Course Locker Maintenance Guide

Contents

1.0 Introduction

Lockers are used for many different things in the Athena Computing Environment and many different people are responsible for maintaining, upgrading, and supporting the contents of these lockers. This document addresses some of the more common issues and questions surrounding the maintenance of a particular type of locker used by academic courses. Called course lockers, these lockers are used for storing files used by courses including but not limited to handouts, data files, web pages, and applications for students to run.

1.1 Topics Covered

Since this document is intended for course locker maintainers, it focuses on issues pertaining to course lockers. Some topics may apply to lockers in general, but this document does not attempt to set the standard for all types of lockers.

Section 2.0 provides some basic background information about lockers in general and describes appropriate and inappropriate uses of course lockers. This section is useful for people who have never dealt with a locker before. Section 3.0 addresses the issues of quota, access control, and locker configuration. Section 4.0 details some of the technical issues behind the way course lockers are created, allocated, and managed, and may be useful to more advanced locker maintainers. Finally, Section 6.0 addresses commonly asked questions in an easy-to-read and understand format.

1.2 How to Use this Document

If you are new to Athena and course lockers, you should probably read through the introductory sections Section 2.0 and Section 3.0 for background information about lockers. If you are simply looking for more details about configuring a course locker or want to know why things are done a certain way, read Section 4.0. Finally, if you simply want a quick answer to a specific question, look for in the frequently asked questions list in Section 6.0. If you do not see the answer to your question here, please send email to *f_l@mit.edu* and we will be glad to both answer it and update our FAQ list.

1.3 Terms and Definitions

This document makes use of the following terms and commands.

file server
In a distributed environment such as Athena, file servers are machines which store files. The remote file system used by most of the Athena environment, AFS, makes the files accessible to you without you knowing which machine stores them.

locker
In the Athena environment, a locker is the name given to a unit of disk space which is stored on a file server and is accessible via the attach command.

AFS
AFS is the specific remote file system software run on most Athena file servers. AFS comes from The Transarc Corporation.

quota
Quota is a a term used to mean the amount of disk space allocated. Under AFS, quota is allocated on a per-volume basis, meaning that all the files and directories in a volume share the same quota allocation. Most Athena lockers consist of a single volume and therefore all files in a locker usually share the same quota allocation.

file system
The term file system refers to the hierarchy of files and directories starting from the root. Standard UNIX machines have a local file system with the root at /. The root of the AFS hierarchy is at /afs. Most athena lockers attach under /mit.

volume
Under AFS, disk space is broken up into cells and within cells, volumes. MIT supports several cells, the two most familiar being the Athena cell and the sipb cell. Within a cell, space is allocated and managed in terms of something called volumes. The most important thing for the average Athena user to know about a volume is that quota is allocated on a per-volume basis.

add
add is a command local to the Athena environment which attaches a locker and modifies the user's command and manual page search paths. add makes it easy for Athena users to access software stored in lockers. For more information see the add man page.

attach
attach is an Athena command used to access to locker. attach figures out where the locker is stored in the remote file system and makes a logical connection under /mit to the remote location.

athrun
athrun is a command introduced in Athena 8.4 (summer 2000) which allows you to run a program from a specific locker in one step, without using an add command first. (It attaches the locker for you and locates the specified program without referencing or modifying your path; in particular, this ensures that the program comes from the locker you asked for instead of somewhere earlier in your path.) For more information see the athrun man page.

fs sa
The AFS command fs sa is used to set the access control list on a directory or directories. For more information type `fs sa -help`.

fs la
The AFS command fs la allows you to list the access control list on a directory or directories.

find
find is a standard UNIX command which searches for files or directories according to specified criteria. It is useful for setting or listing ACLs on an directory tree.

rm
rm is a standard UNIX command used to remove files permanently.

delete
delete is an Athena command which removes files non-destructively. rm is forever; delete is not.

undelete
undelete is an Athena command which undoes a delete.

purge
purge is an Athena command which permanently removes files marked for deletion by the delete command.

mkdir
mkdir is a UNIX command which creates a new directory.

du
du is a UNIX utility which summarizes the disk space used by a directory hierarchy or hierarchies.

1.4 Conventions

- `This is text that the user enters.`
- **This is the name of a command which can be run at the shell prompt.**

The athena shell prompt athena% should not be typed.

In an example which describes the syntax of a command, angle brackets (<>) denote a required parameter and square brackets ([]) denote optional parameters. In neither case should you actually type the brackets.

Many of the commands shown in this document expect to have a directory name or names as parameters. In each case, directories may be named with an absolute pathname, that is, starting with / at the top of the hierarchy, or they may be relative to the current directory.

