.mw_port <port>;
    }


    ###########################################################################
    # AmiNET125 Configuration Section                                         #
    ###########################################################################
    #                                                                         #
    # class "AmiNET125 mboot" - boot state when requesting bootstrap image    #
    # class "AmiNET125 upgrd" - boot state when requesting main upgrade image #
    # class "AmiNET125 fisys" - boot state when in normal state               #
    #                                                                         #
    # The only items that may need changing are as follows:                   #
    #                                                                         #
    # option AMINO.address 225.50.50.50; - the multicast address you are      #
    # streaming on                                                            #
    # option AMINO.port 11111; - the port you are streaming on                #
    #                                                                         #
    # If you change any of these options you must also make sure you make the  #
    # appropriate changes to /etc/mcastbootd.conf                             #
    #                                                                         #
    # In the mboot class, the bootrom version (e.g. 1.32) must be given in the #
    # vendor-class-identifier.  If multiple bootrom versions need to be       #
    # supported multiple match cases may be used.                             #
    #                                                                         #
    ###########################################################################


    ###########################################################################
    # Class "AmiNET125 mboot"                                                 #
    # AmiNET125 - response to bootrom request for a bootstrap image           #
    ###########################################################################
    class "AmiNET125 mboot"
    {
      match if (option vendor-class-identifier="aminoAMINET125mboot<bootrom
version>") or
               ((substring(option vendor-encapsulated-options, 2,
9)="AMINET125")
                and (substring(option vendor-encapsulated-options, 13,
5)="mboot"));

      vendor-option-space AMINO;
      option AMINO.address <mcast address>;
      option AMINO.port <port>;
    }

    ###########################################################################
    # Class "AmiNET125 upgrd"                                                 #
    # AmiNET125 - response to bootstrap request for a main upgrade image      #
    ###########################################################################
    class "AmiNET125 upgrd"
    {
```

```
     match if (option vendor-class-identifier="Aminoaminet125upgrd") or
              ((substring( option vendor-encapsulated-op-
tions,2,9)="aminet125")
              and (substring( option vendor-encapsulated-options,13,5)="up-
grd"));

     vendor-option-space AMINO;
     option AMINO.address <mcast address>;
     option AMINO.port <port>;
   }

   ###########################################################################
   # Class "AmiNET125 fisys"                                                 #
   # AmiNET125 - response when booting in normal boot state                  #
   ###########################################################################
   class "AmiNET125 fisys"
   {
     match if (option vendor-class-identifier="Aminoaminet125fisys") or
              ((substring( option vendor-encapsulated-options, 2,
9)="aminet125")
               and (substring(option vendor-encapsulated-options, 13,
5)="fisys"));
     vendor-option-space AMINO;
     option AMINO.middleware <mcast address>;
     option AMINO.mw_port <port>;
   }




   ###########################################################################

   # Class "AmiNET120 mboot"                                                 #
   # AmiNET120 - response when bootstrapping                                 #
   ###########################################################################
   class "AMINET120 mboot"
   {
     match if (option vendor-class-identifier="aminoAMINET120mboot") or
              ((substring(option vendor-encapsulated-options, 2,
9)="AMINET120")
               and (substring(option vendor-encapsulated-options, 13,
5)="mboot"));

   #  option log-servers10.172.2.20;
     vendor-option-space AMINO;
     option AMINO.address 239.255.224.100;
     option AMINO.port 1001;
   }

   ###########################################################################
   # Class "AmiNET120 upgrd"                                                 #
   # AmiNET120 - response when upgrading                                     #
   ###########################################################################
   class "aminet120 upgrd"
   {
     match if (option vendor-class-identifier="Aminoaminet120upgrd") or
              ((substring( option vendor-encapsulated-op-
tions,2,9)="aminet120")
```

```
              and (substring( option vendor-encapsulated-options,13,5)="up-
grd"));

      vendor-option-space AMINO;
      option AMINO.address 239.255.224.101;
      option AMINO.port 1001;
    }



