Linux Ubuntu 22.4.5 Manual Pages on command 'attr.5'

$ man attr.5

XATTR(7)   Linux Programmer's Manual   XATTR(7)

NAME

    xattr - Extended attributes

DESCRIPTION

    Extended attributes are name:value pairs associated permanently with files and di?

    rectories, similar to the environment strings associated with a process. An attri?

    bute 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 at?

    tributes.

    Users with search access to a file or directory may use listxattr(2) to retrieve a

    list of attribute names defined for that file or directory.

    Extended attributes are accessed as atomic objects. Reading (getxattr(2)) re?

    trieves the whole value of an attribute and stores it in a buffer. Writing (setx?

    attr(2)) replaces any previous value with the new value.

    Space consumed for extended attributes may be counted towards the disk quotas of

    the file owner and file group.

  Extended attribute namespaces

    Attribute names are null-terminated strings. The attribute name is always speci?

fied in the fully qualified namespace.attribute form, for example, 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; for example, 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 Secu‐ rity Enhanced Linux, and also to implement file capabilities (see capabilities(7)). Read and write access permissions to security attributes depend on the policy im‐ plemented 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.

System extended attributes

System extended attributes are used by the kernel to store system objects such as Access Control Lists. Read and write access permissions to system attributes de‐ pend 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. Attributes in this class are used to implement mech‐ anisms in user space (i.e., outside the kernel) which keep information in extended attributes to which ordinary processes should not have access.

User extended attributes

User extended attributes may be assigned to files and directories for storing arbi‐ trary 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 permis‐ sion bits: read permission is required to retrieve the attribute value, and writer permission is required to change it.

The file permission bits of regular files and directories are interpreted differ‐ ently from the file permission bits of special files and symbolic links. For regu‐

lar files and directories the file permission bits define access to the file's con?

tents, 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, user extended attributes are allowed only for regular files and

directories, and access to user extended 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 the sticky bit).

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. The VFS imposes limita?

tions that an attribute names is limited to 255 bytes and an attribute value is

limited to 64 kB. The list of attribute names that can be returned is also limited

to 64 kB (see BUGS in listxattr(2)).

Some filesystems, such as Reiserfs (and, historically, ext2 and ext3), require the

filesystem to be mounted with the user_xattr mount option in order for user ex?

tended attributes to be used.

In the current ext2, ext3, and ext4 filesystem implementations, the total bytes

used by the names and values of all of a file's extended attributes must fit in a

single filesystem block (1024, 2048 or 4096 bytes, depending on the block size

specified when the filesystem was created).

In the Btrfs, XFS, and Reiserfs filesystem implementations, there is no practical

limit on the number of extended attributes associated with a file, and the algo?

rithms used to store extended attribute information on disk are scalable.

In the JFS, XFS, and Reiserfs filesystem implementations, the limit on bytes used

in an EA value is the ceiling imposed by the VFS.

In the Btrfs filesystem implementation, the total bytes used for the name, value,

and implementation overhead bytes is limited to the filesystem nodesize value

(16 kB by default).

CONFORMING TO

Extended attributes are not specified in POSIX.1, but some other systems (e.g., the

BSDs and Solaris) provide a similar feature.

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.

This page was formerly named attr(5).

SEE ALSO

getfattr(1), setfattr(1), getxattr(2), ioctl_iflags(2), listxattr(2), removex?

attr(2), setxattr(2), acl(5), capabilities(7) selinux(8)

COLOPHON

This page is part of release 5.05 of the Linux man-pages project. A description of

the project, information about reporting bugs, and the latest version of this page,

can be found at https://www.kernel.org/doc/man-pages/.

Linux   2019-08-02   XATTR(7)