1.5 Other Sources of Information

This document does not cover every situation and issue surrounding course lockers. If you have any questions particular to your locker, feel free to contact the Faculty Liaison Office

(f_l@mit.edu, x3-0115, N42-040), or the OLC consultants. To reach the consultant's office, type `olc` at the athena% prompt or call x3-4435.

The public mailing list locker-maintainers@mit.edu is for announcements and discussion relevant to people maintaining software in Athena lockers. You may add yourself to this list using listmaint or by typing:

```
athena% blanche locker-maintainers -a $user
```

Mail sent to the list is archived in the locker-maintainers discuss meeting on menelaus. You may view this on the web.

2.0 General Information

2.1 What are Lockers Used For?

Course lockers are useful for storing all kinds of data related to an academic course. Course lockers may contain web pages, handouts and problem sets or problem set solutions for students to view online, or programs for students to run. Some people store data files in lockers; others develop software specifically for a course and put it in the locker for students to run. Lockers will store just about anything subject to resource and security limitations. It's up to each locker maintainer to decide what is appropriate, but everyone with write access to the locker should be made aware that Section 7 of MIT's Student Information Policy stipulates that some types of course-related information must be restricted to class participants.

2.2 What are Some Limitations to Lockers?

Networked file space is always more limited than what you may have on your own computer; we have limited resources and creating lockers above a certain size is problematic (for technical reasons we need to divide larger allocations into separate AFS volumes). Because of this, lockers are not ideal for storing space-intensive data such as sound and movies.

In addition to complying with MIT's Student Information Policy restrictions, you should be very careful about using a locker to store data which you consider sensitive, such as solutions for upcoming problem sets. Make sure you understand access control settings before putting anything in the locker which shouldn't be public.

Course lockers are specifically for course-related activities. They are not to be used for storing personal or research-related files. Finally, course lockers are not intended to be used for student work space. If you feel your students need more disk space for course work than is available in their home directory space, please contact the Faculty Liaison Office to discuss an extra allocation for the duration of the course.

2.3 What are the Different Types of Lockers?

In addition to course lockers, there are many other types of lockers on Athena. A locker's type identifies it to the operations staff and also determines where it lives in the file system hierarchy. The following table describes other common locker types.

TABLE 1.

Type	Description	Location in AFS hierarchy
system	system files	/afs/athena/system
software	commercial software	/afs/athena/software
user	home directories	/afs/athena/user
project	Athena-related projects	/afs/athena/project
dept	Departmental leased disk space	/afs/athena/dept
org	Organizations within MIT. org lockers are typically used for web pages	/afs/athena/org
course	course locker allocations	/afs/athena/course
urop	urop projects	/afs/athena/course/urop
activity	Recognized MIT activities	/afs/athena/activity

3.0 Maintaining a Course Locker

3.1 Quota

Each volume in a course locker has been assigned a quota - a limit to its size on disk. The following sections tell you how to view the quota usage and how to clean out old files that are wasting space.

3.1.1 How to determine your quota usage

The AFS command **fs lq** (**F**ile **S**erver **L**ist **Q**uota) command displays the quota allocated to and used by any locker. Since AFS quota is allocated in terms of volumes and not lockers, **fs lq** is not guaranteed to list the entire quota allocated a locker. However, most course lockers consist of a single volume, making the numbers returned by **fs lq** accurate.

The syntax of the **fs lq** command is

```
fs lq <list of directories>
```

where <list of directories> is a space separated list of one or more directory names. Example 1 shows how to use **fs lq** to list the quota on the locker 25.678, if it really existed, that is.

```
Example 1  Listing the quota on the 25.678 locker
-------------------------------------------------
athena% attach 25.678
```

```
attach: /afs/athena.mit.edu/course/25/25.678 linked to /mit/25.678 for file
system 25.678
athena% fs lq /mit/25.678
Volume Name            Quota    Used     % Used   Partition
course.25.678          15000    13016      87%         82%
```

fs lq reports quota allocated and used in terms of kilobytes. This example locker is using slightly more than 13 MB of a 15 MB quota (15000 KB = 15 MB). There will also be a warning indicator if the percentage used is greater than 90%. The field labeled Partition refers to the file server where this locker lives and is important only to Athena operations. Note that this example shows the default quota from several years back; when new lockers are created they are given a higher default (100 MB for fall 2001), but older lockers are not automatically increased; please contact us if you need more space.

3.1.2 Finding where the quota is being used

Once you have determined how much quota your locker has used, you may also want to figure out which files or directories are using the quota. The **du** command returns the amount of disk space used by files or directories. Example 2 shows how to use **du** to compute how much space two directories are using.

```
Example 2  Using du to compute disk usage
----------------------------------------
athena% cd /mit/25.678
athena% du -sk Fall94 Fall95
2345Fall94
8945Fall95
```

This indicates that the directory Fall94 is using 2.3 MB; Fall95 is using 8.9 MB.