    ###########################################################################
    # AmiNET130 Configuration Section                                         #
    ###########################################################################
    #                                                                         #
    # class "AmiNET130 mboot" - boot state when requesting bootstrap image    #
    # class "AmiNET130 upgrd" - boot state when requesting main upgrade image #
    # class "AmiNET130 fisys" - boot state when in normal state               #
    #                                                                         #
    # The only items that may need changing are as follows:                   #
    #                                                                         #
    # option AMINO.address 225.50.51.50; - the multicast address you are      #
    # streaming on                                                            #
    # option AMINO.port 11111; - the port you are streaming on                #
    #                                                                         #
    # If you change any of these options you must also make sure you make the  #
    # appropriate changes to /etc/mcastbootd.conf                             #
    #                                                                         #
    # In the mboot class, the bootrom version (e.g. 1.32) must be given in the #
    # vendor-class-identifier.  If multiple bootrom versions need to be       #
    # supported multiple match cases may be used.                             #
    #                                                                         #
    ###########################################################################



    ###########################################################################
    # Class "AmiNET130 mboot"                                                 #
    # AmiNET130 - response to bootrom request for a bootstrap image           #
    ###########################################################################
    class "AmiNET130 mboot"
    {
      match if (option vendor-class-identifier="aminoAMINET130mboot<bootrom
version>") or
               ((substring(option vendor-encapsulated-options, 2,
9)="AMINET130")
                and (substring(option vendor-encapsulated-options, 13,
5)="mboot"));

      vendor-option-space AMINO;
      option AMINO.address 239.255.225.100;
      option AMINO.port 11111;
    }

    ###########################################################################
    # Class "AmiNET130 upgrd"                                                 #
    # AmiNET130 - response to bootstrap request for a main upgrade image      #
    ###########################################################################
```

```
   class "AmiNET130 upgrd"
   {
     match if (option vendor-class-identifier="Aminoaminet130upgrd") or
              ((substring( option vendor-encapsulated-op-
tions,2,9)="aminet130")
               and (substring( option vendor-encapsulated-options,13,5)="up-
grd"));

     vendor-option-space AMINO;
     option AMINO.address 239.255.225.101;
     option AMINO.port 11111;
   }

   ###########################################################################
   # Class "AmiNET130 fisys"                                                 #
   # AmiNET130 - response when booting in normal boot state                  #
   ###########################################################################
   class "AmiNET130 fisys"
   {
     match if (option vendor-class-identifier="Aminoaminet130fisys") or
              ((substring( option vendor-encapsulated-options, 2,
9)="aminet130")
              and (substring(option vendor-encapsulated-options, 13, 5)="fi-
sys"));
     vendor-option-space AMINO;
     #option AMINO.middleware <mcast address>;
     #option AMINO.mw_port <port>;
   }


   ###########################################################################
   # AmiNET130M Configuration Section                                        #
   ###########################################################################
   #                                                                         #
   # class "AmiNET130M mboot" - boot state when requesting bootstrap image   #
   # class "AmiNET130M upgrd" - boot state when requesting main upgrade image#
   # class "AmiNET130M fisys" - boot state when in normal state              #
   #                                                                         #
   # The only items that may need changing are as follows:                   #
   #                                                                         #
   # option AMINO.address 225.50.52.50; - the multicast address you are      #
   # streaming on                                                            #
   # option AMINO.port 11111; - the port you are streaming on                #
   #                                                                         #
   # If you change any of these options you must also make sure you make the  #
   # appropriate changes to /etc/mcastbootd.conf                             #
   #                                                                         #
   # In the mboot class, the bootrom version (e.g. 1.32) must be given in the #
   # vendor-class-identifier.  If multiple bootrom versions need to be       #
   # supported multiple match cases may be used.                             #
   #                                                                         #
   ###########################################################################



