

# CONTRIBUTED DOCUMENTATION

This document is NOT supported by Computing Services.  
DO NOT contact the Help Center with questions on this document.

## AFS Authentication

---

### Introduction

To use the AFS distributed file system, you must become authenticated, or verified, as a valid AFS user. In order to become authenticated, you will need to obtain a token in an AFS cell so that you will be able to access, read, and write to files in your AFS account. This document covers information about tokens, obtaining tokens using the klog command, expiring tokens using the unlog command, cells, and authenticating to multiple cells using the cklog command.

### Getting and Destroying Tokens

A token is what proves you are authenticated. It is like a badge that companies give out; your password is what gives you this badge to access files. If your cache manager (the program that checks user access to files and retrieves them from AFS) can unscramble the token and use its information, you are verified as an authorized AFS user.

### System Security Techniques

Several security techniques are used in the AFS distributed file system: passwords, mutual authentication, and access control lists

#### Passwords

The first security technique in AFS is the use of passwords. A user can only gain access to a token when his/her password is verified. Once the cache manager receives a token, the user is approved as a valid AFS user.

#### Mutual Authentication

Another method of system security is mutual authentication; both parties communicate across the network, proving their identities to one another. AFS requires mutual authentication whenever a server and client (most often, a cache manager) communicate with each other.

Both passwords and mutual authentication ensure that users accessing files are valid AFS users.

#### Access Control Lists

The third security technique, access control lists, allow individual users to restrict access to directories they administer (see the online help for protection-groups and protection-setting for more information).

#### Checking your Tokens

To check if you are authenticated, use the tokens command to list the tokens you have. To see your tokens,

type tokens and press Return.

If you are authenticated, tokens lists your tokens, the AFS ID they are issued to (the protection group ID), the cellnames the tokens are issued in (the file space you are working in), and the expiration date:

```
Tokens held by the Cache Manager:  
User's (AFS ID 1022) tokens for afs.andrew.cmu.edu [Expires Jan 3 14:35]  
User ju33's tokens for krbtgt.ANDREW.CMU.EDU@andrew.cmu.edu [Expires Jan 3 14:35]  
User ju33's tokens for zephyr.zephyr@andrew.cmu.edu [Expires Jan 3 14:35]  
--End of list--
```

In the previous example, the AFS tokens listed for AFS ID 1022 (ju33) expire on January 3 at 2:35 p.m. for cell afs.andrew.cmu.edu. Also note the user's tokens for other Kerberos services (e.g. tokens for zephyr).

If you are not authenticated because your tokens have expired, you will see the same information listed above, but your expiration date will be listed as "Expired." The following output will then appear:

```
Tokens held by the Cache Manager:  
--End of list--
```

## Destroying Tokens

It is good practice to issue the unlog command when you leave your workstation for any period of time without logging out. Without doing this, anyone can use your directory to look at your files, send mail, or make bboard posts that will name you as the sender. If you use the unlog command, other users can still use your account to an extent, but not with your tokens: they can not prove your identity on any actions they perform, and they can not do everything that you normally could do, such as delete all your files. When you return to your workstation, use the klog command to re-authenticate yourself and regain access to your directories. See [klog](#) and [unlog](#) sections of this document for more information.

## Expiring Tokens

Some jobs (e.g. writing files) require you to have a token. If you have been logged in continuously for a long period of time, it is possible that your tokens will expire. If you are experiencing a high number of error messages while working or if you receive a message stating that your tokens have expired, use the klog command to re-authenticate and get new tokens. See the next section on the klog command for more information on re-authenticating.

---

## klog: AFS Authentication

The klog command creates a token for you in a particular cell, replacing any existing tokens you may have in that cell. If your cell uses the AFS login program, klog is an automatic feature. The andrew.cmu.edu cell at Carnegie Mellon uses the AFS login program, but other cells on campus may not. More information on cells and the cklog command are discussed later in this document.

## What if my tokens expire?

The difference between how the login and authentication processes work have some consequences:

You may be logged into the system, but your tokens have expired. In this case, you can still access AFS, but

you can not access protected files or directories. Tokens eventually expire automatically, even if you do not logout. Without a token, AFS will no longer recognize you as an authenticated AFS user, and you will need to re-authenticate using the klog command.

Conversely, you may be logged out, but your tokens are still active. A token can last until it expires or until you use the unlog command to destroy it. The andrew.cmu.edu cell destroys your tokens automatically when you logout, but other cells may not. If you are not sure whether or not the cell you are working in performs this automatic destruction of tokens, it is best to issue the unlog command (see next section for more information) before logging out, just in case.