3.1.3 How to clean out old files and directories

Most lockers tend to grow continuously, but you can help this process by cleaning out old and useless files periodically. While only you can know if a particular file is old or not, there are several kinds of files which can be created by programs as temporary or backup files and which can be safely removed.

TABLE 2.

File Name	What Is It?
file~	A file ending with a tilde (~) is an Emacs backup file.
file.o	Files ending with.o are object file from compiling. These files are used to build the actual executable.
#*	Files marked for deletion by delete program
core	Core files are information dumped by programs that die ungracefully. They tend to be very large.
file.dvi, file.aux, file.log	Files ending with.dvi,.aux, and.log are temporary files created by latex.

3.1.4 What is this OldFiles directory?

As you start looking around a locker for files to clean up, you will probably stumble across a directory named OldFiles. While it looks like this directory contains duplicate information that may be eating up your quota, it in fact does not affect the locker's quota at all. You can think of OldFiles as a sort of on-line backup, containing the contents of your locker as it existed yesterday.

The OldFiles directory is extremely useful if you accidently delete a file from the course locker. You can copy the file from the OldFiles backup into the main locker as shown in Example 3.

```
Example 3  Copying a file from OldFiles
---------------------------------------
athena% cd /mit/25.678
athena% rm myfile
oops!
athena% cp OldFiles/myfile ./myfile
```

Note that OldFiles is read-only; you can copy files out of it to the rest of the locker, but you cannot make any changes to it. For more details, see this OLC Stock Answer.

3.1.5 Getting more quota

So, you have deleted everything you can think of and you're still approaching 100% of your quota. In this case, send email to the Faculty Liaisons (*f_l@mit.edu*) and explain how much you think you need for what locker and why you need it. We'll get back to you as soon as possible.

3.2 Access Control

The term access control refers to the mechanism which controls which individuals or groups can access files and directories. In short, access control lists, or ACLs for short, determine who can read files, who can write files, and who cannot access files in a locker.

3.2.1 Introduction to AFS file permissions

Under AFS, file and directory access is managed by access control lists. Unlike NFS, where individual files can have access control permissions, AFS maintains ACLs on a per-directory basis. For example, if the ACL on a directory allows a person or persons read access, then he, she, or they have read access to all the files in that directory. ACLs may be set for individuals and/or groups, with different types of permissions detailed below.

The ACL on different directories may be completely different; ACL modifications affect only a single directory unless you specify otherwise. ACLs are inherited in the sense that when you create a new subdirectory, it gets an ACL identical to the parent directory's (but you can subsequently change either the parent or child directory's ACL without affecting the other one).

Table 3 lists the seven types of permissions that can be given to users or groups of users.

TABLE 3.

Right	Enables users to
r	read the contents of files in the directory
l	list the names of files in the directory
i	insert files into the directory
d	delete files from the directory
w	write or modify files in the directory
k	lock (or modify the write-mode bit) of files in the directory
a	administer or change the ACL of the directory

No rights imply any others; for example, a user with "write" permission is not automatically given "insert" permission as well.

While this sounds complicated, there are four standard combinations of access which are applied to course lockers. Table 4 lists these standard combinations.

TABLE 4.

Combination	Shorthand	Meaning
rl	read	Read-only access. Users with rl access can read files and traverse directories.
rlidwk	write	Write access. Users with rldwk access can create new files and delete and modify files, but cannot modify the ACL.
rlidwka	all	All access. Users with rlidwka access can do everything above plus changes the ACL.
none	No access.	

3.2.2 The default access control list

By default, course lockers are created with an ACL that allows faculty, TAs, and the Faculty Liaisons all access, and everyone else on Athena read access. By convention, when the Faculty Liaisons create a new course locker, we also create a group with the same name as the locker. We put the Athena user ids of any current TAs, faculty members, and other people who will be administering the locker into this group. This group then gets all access to the locker. Therefore, adding or deleting TAs to the access control list is as simple as modifying the membership of a group.