   ###########################################################################
#######
   # Class "AmiNET130M mboot"                                                #
```



```
   # AmiNET130M - response to bootrom request for a bootstrap image
#
   ##########################################################################
#######
   class "AmiNET130M mboot"
   {
     match if (option vendor-class-identifier="aminoAMINET130Mmboot<bootrom
version>") or
              ((substring(option vendor-encapsulated-options, 2,
10)="AMINET130M")
               and (substring(option vendor-encapsulated-options, 14,
5)="mboot"));

     vendor-option-space AMINO;
     option AMINO.address 225.50.52.52;
     option AMINO.port 11111;
   }

   ##########################################################################
#######
   # Class "AmiNET130M upgrd"                                               #
   # AmiNET130M - response to bootstrap request for a main upgrade im-
age         #
   ##########################################################################
#######
   class "AmiNET130M upgrd"
   {
     match if (option vendor-class-identifier="Aminoaminet130mupgrd") or
               ((substring( option vendor-encapsulated-op-
tions,2,10)="aminet130m")
                and (substring( option vendor-encapsulated-options,14,5)="up-
grd"));

     vendor-option-space AMINO;
     option AMINO.address 225.50.52.53;
     option AMINO.port 11111;
   }

   ##########################################################################
#######
   # Class "AmiNET130m fisys"                                               #
   # AmiNET130 - response when booting in normal boot state                 #
   ##########################################################################
#######
   class "AmiNET130M fisys"
   {
     match if (option vendor-class-identifier="Aminoaminet130mfisys") or
              ((substring( option vendor-encapsulated-options, 2,
10)="aminet130m")
               and (substring(option vendor-encapsulated-options, 14,
5)="fisys"));

     vendor-option-space AMINO;
     option AMINO.middleware <mcast address>;
     option AMINO.mw_port <port>;
   }
```

```
   ###########################################################################
   # AmiNET130H Configuration Section                                        #
   ###########################################################################
   #                                                                         #
   # class "AmiNET130H mboot" - boot state when requesting bootstrap image   #
   # class "AmiNET130H upgrd" - boot state when requesting main upgrade image#
   # class "AmiNET130H fisys" - boot state when in normal state              #
   #                                                                         #
   # The only items that may need changing are as follows:                   #
   #                                                                         #
   # option AMINO.address 225.50.52.50; - the multicast address you are      #
   # streaming on                                                            #
   # option AMINO.port 11111; - the port you are streaming on                #
   #                                                                         #
   # If you change any of these options you must also make sure you make the  #
   # appropriate changes to /etc/mcastbootd.conf                             #
   #                                                                         #
   # In the mboot class, the bootrom version (e.g. 1.32) must be given in the #
   # vendor-class-identifier.  If multiple bootrom versions need to be       #
   # supported multiple match cases may be used.                             #
   #                                                                         #
   ###########################################################################



   ###########################################################################
   # Class "AmiNET130H mboot"                                                #
   # AmiNET130H - response to bootrom request for a bootstrap image          #
   ###########################################################################
   class "AmiNET130H mboot"
   {
     match if (option vendor-class-identifier="aminoAMINET130Hmboot<bootrom
version>") or
              ((substring(option vendor-encapsulated-options, 2,
10)="AMINET130H")
               and (substring(option vendor-encapsulated-options, 14,
5)="mboot"));

     vendor-option-space AMINO;
     option AMINO.address 225.50.52.52;
     option AMINO.port 11111;
   }

   ###########################################################################
   # Class "AmiNET130H upgrd"                                                #
   # AmiNET130H - response to bootstrap request for a main upgrade image   #
   ###########################################################################
   class "AmiNET130H upgrd"
   {
     match if (option vendor-class-identifier="Aminoaminet130hupgrd") or
              ((substring( option vendor-encapsulated-op-
tions,2,10)="aminet130h")
               and (substring( option vendor-encapsulated-options,14,5)="up-
grd"));

     vendor-option-space AMINO;
     option AMINO.address 225.50.52.53;
     option AMINO.port 11111;
```

