NAME

attr - Extended attributes

DESCRIPTION

Extended attributes are name:value pairs associated permanently with files and directories, similar to the environment strings associated with a process. An attribute may be defined or undefined. If it is defined, its value may be empty or non-empty.

Extended attributes are extensions to the normal attributes which are associated with all inodes in the system (i.e. the **stat**(2) data). They are often used to provide additional functionality to a filesystem − for example, additional security features such as Access Control Lists (ACLs) may be implemented using extended attributes.

Users with search access to a file or directory may retrieve a list of attribute names defined for that file or directory.

Extended attributes are accessed as atomic objects. Reading retrieves the whole value of an attribute and stores it in a buffer. Writing replaces any previous value with the new value.

Space consumed for extended attributes is counted towards the disk quotas of the file owner and file group.

Currently, support for extended attributes is implemented on Linux by the ext2, ext3, ext4, XFS, JFS and reiserfs filesystems.

EXTENDED ATTRIBUTE NAMESPACES

Attribute names are zero-terminated strings. The attribute name is always specified in the fully qualified *namespace.attribute* form, eg. *user.mime_type*, *trusted.md5sum*, *system.posix_acl_access*, or *security.selinux*.

The namespace mechanism is used to define different classes of extended attributes. These different classes exist for several reasons, e.g. the permissions and capabilities required for manipulating extended attributes of one namespace may differ to another.

Currently the *security*, *system*, *trusted*, and *user* extended attribute classes are defined as described below. Additional classes may be added in the future.

Extended security attributes

The security attribute namespace is used by kernel security modules, such as Security Enhanced Linux. Read and write access permissions to security attributes depend on the policy implemented for each security attribute by the security module. When no security module is loaded, all processes have read access to extended security attributes, and write access is limited to processes that have the CAP_SYS_ADMIN capability.

Extended system attributes

Extended system attributes are used by the kernel to store system objects such as Access Control Lists and Capabilities. Read and write access permissions to system attributes depend on the policy implemented for each system attribute implemented by filesystems in the kernel.

Trusted extended attributes

Trusted extended attributes are visible and accessible only to processes that have the CAP_SYS_ADMIN capability (the super user usually has this capability). Attributes in this class are used to implement mechanisms in user space (i.e., outside the kernel) which keep information in extended attributes to which ordinary processes should not have access.

Extended user attributes

Extended user attributes may be assigned to files and directories for storing arbitrary additional information such as the mime type, character set or encoding of a file. The access permissions for user attributes are defined by the file permission bits.

The file permission bits of regular files and directories are interpreted differently from the file permission bits of special files and symbolic links. For regular files and directories the file permission bits define access to the file's contents, while for device special files they define access to the device described by the special file. The file permissions of symbolic links are not used in access checks. These differences would allow users to consume filesystem resources in a way not controllable by disk quotas for group or world writable special files and directories.

For this reason, extended user attributes are only allowed for regular files and directories, and access to extended user attributes is restricted to the owner and to users with appropriate capabilities for directories with the sticky bit set (see the **chmod**(1) manual page for an explanation of Sticky Directories).

FILESYSTEM DIFFERENCES

The kernel and the filesystem may place limits on the maximum number and size of extended attributes that can be associated with a file. Some file systems, such as ext2/3 and reiserfs, require the filesystem to be mounted with the **user_xattr** mount option in order for extended user attributes to be used.

In the current ext2, ext3 and ext4 filesystem implementations, each extended attribute must fit on a single filesystem block (1024, 2048 or 4096 bytes, depending on the block size specified when the filesystem was created).

In the XFS and reiserfs filesystem implementations, there is no practical limit on the number or size of extended attributes associated with a file, and the algorithms used to store extended attribute information on disk are scalable.

In the JFS filesystem implementation, names can be up to 255 bytes and values up to 65,535 bytes.

ADDITIONAL NOTES

Since the filesystems on which extended attributes are stored might also be used on architectures with a different byte order and machine word size, care should be taken to store attribute values in an architecture independent format.

AUTHORS

Andreas Gruenbacher, <*a.gruenbacher@bestbits.at*> and the SGI XFS development team, <*linux-xfs@oss.sgi.com*>.

SEE ALSO

getfattr(1), setfattr(1).