

Using Extended Attributes for Data Driven Workflows

High Performance Computing
AWE

OakRidge National Laboratory, 19th September 2018

Introduction

- User and codes are not good at data management
 - Just scan your filesystems That's another conversation !
- Extended Attributes (EA) may be able to assist
 - Enhance existing metadata
 - Attach EAs to files/directories so they provide additional info
 - Query EAs in the inode without opening the object
 - Users/codes/scripts can tag data
 - Important data can now be identified
 - Workflows could be data aware using EAs
 - Data Staging/transfer scripts
 - Tag unnecessary data for deletion/expiry

GPFS Inode Structure

- **Header** (mandatory) - 128 bytes (GPFS or SS 4.2, likely the same for earlier versions)
- **Pointers to data blocks**
 - The file data (unless the file data fits in the inode = 4.1+). A DA may be a pointer to a block of data (=blocksize of data pool for that file system), or may be a pointer to a sub-block (1/32 of a storage pool block e.g. data pool is 16 MiB, sub-block is 1/2 MiB). Each DA is 12 bytes
- **Pointer to indirect blocks** (optional, depends on size of file etc)
 - Used if you need more DAs (block pointers) than will fit into a single inode. An indirect block is 32KiB or at least one sub-block in size
- **Extended Attributes** (optional, mandatory if using ILM/tiering)
 - EAs can be user defined, or created by the system (example when using ILM to Archive)
- **Pointer to Extended Attribute overflow block (if no room in inode for EAs)**
 - The overflow block size is either the metadata subblock size or 64 KiB, whichever is smaller. There is at most one EA overflow block per inode, allocated on demand during EA set operations, and freed when file is deleted

Inode EA limits

- **EA limits**
 - Total EA space per file : 64K + inode space
 - 1000 EAs per file, 16K size limit per EA
 - An E_NOSPC (a no space) error is returned when EA space is exhausted (err 28)
 - An invalid attribute error is return if the size of the EA exceeds the limit of 16K (err 22)
- **EAs have a higher priority than disk pointers**
 - Disk pointers moved to indirect pointers if space is needed for EAs
 - Priority in EAs
 - High : GPFS, DMAPI, archive
 - Medium : system, trusted, security
 - Low : user

Note - Above was valid for at least version 3.4, based on Yuri Volobuev's white paper (GPFS Wiki)

File System EA Support

- GPFS
 - `$ mmlsfs fsID | grep fastea`
- EXT4
 - `$ grep xattr /proc/fs/ext4/sda1/options`
user_xattr
- Lustre
 - Needs xattrs on mount line

So once the target and source file systems are enabled with the necessary extended attribute setting everything should work OK right ?

Setting User EAs

GPFS

```
$ mmchattr --set-attr user.code.name="Supernova_sim_3D" /gpfs/userA/nova1.h5
```

```
$ mmchattr --set-attr user.code.version="git-branch_0.01" /gpfs/userA/nova1.h5
```

```
$ mmchattr --set-attr user.code.group="astro" /gpfs/userA/nova1.h5
```

GNU (EXT4/GPFS/Lustre)

```
$ setfattr -x user.code.group -v astro /gpfs/userA/nova1.h5
```

Notes

- Can only use one attribute per set command
- EAs can contain multiple properties as in the case of the selinux EA

Querying EAs (GPFS)

```
$ mmlsattr -L -d /gpfs/userA/nova1.h5
file name:                /gpfs/userA/nova1.h5
metadata replication: 1 max 2
data replication:        1 max 2
immutable:               no
appendOnly:              no
flags:
storage pool name:       system
fileset name:            userA
snapshot name:
creation time:           Fri Aug 10 10:04:40 2018
Misc attributes:         ARCHIVE
Encrypted:               no
user.code.name:       "Supernova_sim_3D"
user.code.version:   "git-branch_0.01"
user.code.group:     "astro"
```

Querying EAs (GNU)

```
$ getfattr --absolute-names -d -m - /gpfs/userA/nova1.h5
# file: gpfs/userA/nova1.h5
security.selinux="unconfined_u:object_r:unlabeled_t:s0"
user.code.group="astro"
user.code.name="Supernova_sim_3D"
user.code.version="git-branch_0.01"
```

```
$ getfattr --absolute-names -n user.code.name /gpfs/userA/nova1.h5
# file: gpfs/userA/nova1.h5
user.code.name="Supernova_sim_3D"
```

Selinux EA is also visible to user via GNU ls command (ls -Z)

```
-rw-r--r--. userA userA unconfined_u:object_r:unlabeled_t:s0
/gpfs/userA/nova1.h5
```

Removing User EAs

GPFS Command

```
$ mmchattr --delete-attr user.code.name /gpfs/userA/nova1.h5
```

GNU Command

```
$ setfattr -x user.code.name /gpfs/userA/nova1.h5
```

Users cannot modify/remove reserved EAs

```
$ setfattr -x security.selinux /gpfs/userA/nova1.h5  
setfattr: /gpfs/userA/nova1.h5: Permission denied
```

```
$ setfattr -x system.checksum.sha256 /gpfs/userA/nova1.h5  
setfattr: /gpfs/userA/nova1.h5: Operation not permitted
```