```
    }

    ###########################################################################
    # Class "AmiNET130h fisys"                                                #
    # AmiNET130 - response when booting in normal boot state                  #
    ###########################################################################
    class "AmiNET130H fisys"
    {
      match if (option vendor-class-identifier="Aminoaminet130hfisys") or
               ((substring( option vendor-encapsulated-options, 2,
10)="aminet130h")
                and (substring(option vendor-encapsulated-options, 14,
5)="fisys"));

      vendor-option-space AMINO;
      option AMINO.middleware <mcast address>;
      option AMINO.mw_port <port>;
    }



    ###########################################################################
    # AmiNET131 Configuration Section                                         #
    ###########################################################################
    #                                                                         #
    # class "AmiNET131 mboot" - boot state when requesting bootstrap image    #
    # class "AmiNET131 upgrd" - boot state when requesting main upgrade image #
    # class "AmiNET131 fisys" - boot state when in normal state               #
    #                                                                         #
    # The only items that may need changing are as follows:                   #
    #                                                                         #
    # option AMINO.address 225.50.53.50; - the multicast address you are      #
    # streaming on                                                            #
    # option AMINO.port 11111; - the port you are streaming on                #
    #                                                                         #
    # If you change any of these options you must also make sure you make the  #
    # appropriate changes to /etc/mcastbootd.conf                             #
    #                                                                         #
    # In the mboot class, the bootrom version (e.g. 1.32) must be given in the #
    # vendor-class-identifier.  If multiple bootrom versions need to be       #
    # supported multiple match cases may be used.                             #
    #                                                                         #
    ###########################################################################



    ###########################################################################
    # Class "AmiNET131 mboot"                                                 #
    # AmiNET131 - response to bootrom request for a bootstrap image           #
    ###########################################################################
    class "AmiNET131 mboot"
    {
      match if (option vendor-class-identifier="aminoAMINET131mboot<bootrom
version>") or
               ((substring(option vendor-encapsulated-options, 2,
9)="AMINET131")
```

```
                and (substring(option vendor-encapsulated-options, 13,
5)="mboot"));

      vendor-option-space AMINO;
      option AMINO.address 225.50.53.52;
      option AMINO.port 11111;
    }

    ##########################################################################
    # Class "AmiNET131 upgrd"                                                #
    # AmiNET131 - response to bootstrap request for a main upgrade image     #
    ##########################################################################
    class "AmiNET131 upgrd"
    {
      match if (option vendor-class-identifier="Aminoaminet131upgrd") or
                ((substring( option vendor-encapsulated-op-
tions,2,9)="aminet131")
                and (substring( option vendor-encapsulated-options,13,5)="up-
grd"));

      vendor-option-space AMINO;
      option AMINO.address 225.50.53.53;
      option AMINO.port 11111;
    }

    ##########################################################################
    # Class "AmiNET131 fisys"                                                #
    # AmiNET131 - response when booting in normal boot state                 #
    ##########################################################################
    class "AmiNET131 fisys"
    {
      match if (option vendor-class-identifier="Aminoaminet131fisys") or
                ((substring( option vendor-encapsulated-options, 2,
9)="aminet131")
                 and (substring(option vendor-encapsulated-options, 13,
5)="fisys"));
      vendor-option-space AMINO;
      option AMINO.middleware <mcast address>;
      option AMINO.mw_port <port>;
    }




    ##########################################################################
    # AmiNET530 Configuration Section                                        #
    ##########################################################################
    #                                                                        #
    # class "AmiNET530 mboot" - boot state when requesting bootstrap image   #
    # class "AmiNET530 upgrd" - boot state when requesting main upgrade image #
    # class "AmiNET530 fisys" - boot state when in normal state              #
    #                                                                        #
    # The only items that may need changing are as follows:                  #
    #                                                                        #
    # option AMINO.address 225.50.54.50; - the multicast address you are     #
    # streaming on                                                           #
    # option AMINO.port 11111; - the port you are streaming on               #
```