3.2.3 Listing the ACL on directories

The AFS command **fs la** (**F**ile **S**erver **L**ist **A**ccess) allows you to view the ACL on any directory. Type this command to view the ACL set on a directory or directories:

```
athena% fs la [directory1 directory2 ...]
```

where [directory1 directory2 ...] is a space separated list of zero or more directory names to be examined. (To see the current directory, the directory list can be left out, making the command simply **fs la**.)

```
Example 4  Listing the ACL on a Course Locker
---------------------------------------------
athena% fs la /mit/25.678
Access list for /mit/25.678 is
Normal rights:
  system:authuser rl
  system:25.678 rlidwka
  system:facdev rlidwka
  system:expunge ld
  joeprof rlidwka
```

The list contains pairs of users or groups, and their respective permissions on that directory (groups appear as system:*groupname* and users appear simply as *username*). In this example, because joeprof is the owner of the locker, he has all access, as do members of the groups 25.678 and facdev. (facdev is a group consisting of all the Faculty Liaisons, and is there for convenience if you request our help in checking or modifying course locker ACLs, directories, etc.) system:authuser corresponds to anyone on Athena, meaning that anybody with an Athena account can read and lookup files, and system:expunge allows the system to automatically purge files which you have marked for deletion.

Here are some of the possible group specifications you can make (e.g., as a *user-or-group* field in an **fs** command), including several special system-owned groups:

Group	**Purpose**
system:authuser	Any AFS user with valid Kerberos tickets in the same cell (e.g., under athena.mit.edu). For all practical purposes, this is any user at MIT. By default, this group has read access to the course locker; if you wish to restrict portions of the locker, see the next section.
system:anyuser	Any user, including AFS users not at MIT. By default, this is on a course locker with list access at the top-level, and with read access on the www directory (to allow files you place there to be viewed through any web browser; list access at the top-level is necessary for web browsers to have "pass-through" access to the www subdirectory).
system:expunge	The process which runs automatically on your fileserver to remove old **delete**'d files permanently. This group is given **ld** access to your directories by default, so that the process can look up the old **delete**'d files and remove

	them.
system:htaccess.mit	A special group (used in conjunction with an .htaccess.mit file) to restrict access to web pages to specified MIT users. For details, see our guide to Protecting Content with Certificates
system:*groupname*	A system-owned (Moira) group, whose members can be edited with **listmaint** or **blanche**. To create a group for access to course locker, contact the Faculty Liaisons.

3.2.4 Changing ACLs

The AFS command **fs sa** (**F**ile **S**erver **S**et **A**ccess) allows you to change an ACL on a directory if you already have all access. The syntax of the **fs sa** command is as follows:

```
fs sa directory user-or-group ACL
```

where user-or-group is either an Athena user id or the name of a group proceeded by the string system:, e.g. system:facdev or system:25.678.

The two most common reasons to change an ACL in a course locker are to control web page access, and to limit access to the locker and/or directory contents to a group of students in the course.

For web page access, you may choose to have directories world-readable or restricted; this is done on a per-directory basis, so you can have different parts of your web site with different privacy levels.

The restrictions can be set to allow only specific MIT users or groups, or to allow any MIT user (with any of these restrictions, non-MIT users will not be able to view your pages). For details, see our Protecting Content.

For locker access through Athena (AFS), you can restrict directories to be readable only to students taking the course by removing group system:authuser from the ACL and adding a group corresponding to the students.

```
Example 5  Limiting Access to a Certain Group
---------------------------------------------
athena% fs sa /mit/25.678/data system:authuser none
athena% fs sa /mit/25.678/data system:25.678-students read
```

Note that any subdirectories you later create under this directory will inherit the modified ACL:

```
Example 6 Inheriting an ACL from the parent directory
-----------------------------------------------------
athena% cd /mit/25.678/data
athena% fs la
```

```
Access list for . is Normal rights:
  system:25.678 rlidwka
  system:expunge ld
  system:facdev rlidwka
  joeprof rlidwka
  system:25.678-students rl
athena% mkdir hw-1
athena% fs la hw-1
Access list for hw-1 is
Normal rights:
  system:anyuser rl
  system:25.678 rlidwka
  system:expunge ld
  system:facdev rlidwka
  joeprof rlidwka
  system:25.678-students rl
```

To set the ACL on more than one directory, or to add more than one user-or-group and ACL pair, use the following command:

```
fs sa -dir dir1 [dir2]  -acl user-or-group1 acl1 [acl2]
```

```
Example 7  Setting the ACL on more than one directory
-----------------------------------------------------
athena% cd /mit/25.678
athena% fs sa -dir ProblemSets Handouts -acl system:anyuser read
```

```
Example 8  Adding multiple ACLs to a directory
----------------------------------------------
athena% fs sa -dir forjane -acl joeprof write janeuser all
```

You may specify more directories and/or more user-or-group and ACL pairs.

To set the ACL on a directory hierarchy, you can use the find and **fs sa** commands as in the following example.

```
Example 9  Using find to set the ACL
------------------------------------
athena% cd /mit/25.678/www
athena% find . -type d -exec fs sa {} -acl system:anyuser read \;
```

The syntax of find may be a little confusing, but you can get more help from the find man page, OLC, or the Faculty Liaisons..

