



Silicon Graphics, Inc.

XFS Overview & Internals

07 – Extended Attributes

© Copyright 2006 Silicon Graphics Inc. All rights reserved.
Permission is granted to copy, distribute, and/or modify this document
under the terms of the Creative Commons Attribution-Share Alike, Version
3.0 or any later version published by the Creative Commons Corp. A copy of
the license is available at <http://creativecommons.org/licenses/by-sa/3.0/us/>.

November 2006

Extended Attributes

- Extended Attributes (EA) are a set of <name,value> pairs associated with an inode
- Who uses them?
 - Access Control Lists (ACL)
 - SELinux
 - Beagle indexer
- Name is a null terminated string <= 255 chars
- Value is binary data <= 64K

EA Namespaces

- The EA set is typically divided into namespaces
- For Linux the namespace is the prefix of the EA name:
 - user
 - trusted
 - security
 - System
- For XFS, the namespace is encoded in bits in a flags field with these values:
 - “user” has value of 0x0000 – by default an attribute is in the user namespace

```
#define ATTR_ROOT          0x0002 /* use attrs in root (trusted) namespace */
#define ATTR_SECURE         0x0008 /* use attrs in security namespace */
#define ATTR_SYSTEM          0x0100 /* use attrs in system (pseudo) namespace */
```

- So for XFS, EAs are really a triple <name, value, flags>

EA Command Line Interface

- attr package maintained by Andreas Gruenbacher and SGI
 - getfattr(1) for getting/listing EAs
 - setattr(1) for setting and removing EAs,
 - Names are prefixed with the namespace
- attr command sets/removes/gets/lists EAs
 - Based on IRIX command, provides common interface for XFS EAs
 - Namespace specified with options
 - More closely models what XFS actually stores since names are the actual names

```
# getfattr -e hex -dm '.*' file1
    system.posix_acl_access=0x020000001000600ffffffff040006...
    trusted.SGI_ACL_FILE=0x000000400000001ffffffff0006...
# attr -Rl file1
Attribute "SGI_ACL_FILE" has a 52 byte value for file1
```

EA Ondisk Format

- The name can actually be binary data since it has a length field on disk
 - XFS kernel functions have been changed to handle binary names (used for future parent pointer EA's)
- XFS EA's are stored in a variety of forms according to how big they are
 - Local or short form
 - within the inode
 - Attribute-fork extents in either
 - a filesystem leaf block
 - btree form with node blocks and leaf blocks
 - Extent form
 - the EA value can be in a remote filesystem block if it's large

EA Tuning

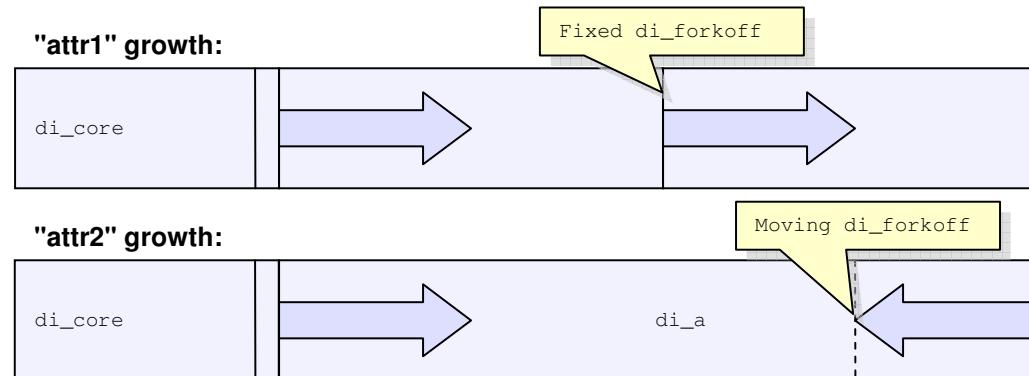
- If using ACLs, every file access will require an EA lookup
- For performance, it is important to keep the EA's within the inode in shortform
 - Only a single filesystem block needs to be read in.
 - ACLs are a good example where the access checks occur frequently
- To increase the chance of being in shortform
 - increase the inode size, and/or
 - enable attr2

```
mkfs.xfs -i size=512,attr=2 device
```

- The short-form EA resides at the end of the inode and competes for space with the data extents.

EA and Attr2

- Attr2 overcomes a restriction in the initial EA implementation that divided up the literal area at a fixed location (fork offset)
- With attr2 the fork offset is variable



EA and Backup

- Cpio and tar do not backup and restore extended attributes
- If using extended attributes, an EA backup tool like xfsdump and xfsrestore must be used