```
   #                                                                      #
   # If you change any of these options you must also make sure you make the  #
   # appropriate changes to /etc/mcastbootd.conf                          #
   #                                                                      #
   # In the mboot class, the bootrom version (e.g. 1.32) must be given in the #
   # vendor-class-identifier.  If multiple bootrom versions need to be    #
   # supported multiple match cases may be used.                          #
   #                                                                      #
   ########################################################################



   ########################################################################
   # Class "AmiNET530 mboot"                                              #
   # AmiNET530 - response to bootrom request for a bootstrap image        #
   ########################################################################
   class "AmiNET530 mboot"
   {
     match if (option vendor-class-identifier="aminoAMINET530mboot<bootrom
version>") or
              ((substring(option vendor-encapsulated-options, 2,
9)="AMINET530")
               and (substring(option vendor-encapsulated-options, 13,
5)="mboot"));

     vendor-option-space AMINO;
     option AMINO.address 225.50.54.52;
     option AMINO.port 11111;
   }

   ########################################################################
   # Class "AmiNET530 upgrd"                                              #
   # AmiNET530 - response to bootstrap request for a main upgrade image   #
   ########################################################################
   class "AmiNET530 upgrd"
   {
     match if (option vendor-class-identifier="Aminoaminet530upgrd") or
              ((substring( option vendor-encapsulated-op-
tions,2,9)="aminet530")
              and (substring( option vendor-encapsulated-options,13,5)="up-
grd"));

     vendor-option-space AMINO;
     option AMINO.address 225.50.54.53;
     option AMINO.port 11111;
   }

   ########################################################################
   # Class "AmiNET530 fisys"                                              #
   # AmiNET530 - response when booting in normal boot state               #
   ########################################################################
   class "AmiNET530 fisys"
   {
     match if (option vendor-class-identifier="Aminoaminet530fisys") or
              ((substring( option vendor-encapsulated-options, 2,
9)="aminet530")
               and (substring(option vendor-encapsulated-options, 13,
5)="fisys"));
```

```
  vendor-option-space AMINO;
  option AMINO.middleware <mcast address>;
  option AMINO.mw_port <port>;
}


#########################################################################
# Subnet Declaration                                                    #
#########################################################################
subnet 192.168.1.0 netmask 255.255.255.0 {

#########################################################################
# Default Gateway - This MUST be set!!                                  #
#########################################################################
option routers 192.168.1.1;

#########################################################################
# Subnet Mask - This MUST be set!!                                      #
#########################################################################
option subnet-mask 255.255.255.0;

#########################################################################
# Domain Name - Optional                                                #
#########################################################################
option domain-name "blahblah.com";

#########################################################################
# DNS Servers  - Optional                                               #
#########################################################################
option domain-name-servers 192.168.1.1,192.168.1.2;

#########################################################################
# Time Offset - Optional                                                #
#########################################################################
option time-offset -5; # Eastern Standard Time

#########################################################################
# Address Pool - This MUST be set!!                                     #
#                                                                       #
# In this address pool we list the classes which we wish to give addresses #
# to, unless a device is in this list it will not be given a address!   #
#                                                                       #
#########################################################################


pool {
        range dynamic-bootp 192.168.1.50 192.168.1.100;
        range 192.168.1.101 192.168.1.200;

          }

}
```

```
#########################################################################
# End dhcpd.conf                                                        #
#########################################################################
```

For a range of set-top boxes

Replace the `pool` section of the `dhcpd.conf` file with:

```
pool{
    range 192.168.1.50 192.168.1.100;