For more help with AFS permissions, see the Making Your Files Accessible section of Working on Athena (AC-11).

3.3 Managing Executables

Many instructors want to make executable programs available to students via a course locker. Due to the nature of the Athena environment, this is slightly more complicated than just plunking a binary into the locker. The guidelines outlined here conform to the standards used by Athena

developers and make the course locker easy to maintain and course software easy to use. The **lockers** man page is the definitive description of Athena locker organization conventions.

3.3.1 Introduction to binary directories

Historically, locker maintainers have used a separate directory to store binaries for each supported platform, or workstation type. In the early days of Athena, the supported platforms were Digital VAXstations and IBM RTs; later platforms included Digital DECstations and IBM RISC/6000s. Currently supported platforms are Sun SPARCstations and Ultras, RedHat Linux, and SGI Indys and O2s. Until the introduction of the SGI Indys, the locker configuration guidelines suggested a binary directory for each platform named according to the value returned by the machtype command plus the string bin. Thus, lockers could contain directories named vaxbin, rtbin, sun4bin, decmipsbin, and rsaixbin, respectively.

However, it became apparent that this naming scheme did not contain enough information because sometimes a binary which ran on one version of the operating system would not run under a different version. The new naming scheme looks for binaries to be stored in a directory under

```
arch/$ATHENA_SYS/bin
```

where $ATHENA_SYS is a system-wide environment variable set to a machine-specific value. This naming scheme is detailed in Section 4.0 Technical Information and summarized in Table 5. If you follow this convention, students will be able to use the **add** or **athrun** commands to quickly access a program in a locker.

```
Example 10  Using add or athrun to Access a Program
-------------------------------------------------
athena% add 25.578
athena% myprog

athena% athrun 29.123 yourprog
```

This works because the above commands know how to find binaries configured according to these conventions.

See the **add** and **lockers** man pages for more information.

TABLE 5: Binary conventions

Current Values (Athena 9.0)	
Directory Name	**Platform and Operating System**
arch/sun4x_58/bin	Sun, Solaris 8 (a.k.a. 2.8)
arch/sgi_65/bin	SGI, Irix 6.5 (no change)
arch/i386_linux24/bin	Linux, Red Hat 7.1

```
Previous Values
-----------------------------------------------------
Directory Name        Platform and Operating System
-----------------------------------------------------
arch/sun4x_57/bin      Sun, Solaris 7 (a.k.a. 2.7)
arch/sgi_65/bin        SGI, Irix 6.5
arch/i386_linux22/bin  Linux, Red Hat 6.2
arch/i386_linux3/bin   Linux, Red Hat 5.2 (SIPB, desupported 6/2001)
arch/i386_nbsd1/bin    NetBSD (SIPB, desupported 6/2001)
arch/sun4x_56/bin      Sun, Solaris 2.6
arch/sgi_65/bin        SGI, Irix 6.5
arch/sun4x_55/bin      Sun, Solaris 2.5
arch/sun4m_54/bin      Sun, Solaris 2.4
arch/sgi_63/bin        SGI O2, Irix 6.3
arch/sgi_62/bin        SGI Indy, Irix 6.2
arch/sgi_53/bin        SGI Indy, Irix 5.3
arch/pmax_u14/bin      Digital DECstation, Ultrix 4.3
arch/rs_aix32/bin      IBM RISC/6000, AIX 3.2
-----------------------------------------------------
```

The system type that will initially be used for Project Pismere and Windows 2000 systems is `i386_nt40`.

3.3.2 Configuring executables

There are two cases to consider when configuring executables. In the first case, users run the program without any special environment settings, or else are expected to set up the environment for themselves. The second case are programs for which you want to set up the environment for students before invoking the executable itself.

The first case is simple; just put the executables into the directories as described above and tell users to use **add** as in Example 10.

In the second case, the program installed in the binary directory will actually be some sort of startup script which sets the environment and then invokes the real binary. Example 11 shows a simple startup script that sets an environment variable, attaches an extra locker, and then invokes a binary. For this example, the real binary has been renamed to start with a capital letter so that the script and the binary can co-exist in the same directory. Note also that we use athdir to find the correct binary directory.

```
Example 11  Using a Startup Script
----------------------------------
#!/bin/csh -f
setenv TEMP /var/tmp
attach infoagents
set bindir=`athdir /mit/25.578`
exec $bindir/Myprog $*
```

If you have questions about writing startup scripts, please contact one of the Faculty Liaisons for assistance.

3.3.3 Source code

If you have source code for a program or programs developed here at MIT or elsewhere, we recommend that you install it in a subdirectory called src. If you expect to build and install this code for more than one platform, we also recommend the use of Imakefiles to simplify the process. Imakefiles are beyond the scope of this document, but feel free to contact the Faculty Liaison Office for assistance.