## Authenticating using klog

The general syntax of the klog command is:

```
% klog  
Password: <password>
```

When the password prompt appears, type your password for your user ID. Once the system prompt appears, you have authenticated successfully and you have a new token. You can then use the token command to verify that you are authenticated.

---

## Unlog: Unauthenticating with AFS

Unlog is a command that allows you to leave your workstation without having to worry that another user will access your private files while you're away. Unlog is the only guaranteed way to keep unauthorized users from accessing your account.

## Unauthenticating using unlog

Unlog destroys your tokens so that other users cannot access your file directories. You can then issue the tokens command to verify that your tokens have been destroyed. If you wish to resume work, use the klog command to re-authenticate yourself.

When you decide to stop working and leave your workstation, but you don't want to log out, use the unlog command. At the system prompt, type **unlog** and press Return.

---

## AFS Cells: Carnegie Mellon's Distributed File System Cells

A cell in the AFS file system's file space acts as an administrative domain. A cell's domain can consist of a company, a university department, or any defined group of users. Each cell is autonomous; its administrators determine how workstations are configured and how much storage space is available to each user. At the same time, each cell can also connect with the file space of other cells running AFS. The result is a huge file space, a shared "community file space," composed of the all the individual cells using AFS. These cells may be geographically distant, allowing file sharing across the country as easily as across the campus. At Carnegie Mellon, for example, Andrew, Civil Engineering, Computer Science, and Electrical and Computer Engineering are all separate cells.

## Listing Cells

Listing the AFS cells

For a complete listing of AFS cells visible from your workstation, at the system prompt type:

```
ls /afs
```

## Listing your Local Cell

The cell controlling the workstation or unix server you are using is called your local cell. Other cells in the AFS file space are referred to as foreign cells. To list which cell your workstation is in, at the system prompt type:

```
fs wscell
```

For more information about the fs command, see help on fs.

## Authenticating in Multiple Cells

### Cross-cell authentication

Cross-cell authentication lets you access files in foreign cells where you have an account. To become an authenticated AFS user, you need to provide an authentication password to AFS each time you login. In cells using the AFS login program, logging in is a one-step process; your initial login provides authentication (the Andrew cell uses the AFS login program).

In other cells, you may have to use the klog command to become authenticated to AFS -- contact the system administrators of that cell and/or the workstation administrator. The klog command is used to "register" you with AFS -- this form of authentication is referred to as cross-cell authentication. Cross-cell authentication requires you to have accounts in each cell you wish to authenticate to.

### Cross-Cell tokens: Authenticating in foreign cells

You can have tokens in use simultaneously for your local cell and for any foreign cells in which you have an account. To obtain tokens in foreign cells, use the klog command with your cross-realm identity. A cross-realm identity is in the form

```
<userID>@<cellname>
```

For example, if you have a Computer Science account in the CS cell and you decide that you want to authenticate in that cell, you can type:

```
% klog <CSuserid>@cs.cmu.edu  
Password: <yourCSpasword>
```

and press Return. You will then have a token in the cs.cmu.edu cell.

**Note:** You must have an account in the cell in which you are trying to authenticate.

If you then issue the tokens command, you may see similar output as shown below (the Andrew userid is ju33 and the CS userid is bovik):

Tokens held by the Cache Manager:

```
User's (AFS ID 1022) tokens for afs.andrew.cmu.edu [Expires Jan 3 14:35]
User's (AFS ID 1229) tokens for afs.cs.cmu.edu [Expires Jan 3 14:35]
User ju33's tokens for krbtgt.ANDREW.CMU.EDU@andrew.cmu.edu [Expires Jan 3 14:35]}
User bovik's tokens for krbtgt.CS.CMU.EDU@andrew.cmu.edu [Expires Jan 3 14:35]
User ju33's tokens for zephyr.zephyr@andrew.cmu.edu [Expires Jan 3 14:35]
--End of list--
```

This indicates that the cache manager has the tokens that identify the above user as "ju33" to the Andrew cell, as well as the tokens that identify the user as "bovik" to the CS cell.

## Cross-realm Authentication

It is also possible to establish what is known as cross-realm authentication between cooperating cells. With cross-realm authentication, users can share information between their respective cells, without having accounts in each cell. The `cklog` command is used to establish cross-realm authentication.

```
cklog
```

The `cklog` service uses what is called Kerberos cross-realm authentication. This extends the standard AFS cross-cell authentication that uses the `klog` command to establish authentication. `cklog` allows users to grant file access to people who do not have accounts in the cell by adding these people to their access control lists. The service also makes it easier for people who have accounts in two different cells to access one account from the other.