    }
```

where `range` is the range of the IP addresses for multicast booting and upgrades, in this case from `192.168.1.50 - 100`.

For individual set-top boxes

In the case where you have a smaller number of set-top boxes and wish to manage them individually, as in a development network, the pool information should be replaced with STB specific information similar to the following:

```
host aminet110_001 {
              hardware ethernet 00:03:0A:0C:92:91;
                fixed-address 10.172.227.64;
                   }

host aminet110_002 {
             hardware ethernet 00:02:0A:13:AA:61;
             fixed-address 10.172.227.65;
                   }
```

Appendix C—The mcastbootd.conf file

The `mcastbootd.conf` file (located in the `/etc` file folder) configures parameters for the `mcastbootd` application to supply a bootstrap and filesystem to Amino set-top boxes via IP multicast over the network. The `mcastbootd.conf` file also works in conjunction with information supplied to Amino STBs via `dhcpd/dhcpd.conf`, the `STBremoteconf` application or the STB's management pages/configuration files.

C.1 mcastbootd.conf file components

This section describes the main parts of the `mcastbootd.conf` file.

C.1.1 File header

```
# Configuration file: <weekday> <month> <day> <hr>:<min>:<sec> <year>
```

Each time the `signupgradeimage` script is run it updates the timestamp in the Configuration file line.

C.1.2 [Server] section

This is the part of the `mcastbootd.conf` file where global parameters are set.

Global Server Parameters

`LogLevel= <number>` Sets the debug log level from 0 to 10 for the `mcastbootd -D` parameter. The default value is 4.

`ImageDir= <file path>`
Sets the common file path location where `mc2` images and bootstraps are stored.

`MulticastTTL= <number>`
Sets the number of router hops to allow the multicast packets to traverse in order to contain the scope of the multicast image. The default value is `1`.

`MulticastInterface = <interface>`
On a system with multiple interfaces this option specifies which interface to use for the Multicast traffic.

This is specified as the name of the interface:

```
MulticastInterface=eth0
```

C.1.3 Bootstrap and mc2 filesystem section commonalities

`MulticastIPAddress=<Multicast IP Address>`
Sets the Multicast IP Address an Amino STB will join to download the bootstrap and the `mc2` filesystem. This must match the corresponding section of the `dhcpd.conf` file. The Multicast IP Address and port combination should be unique and unused by other applications on the network.

`MulticastUDPPort=<Port Number>`
Sets the Multicast Port Number

`Description` `<any text can go here>`

`SerialNumber` `<integer>` originally designated to control filesystem upgrades, but not fully implemented. Number should not be altered or removed as it may cause `mcastbootd` not to function properly. `SerialNumber` is incremented each time the `signupgradeimage` script is run.

`CycleTime` `<integer>` Repeat cycle every `n` seconds or `0` for repeat continuously. Defaults are set at `5` seconds and `repeat continuously` for bootstrap and filesystem respectively.

`DataRate` `<integer>` Data rate in KB/s. The default for the bootstrap is 128KB/s = 1 Mb/s. The default for the `mc2.mcfs` is also 128KB/s = 1 Mb/s. An image using these two settings would output a multicast stream of roughly 2 Mb/s aggregately.

A quick way to convert the DataRate from KB/s to Mb/s is do simply divide the KB/s by 128. For example, 128KB/s = 1 Mb/s, 256KB/s = 2 Mb/s, 512KB/s = 4 Mb/s.

Note: The bandwidth allocated should take into consideration that the bootstrap and `mc2` filesystem multicast are delivered simultaneously. This means, for example, that the bandwidth set for the bootstrap image must be added to the bandwidth set for the mc2 filesystem when considering total network bandwidth consumed by the `mcastbootd` application.

C.1.4 Bootstrap Section

`PacketSize` `<integer>` Sets the size in Bytes of bootstrap packets to conform to network MTU size requirements and prevent packet fragmentation. Default is set at 1456.

`FileName` `<absolute or relative filepath><bootstrap filename>` The default is set to `<relative path of the present working folder>/bootstrap.signed`

`ImageType` 1

C.1.5 mc2 Filesystem Section:

`DirsPerCycle` `<integer>` The n number of times to multicast a filesystem from beginning to end before moving to the next filesystem (assuming there is more than one). The default is set to 128.

`PacketSize` not used in the `mc2` filesystem section. The PacketSize for the mc2 filesystem section is instead controlled by the '`signupgradeimage -s<PacketSizeInBytes <integer>>`' process.

C.2 Example mcastbootd.conf file