3.3.4 Removing binaries for obsolete platforms

With the passage of time, some platforms become obsolete and are no longer supported in the Athena environment. The following platforms are no longer supported: Digital VAXstations, IBM RTs, Digital DECstations, IBM RISC/6000s. You may safely remove any of these binaries to free up locker space and we encourage you to do so. These binaries would typically be stored in directories named vaxbin, rtbin, decmipsbin, and rsaixbin, respectively.

3.3.5 Dealing with operating system upgrades

One side effect of the new naming conventions for binary directories is that locker maintainers must be aware of operating system upgrades. We recommend adding yourself to the locker-maintainers mailing list, to receive announcements of upgrades, new sysnames, and other relevant information in timely fashion (see section 1.5 for details on adding yourself to the list). OS upgrades occur approximately once a year as part of the Athena Release; the ACS Insider newsletter also provides information on upcoming releases.

Whenever an operating system is upgraded, you should make sure that all your binaries run correctly. At the very least, this requires that you create a new link in your arch subdirectory for the new operating system version (see 3.3 Managing Executables) and run each program on a workstation running the new release. At the most, it will require recompiling software for which you have sources or possibly getting a new version from the vendor. ACS will be glad to help you in this process. We also have test workstations which you may use to set up and check your binaries; contact f_l@mit.edu for more information.

During the summer of 2001 we upgraded the operating system on Sun workstations to Solaris 8 and Linux machines to RedHat 7.1; SGIs will be stay at IRIX 6.5.7. The corresponding system values are:

```
sun4x_58      Solaris 8
sgi_65        Irix 6.5
i386_linux24  Red Hat Linux 7.1
```

There are two different ways to set up new binary directories in a course locker.

1. If all of your binaries work correctly under the new operating system, you can just make the new directory a symbolic link to the old one.

```
        Example 12  Creating a Symlink to the Old Binary Directory
        ------------------------------------------------------------
        athena% attach 25.578
        athena% cd /mit/25.578/arch
        athena% ln -s sun4x_57 sun4x_58
```

7. If any of the binaries need to be rebuilt, you should create a separate directory for the new operating system.

```
        Example 13  Creating a New Binary Directory
        -------------------------------------------
          athena% attach 25.578
          athena% cd /mit/25.578/arch
          athena% mkdir sgi_65
```

Note: If an old binary still runs under the new operating system, you can just create a link to it in the new directory.

```
        Example 14  Sharing Some Binaries between Operating System
Versions
        ------------------------------------------------------------------
---
        athena% cd sgi_65
        athena% ln -s ../sgi_63/prog1
```

3.4 Other Types of Files and Directories

If you have questions about configuring other types of files or directories, please contact the Faculty Liaison Office.

4.0 Technical Information

4.1 AFS

4.1.1 About AFS

For a thorough discussion of AFS, see Athena publication Working on Athena (AC-11). This section covers only those topics applicable to course lockers and leaves out many details.

AFS is organized by units called cells. Within each cell, quota is allocated in terms of volumes. Most of the entities we call lockers consist of a single volume, though it is possible to have multi-volume lockers. Volumes are mounted in the AFS file system hierarchy which begins at /afs. Mount points appear to be directories in the afs hierarchy. Everything in the Athena cell is

located under /afs/athena.mit.edu (/afs/athena for short), and all course volumes are under /afs/athena/course. Thus, you do not have to attach a locker to gain access to its contents, though we strongly recommend this for many reasons.

AFS supports different types of volumes: read-write volumes, read-only volumes, and backup volumes. The directory OldFiles in your home directory or course locker is simply a mount point to the backup volume for the volume comprising the locker. (At least, the volume if the locker consists of a single volume.) AFS manages quotas on a per-volume basis rather than a per-user basis, so anyone with write access to a volume can use up its quota allocation.

4.1.2 Groups and AFS

AFS does not use Athena groups directly but rather maintains its own database of groups and group members. When you create a new group or make changes to an existing group, there is an automatic propagation of the changes into the corresponding AFS group. AFS groups are used to set access control lists and are identified by a system: prefix. Examples of AFS groups are system:facdev, system:2.14, and system:anyuser. system:anyuser is a special group that corresponds to anyone using AFS whether or not he or she has an MIT user id, and system:authuser is anyone who has authenticated themselves locally in the Athena cell. This corresponds roughly to anyone on Athena.