If you have an account in the CS cell, you can run `cklog cs.cmu.edu`. When you then check with the `tokens` command, you will receive the following similar response:

```
Tokens held by the Cache Manager:
User's (AFS ID 0) tokens for afs@cs.cmu.edu [Expires Apr 29 17:40]
User's (AFS ID 99999) tokens for afs@andrew.cmu.edu [Expires Apr 29 17:36]
User ju33's tokens for krbtgt.ANDREW.CMU.EDU@andrew.cmu.edu [Expires Apr 29 17:36]
User ju33's tokens for krbtgt.CS.CMU.EDU@andrew.cmu.edu [Expires Apr 29 17:40]
--End of list--
```

In the previous example, Joe User is known to both the Andrew and CS cells as "ju33," rather than in the previous example where he was known as "ju33" to Andrew and "bovik" to CS.

## How does `cklog` work?

`cklog` creates an AFS cross-realm token for a user, presents that token to the protection server database in the target cell, then adds the token to access control lists in the target cell. A cross-realm token has the form

```
<userID>@<cellname>
```

To create a cross-realm token, use the `cklog` command in the form

```
cklog <cellname>
```

For example, if you want to set up a cross-realm token with the Computer Science cell from your Andrew account, type

```
cklog cs.cmu.edu
```

Issuing the above cklog command does several things:

- It provides cross-realm tokens for the CS cell to the AFS cache manager on your Andrew workstation, which will be used when contacting the CS cell file servers. (A cache manager is the program that checks user access to files and retrieves them from AFS.)
- The cklog program also checks to see if there is an entry for this Andrew cross-realm token in the protection server database in CS and creates one if there is not. Once your Andrew identity has been established with the CS cell, CS users can place your Andrew identifier token on their access control lists so that you can access their files.

You need to set up your cross-realm token only once. However, you will still need to present your identification to the AFS cell before you can access any files in that cell. Use the cklog command to do this as well. In fact, if you find you need to access the a particular AFS cell frequently, you can put the **cklog <cellname>** command in your .login file so that this happens automatically each time you login.

## Giving access to other cells using cklog

You can use cklog to setup a directory in another cell (such as CS) and allow users in other cells to access that directory. For example, a CS user with the userid jqs@cs.cmu.edu may have a directory called /afs/cs.cmu.edu/user/jqs/project. She may have also created a protection group called projmems for this directory (see the online help for protection-groups for further information). If the group now wants to give a user without a CS account access to the same directory, follow these steps:

1. While logged into his Andrew account, the non-CS user acquires a cross-realm token by typing the following at his system prompt

```
cklog cs.cmu.edu
```

This action secures a cross-realm token for that user in the CS cell and assures an entry of the cross-realm identity (<userid>@andrew.cmu.edu) in the protection server in the CS cell. The tokens command can be used to verify that you has a cross-realm token.

2. Once this identifier has been acquired, CS user jqs can now add it to the projmems protection group on the access control list for their project directory (see the online help for protection-setting to learn more about access control lists). To verify that the non-CS user is now in the projmems group, CS user jqs can use the pts examine command (see the online help for protection-groups for more information).
3. When the non-CS user wants to access CS user jqs' project directory, all they have to do is present their cross-realm token to the CS cell by typing the following at the system prompt

```
cklog cs.cmu.edu
```

## cklog to gain access to another cell

Another use for cklog is to gain easy access to your account in one cell from your account in another cell:

If you have a CS account, the standard way to authenticate yourself in the CS cell for your Andrew account would be to use the klog command, which requires you to type in your CS password before you are authenticated with your CS identity.

Using cross-realm authentication, however, you only needs to obtain a cross-realm token for yourself in the CS cell and add your cross-realm user ID (<userid>@andrew.cmu.edu) to the access control list for your CS directory. Once you have done this, you can use the cklog command each time you wishes to access his CS account. The steps for setting this up are identical to the previous example:

1. While logged into his Andrew account, you uses the `cklog cs.cmu.edu` command to obtain the cross-realm token.
2. You then adds your cross-realm ID (`<userid>@andrew.cmu.edu`) to the access control list for your CS account.
3. To access your CS directory, you again issue the **`cklog cs.cmu.edu`** command to present your cross-realm identity to the CS file servers.