```
# Configuration file: Fri Apr 11 12:39:40 2003
[Server]
LogLevel=4
MulticastTTL=1
[Image bootstrap.signed]
MulticastIPAddress=225.50.50.52
MulticastUDPPort=11111
FileName=bootstrap.signed
Description=Linux bootstrap image
ImageType=1
SerialNumber=1
PacketSize=1456
CycleTime=0
[Filesystem mc2]
MulticastIPAddress=225.50.50.53
MulticastUDPPort=11111
ImageName=mc2
Description=upgrade filesystem
SerialNumber=2
DirsPerCycle=128
DataRate=256
CycleTime=0
[Filesystem mc2]
MulticastIPAddress=225.50.50.254
MulticastUDPPort=11111
ImageName=mc2
Description=upgrade filesystem
SerialNumber=2
DirsPerCycle=128
DataRate=256
CycleTime=0
```

C.3 Troubleshooting and tips to measure and test bandwidth/MTU/TTL

The following sections describe troubleshooting tools and common problems.

C.3.1 Common network monitoring/bandwidth measuring tools

- Ethereal/Tethereal/Wireshark/Tcpdump: These are common pcap/winpcap file generating sniffing applications that can capture or display network traffic.
- Bwm/bwm-ng: Useful network interface bandwidth monitoring tool that can display bandwidth usage on an ethernet interface.
- Ping/Traceroute/Tracert: Common ICMP protocol tools that can help understand basic connectivity, mtu and topological information.

Helpful Linux Commands – may require `sudo` prefix:

- `ps -ef | grep mcastbootd | grep -v grep` - A command to show details of how many instances of `mcastbootd` may be running (only one can run at one time).
- `pgrep mcastbootd` - Returns on the Program ID's (PIDs) of the running `mcastbootd` processes.
- `kill <pid1> <pid2> …` - Stops running the instances by PID number.

- `killall mcastbootd` - Stops running all instances of `mcastbootd`
- `netstat -rn` - Shows all of the routes on the system.
- `Ifconfig` - Shows all interface information on a system.
- `route add -net 224.0.0.0 netmask 240.0.0.0 eth0` - A command that routes all multicast traffic out the `eth0` interface.

C.3.2 dhcpd interaction

The `dhcpd.conf` file interacts with the mcastbootd.conf file by providing the STB with the multicast IP addresses and ports used to get the bootstrap and mc2 filesystem in order to upgrade the STB.

The multicast IP address and port for each STB model mboot section corresponds with the `mcastbootd.conf` bootstrap multicast IP/Port combination for that STB model.

Similarly, the multicast IP address and port for each STB model upgrade section corresponds with the `mcastbootd.conf` `mc2` file multicast IP/Port combination for that STB model.

The `fisys` section can contain a multicast IP address and port when used for auto-upgrading STBs. (see the Amino *Set-Top Box Configuration Guide*)

C.3.2.1 Common problems/mistakes

1. `mcastbootd` not running - The STB will usually provide an LED flashcode of 2:3 or 2:4.
2. Too many instances of `mcastbootd` running. The STB will usually provide a flashcode of 2:3.
3. No route for multicast. - Check routes/add multicast route
4. Firewall blocking route. - Check firewall permissions.
5. Wrong `mc2` file STB type. Means either DHCP is pointing STBs to wrong `mc2` file or `mc2` file is misnamed/incorrect. Usually results in a flashcode of 2:7.
6. File not found - Incorrect file path set for image/file. The `mcastbootd.conf` file can provide path information three ways:
   - using the global setting of `ImageDir`
   - specifying the absolute or relative path in the `ImageName` and/or