4.1.3 The @sys link

AFS supports a special string which can be used in path names and dynamically expands to a special value according to the machine type. This @sys string is sometimes used as a user convenience link in configuring binaries or libraries. For example, you can make the file path /mit/locker/bin evaluate to the correct binary directory for each supported platform. In other words, /mit/locker/bin can point to the correct SGI binary directory if the user is on an SGI, or the Sun binary directory if he/she is using a Sun and so on. The command to set up this link is shown in Example 15.

```
Example 15  Using @sys for binary directories
---------------------------------------------
athena% cd /mit/locker
athena% ln -s arch/@sys/bin bin
```

This will only work if you configure the locker according to the conventions outlined in this document in Section 3.3 , and should only be used for convenience. The **add** command will not find a directory named simply bin! You must follow the conventions outlined in Section 4.2

Note: the string "@sys" should never be used literally, except in making symlinks as above. See Section 4.2.3 on the **athdir** command for references in scripts, makefiles, etc.

4.2 The Athena Binary Directory Convention

4.2.1 Historical context

We used to have a convention of naming binary directories according to the output from the machtype command: vaxbin, decmipsbin, rtbin, vaxbin, and sun4bin. As the number of supported platforms grew, lockers became unwieldy and cluttered with all these bin directories at the top level. Furthermore, as Athena development was responsible for deciding what the machtype command returned, the naming scheme was rather arbitrary and did not support different operating system versions. Under the current scheme, all machine dependent directories are placed under a directory called arch. Under arch is one directory for each supported platform, named after the AFS @sys values (sgi_52, sun4m_53, etc.) The **lockers** man page explains this in more detail, and also covers backwards compatibility issues.

4.2.2 $ATHENA_SYS and $ATHENA_SYS_COMPAT

The environment variable $ATHENA_SYS is set in the system-wide dotfiles at login time; it is used by commands such as **add** and **athrun** to locate the appropriate files for the system on which the user is currently working. The value of $ATHENA_SYS matches the AFS value for the @sys link and can also be determined according to the output of the command machtype -S:

```
athena% machtype -S
sun4m_53
```

When writing your own shell scripts, makefiles, etc., it is recommended that you use **athdir** to locate machine-dependent files (as described in the next section) rather than $ATHENA_SYS; **athdir** is more reliable in case of new operating system releases and changed locker conventions.

The environment variable $ATHENA_SYS_COMPAT was introduced to help lockers which have not yet been updated for a new operating system continue to function until the maintainers have updated their arch directory structure. The system-wide dotfiles set $ATHENA_SYS_COMPAT to a fallback list of @sys values which are known to be generally compatible with the current system; if the correct @sys value isn't supported by a locker, commands such as **add** and **athdir** will attempt to use a value from this fallback list instead. On releases prior to Athena 9.0, the user may see a message like this:

```
athena% add badlocker
add: warning: using compatibility for /mit/badlocker
```

Note: Compatibility is intended as a mechanism to keep lockers working temporarily; it is not to be relied on as a substitute for updating binary directories when operating systems are upgraded. See Section 3.3.5 Dealing with Operating Systems Upgrades for recommended procedures.

4.2.3 athdir

Along with the environment variable $ATHENA_SYS and the new locker conventions, Athena introduced the **athdir** command. **athdir** can be used to find any type of machine dependent directory in a locker. One common use of **athdir** is to locate the machine-specific binary directory in a locker as in Example 16. As this example shows, **athdir** recognizes both the old and new style directory configurations.

athdir can be used in Makefiles and startup scripts as shown in Example 17. **athdir** takes many options as described on its man page.

```
Example 16 The athdir command
------------------------------
athena% attach frame; athdir /mit/frame
/mit/frame/arch/sun4m_53/bin
athena% attach sipb; athdir /mit/sipb
/mit/sipb/sun4bin
```

```
Example 17 athdir in a Makefile
-------------------------------
LOCKER = "/mit/25.578"
BINDIR = `athdir $LOCKER`
INCLUDES = `athdir $LOCKER include`
LIBS = `athdir $LOCKER lib`
```

4.2.4 Using a shared directory for scripts

If you have many startup scripts or other programs which are actually scripts and therefore can be shared between different machine types, you may want to install them into a shared directory. You will still need links in the machine-specific directories, but installing into a single location makes it easier to maintain and update the script. The recommended configuration for using a shared directory is

```
/mit/locker/arch/share/bin
```

For each shared script you must create a soft link from the machine-specific binary directory to the shared installation location. Example 18 shows you how to do this for the sun4m_53 case (Solaris 2.3):

```
Example 18  Sharing a startup script
------------------------------------
athena% attach 25.678
athena% cd /mit/25.678
athena% mkdir -p arch/share/bin
athena% cd arch/share/bin
athena% emacs myscript &
create the script here
athena% chmod +x myscript
athena% cd ../../sun4m_53/bin
athena% ln -s /mit/25.678/arch/share/bin/myscript
```

5.0 Course Locker Re-use Policy

Course lockers are re-used from semester to semester. Locker owners/maintainers are advised that any files they leave behind in a course locker at the end of the semester may be deleted or re-used by course staff who later inherit the locker. At the end of the semester you are responsible for saving any files that you wish to keep, and for deleting anything which you do not wish to share with subsequent course staff.

