Linux security can be somewhat confusing since it has evolved over time. The typical Linux distribution supports the following access controls for files and directories:

* [Discretionary Access Control (DAC)](http://electronicdesign.com/dev-tools/whats-difference-between-posix-acls-and-selinux#dac)
* [Access Control List (ACL)](http://electronicdesign.com/dev-tools/whats-difference-between-posix-acls-and-selinux#acl)
* [Mandatory Access Control.(MAC)](http://electronicdesign.com/dev-tools/whats-difference-between-posix-acls-and-selinux#mac)

The access controls typically support inheritance along the lines of the directory hierarchy. We will not delve into all these details but restrict the discussion to the differences of the three appraoches.

Related Articles

[MILS, MSL, MLS: Figuring Out All Those Secure Acronyms](http://electronicdesign.com/embedded/mils-msl-mls-figuring-out-all-those-secure-acronyms)

[Secure Linux and Wind River](http://electronicdesign.com/embedded/secure-linux-and-wind-river)

[Can The Internet Of Things Be Secure?](http://electronicdesign.com/dev-tools/can-internet-things-be-secure)

[Traveling With Portable Secure Storage](http://electronicdesign.com/blog/traveling-portable-secure-storage)

**Discretionary Access Control**

DAC is the bit-oriented encoding that is normally displayed for the longer, more detailed file listing like this:

$ls -l  
-rw-rw-r--. 1 bwong bwong 779684 Jun 18 2013 more-files.zip  
drwxr-xr-x. 2 bwong bwong 4096 Jul 17 2013 Music  
-rwxrwxr-x. 1 bwong bwong 181 Mar 10 13:49 Note.txt  
drwxrwxr-x. 2 bwong bwong 4096 Jun 18 2013 Old  
drwxr-xr-x. 5 bwong bwong 4096 Oct 13 2012 Old-Home  
-rw-rw-r--. 1 bwong bwong 5030162 Apr 16 12:37 Paper airplane.zip

The access control bits are mapped to characters in the left column. A dash indicates the bit is not set. The 'd' in column one indicates a directory. There are then three sets of rwx columns that control the user's access, the group's access and everyone else. The 'r' indicates read access, 'w' for write access, and 'x' for execute or search access.

The access bits are usually listed as octal values for programs like **chmod** that changes the mode bits as they were sometimes called. For example, this command would set the bits to match the listing above for the Note.txt file. The directory status cannot be changed.

$ chmod 775 Note.txt

The user in this case is bwong (left) and the group is also bwong (right). The group is a list of users. This particular listing is a partial list of my home directory. The usual Linux approach is to create an empty group with the same name as the user.

This same approach is used in other operating systems such as DOS. It is simple to implement and uses little overhead in the file system. It is also how most users deal with their file system security. Microsoft Windows also uses this approach in the FAT file system.

With Linux, DAC is check first and subsequent control mechanisms are checked if access is not denied.

**Access Control List**

An access control list (ACL) is a more advanced approach to security. It can implement the user/group/anonymous user approach with the basic rwx attributes but typically implementations do much more. In the case of Linux, the POSIX access control list (ACL) is usually supported.

Access control lists can be attached to any object. In a directory listing using ls the existence of an ACL for a file or directory is indicated by a plus sign (+) at the end of the DAC characters.

An access control list can include multiple specifications. The Linux applications uses to change and check the ACL are **setfacl** and **getfacl**. A listing of the ACL using **getfacl** shows the difference between this and DAC support.

$ getfacl /home/bwong  
getfacl: Removing leading '/' from absolute path names  
# file: home/bwong  
# owner: bwong  
# group: bwong  
user::rwx  
user:qemu:--x  
group::---  
group:bwong:r-x  
mask::r-x  
other::---

Note that the DAC access is mirrored in this result. There are additional entries though. For example, the user qemu is allow execute access. Also note how groups are listed explicitly.

POSIX ACL support is part of the POSIX standard. This provides portability between operating systems for applications that utilize the POSIX API.

**Mandatory Access Control**

Mandatory Access Control (MAC) is a policy-based approach that provides even more control over how security is implemented and controlled. It extends past the basic access controls of DAC and ACL to allowing an administrator to have fine grained control over what changes users can make. With DAC, a user simply needs write access to be able to change the attributes of a file or directory. The ability to create a file requires write access to the directory a file will be created in.

MAC systems normally provide the ability to specify access attributes as well as the ability to see and modify those attributes. The ability to create and delete files or directories can provide fine grain control. In addition, files and directories can have attributes that can be matched against rules that can control where and how data can be used.

MAC systems usually extend their control beyond the file system. This allows network interfaces, ports and other logical and physical devices to be monitored. This approach can even extend to services such as a system's firewall. An application can be limited to the ports and interfaces they are allowed to use as well as the files, directories and other resources such as applications they have access to. [SELinux](http://selinuxproject.org/) is the MAC normally associated with Linux but there are others as well.

SELinux has the concepts of users, roles, types, contexts and object classes. The policies are implemented as rules. The advantage over ACLs is that the rules are not necessarily restricted to a directory or a file. For example, a rule can specify the only the web server can access a particular type of file regardless of where the file is located.

MAC policies can be complex. Default policies are often provided. For example, [Red Hat](http://www.redhat.com/) Linux uses SELinux. The packages for the system, such as the Apache web server, come SELinux attributes and profiles that allow the standard configurations to work after package installation but administrators can then alter those as necessary.

The ACL and MAC support in Linux is implemented in a modular fashion so it is possible to mix and match security systems although the majority of installations will employ POSIX ACLs and SELinux. The ability to change support is handy when dealing with network file systems where security attributes need to match the native server support. POSIX tends to be the dominate ACL implementation while MAC support varies