   - `FileName` lines or running `mcastbootd` from the folder containing all of the images.

   Only one strategy should be used for consistent results. Secondly, make sure that the actual files for the `mc2.mcfs` files contain the `.mcfs` file extension. However, the reference in the `mcastbootd.conf` file will not contain the extension.
7. The `signupgradeimage` script fails. There are three main reason why this script can fail:
   - There is no dummy mc2 filesystem entry in the `mcastbootd.conf`. The `sign upgrade image` script checks for a dummy `mc2` filesystem like the last entry in the sample above.
   - The `CUSTOMER_KEY` is not properly exported or
   - Utilities have not been copied into the `/us/local/bin` folder from the proper location in the build.
8. Multicast not making it to the STB through the routed network. The multicast TTL needs to be increased. Usually indicated by an LED flashcode of 2:3 or 2:4, but it can be verified that a single instance of `mcastbootd` is multicasting and is able to exit the server.

Index

B

Bootstrap
    See also software images 6, 20
Bootstrap directory 5
Bootstrap image 6, 20
    distributing 7
Bootstrap state
    DHCP request 38
    DHCP response 38
Bootstrap.unsigned 5
Build options 4
    Development builds 4
    Live builds 4

C

Commands file 6
config.txt 6
Configuration pages
    mcnfg.tar 6
createtftpboot 5
customise script 6

D

Development build
    Build options 4
DHCP request
    Bootstrap state 38
    Filesystem state 42
    Upgrade state 41
DHCP response
    Bootstrap state 38
    Filesystem state 42
    Upgrade state 41
DHCP server configuration 7
dhcpd.conf 5

F

Files
    Software release directory contents 5
Filesystem state
    DHCP request 42
    DHCP response 42
    Operation 41
fkeys.conf 6
flashcontents file 5

G

Getting the software 8
Graphic
    splash.gif 37
Graphics
    loading.gif 6
    splash.gif 6

I

imagecomponents directory 6
imgcfg file 6

J

JMACX 7

K

Keys
    Amino engineering keys
        keys directory 6
    Key.private 6
    key.public 6
    master.public 6
    Security Keys 34
    STBrc-key.private 6
    STBrc-key-public 6
keys directory 6

L

LED
    Flashing codes
        List 58

Live build
Build options 4
loading.gif 6

M

Management pages 7
mc2.mcfs 5
mcastbootd binary file 5
mcastbootd.conf 5
mcnfg.tar 6
Multicast system
Copying release files 9
Prerequisites 8

N

netconf file 6

O

opera.ini 6

P

Passwords
shadow file 55
Prerequisites
Setting up Multicast system 8
Private configuration key 6
Private key 6
Production build
see Development build
Public configuration key 6
Public customer key 6
Public master key 6

R

Release directory contents 5
romupgrade directory 5
rsakeys 6

S

Security
Keys 34
server directory 5
settings file 6
Set-top box operation
Filesystem state 41
Upgrade state 39
shadow file 55
signbootstrap script 5
signrom script 6
signupgradeimage script 5
Software builds 3
Software images
Bootstrap 6, 20
Distributing 7
Signing 34
Upgrade 7, 20
Software release
Directory contents 5
splash.gif 6, 37
Static DHCP serttings 36
STBremoteconf 7
STBremoteconf file 6

T

tftpboot directory 5
Troubleshooting
LED flashing codes
List 58

U

Unsigned bootstrap image 5
Upgrade image 7, 20
Distributing 7
See also Software images
Upgrade state
DHCP request 41
DHCP response 41
Operation 39
upgradeimage directory 5
Upgrading software
Upgrarde image
utils directory 6

V

verifysignature script 6