When an existing course locker is reused in later terms, we try to notify the people already on the ACLs as a courtesy, but it is up to new course staff to follow up with previous course staff regarding the disposition of any existing files. We are happy to assist you with archiving existing files in an inherited locker, and clearing out or archiving your own files at the end of the term.

6.0 Answers to Frequently Asked Questions

This section contains some answers to frequently asked questions about course lockers. Answers which contain instructions and step-by-step detail do not contain any explanation or reasons. For further information you should consult the appropriate sections in this document.

1. Who manages the Athena menus? How can I request an update of a Courseware entry, or make something in a course locker available from the menu?

   The menus are maintained by the Faculty Liaisons (f_l@mit.edu, x3-0115). If you have questions, want anything changed for your course, or need help troubleshooting, just let us know.

2. What does the error "add: warning: using compatibility for /mit/xxx" mean?

   This means that the **add** command can't locate files according to the current @sys value for your platform, and is attempting to use a fallback value supplied by $ATHENA_SYS_COMPAT. Most likely, there has been an operating system upgrade, and the locker needs to have its arch directory structure updated; see 3.3.5 Dealing with Operating Systems Upgrades for help.

3. Why doesn't **add** find my Solaris binaries?

   If the **add** command is returning an error such as "Command not found", your locker is probably not configured correctly. Under Athena 9.0, all Solaris binaries should be put into a directory named according to the convention /mit/locker/arch/sun4x_58/bin.

4. My Linux binaries worked on the old SIPB Linux-Athena, but not on Linux-Athena in the clusters; why?

   The IS Linux-Athena ports began with Red Hat 6.2 (Summer 2000); Red Hat's backward compatibility is not as good as we're used to on other platforms, so you may have to recompile programs built under Red Hat 4.x. Specifically:

- Curses-using programs must be recompiled or they will try to look in `/usr/lib/terminfo`, which does not exist in Red Hat 6.x.
- Programs which print dates must be recompiled or they may try to look in `/usr/lib/zoneinfo`, which does not exist in in Red Hat 6.x.

5. Where should I put executables for platform <X>?

   If you are setting up a new locker, the best thing is to follow the new conventions for naming binary directories. Table 5 lists the directories and corresponding platforms.

6. Where should I put my web pages?

   New course lockers are created with a directory named www at the top level; place your web pages in this directory and they will be world-readable.

7. What's the URL for the locker's web page?

   The URL will follow the naming scheme:

   ```
   http://web.mit.edu/locker/www/
   ```

   If you name your top level page index.html, and place it in the www directory of your locker. (See also the Academic Web Page Creation Guide.)

8. Is there any way to restrict access to my web pages to MIT?

   Yes, there is. For more information, see the Protecting Content

9. How can I get more quota for my locker?

   First make sure that you have removed any files and directories which are no longer needed. Athena does not have unlimited resources! Once you have done this, send an email request to accounts@mit.edu stating what locker needs the quota increase, how much more you need, and why you need it. We will listen to each request, but because of resource limitations we cannot grant every one.

10. Last year the TA stored some files in his home directory. He left MIT; how can I get those files back?

    If you are fortunate, the TA's directory is still online. You can find out if this is the case by typing the command

    ```
    hesinfo username filsys
    ```

If this command does not return an error, the directory is still online. In this case, send email to accounts@mit.edu, telling us what files you need to access.

If the TA's directory has been deleted, we will need to restore from tape. To do this, send email to accounts@mit.edu giving us the TA's username and files you need to recover.

11. Last year students used a program in my course locker. Now suddenly it says "Command not found" when I use **add**. What happened?

    Are you using the recommended configuration for binary directories? There was probably a new Athena release and an operating system upgrade. You may be able to fix the problem simply by creating a symbolic link, but contact the Faculty Liaisons for more information. You may also want to read Section 4.2 The New Binary Directory Convention and Section 3.3.5 Dealing with Operating Systems Upgrades.

12. How can I find out what files recently changed in my locker?

    The UNIX find command is useful for finding recently modified files. Go to the top level of your locker and type the command

    ```
    find . -mtime -<n> -print
    ```

    where <n> is the number of days since the files changed.

    You can also get a good idea of what changed recently by using the **ls -ltr** command:

    ```
    ls -ltr | tail -20
    ```

    ls -ltr lists the files and directories in reverse order according to modification date. By piping this command into **tail**, it will list only the last 20 modified files or directories.