Managing User EAs

```
>>> import xattr
>>> testFile="/gpfs/test-xattr.py"
>>> xattr.list(testFile)
['security.selinux', 'user.code.name', 'user.code.version']
>>> # Using the non privileged namespace flag removes the SE context
>>> xattr.list(testFile, namespace=xattr.NS_USER)
>>> xattr.set(testFile, 'user.code.version', 'Beta 0.1')
>>> xattr.set(testFile, 'user.code.name', 'GNUFind-test2')
>>> xattr.get_all(testFile, namespace=xattr.NS_USER)
[('code.name', 'GNUFind-test2'), ('code.version', 'Beta 0.1')]
>>> xattr.remove(testFile, 'user.code.version')
>>> xattr.remove(testFile, 'user.code.name')
>>> xattr.get_all(testFile, namespace=xattr.NS_USER)
>>> []
```

Embedding Data EAs

Example - XML string to provide additional metadata information in one string rather than multiple EAs

```
$ mmchattr --set-attr user.code.xml_tag="<desc><name>Supernova test  
run</name><comment>TypellA implosion simulation</comment></desc>"  
/gpfs/userA/nova1.h5
```

```
$ mmlsattr -L -d /gpfs/userA/nova1.h5 | grep "^user.code"  
user.code.name:      "Supernova_sim_3D"  
user.code.version:   "git-branch_0.01"  
user.code.group:     "astro"  
user.code.xml_tag:   "<desc><name>Supernova test  
run</name><comment>TypellA implosion simulation</comment></desc>"
```

ILM – Simple EA logging

Policy

```
define(USER_EA, GetXattrs('user.*','key:value^n;'))

rule external list 'user_attrs' exec ''
rule 'listall' list 'user_attrs' DIRECTORIES_PLUS WEIGHT(0)
SHOW('-- ' || USER_EA) WHERE LENGTH(USER_EA) > 0
```

Output

```
1048576 42019203 0 --
user.code.name:Supernova_sim_3D;user.code.version:git-
branch_0.01;user.code.group:astro;user.code.xml_tag:<desc><name>Su
pernovtest run</name><comment>TypeIIA implosion
simulation</comment></desc>; -- /gpfs/userA/nova1.h5
```

ILM – Simple EA Parsing

```
$ python
>>> a="1048594 1646716875 0 --
user.checksum.sha256:dd8f8f532467ac02a1f2a041e5fab1ce11a103efbf5204dbc789dbf5
b07eadbe;user.checksum.epoch:1533892227; -- /gpfs/userA/foo.txt"
>>>
>>> foo,fattrs,fname=a.split(' -- ')
>>> fname,fattrs[: -1]
('/gpfs/userA/foo.txt',
'user.checksum.sha256:dd8f8f532467ac02a1f2a041e5fab1ce11a103efbf5204dbc789dbf5
b07eadbe;user.checksum.epoch:1533892227')
>>>
>>> dict(a.split(':') for a in fattrs[: -1].split(';'))
{'user.checksum.epoch': '1533892227', 'user.checksum.sha256':
'dd8f8f532467ac02a1f2a041e5fab1ce11a103efbf5204dbc789dbf5b07eadbe'}
```

ILM – Extended Policy

Example - Additional information such as state of resident, migrated, premigrated file status to the user along with the populated user EAs :

```
define(is_premig, (MISC_ATTRIBUTES LIKE '%M%' AND MISC_ATTRIBUTES NOT LIKE '%V%'))
define(is_mig, (MISC_ATTRIBUTES LIKE '%V%'))
define(is_res, (NOT MISC_ATTRIBUTES LIKE '%M%'))
define(is_symlink, (MISC_ATTRIBUTES LIKE '%L%'))
define(is_dir, (MISC_ATTRIBUTES LIKE '%D%'))
define(OBJ_TYPE, (CASE WHEN is_dir THEN 'dir' WHEN is_symlink THEN 'symlink' WHEN
is_premig THEN 'file_pre' WHEN is_mig THEN 'file_mig' WHEN is_res THEN 'file_res'
ELSE 'unknown' END))

define(USER_EA, GetXattrs('user.*', 'key:value^n;'))

rule external list 'user_attrs' exec ''
rule 'listall' list 'user_attrs' DIRECTORIES_PLUS WEIGHT(0) SHOW('-- ' || OBJ_TYPE
|| ' -- ' || USER_EA) WHERE LENGTH(USER_EA) > 0 AND USER_ID != 0
```

GNU – Patched Tools

GNU findutils (4.6.0)

```
$ ./find /gpfs/userA -xattr user.code.name -print0  
/gpfs/userA/nova1.h5
```

```
$ ./find /gpfs/userA -xattr user.code.name -print0 | xargs -0 getfattr -n  
user.code.name --absolute-names  
# file: /gpfs/userA/nova1.h5  
user.code.name="Supernova_sim_3D"
```

Not as pretty as the BSD find command which provides the --xattr-name flag
so the xargs command is unnecessary !

GNU – Patched Tools

GNU ls (8.2.2)

Unfortunately files with selinux (security EA) show as having an EA so need an indicator to only indicate if the user.* attribute is present. The BSD ls command displays the @ symbol to indicate the presence of user EA as well.

ACL+context

```
-rw-rwx---+ 1 root  root          0 Apr 21 15:02 test-xattr.txt
```

Sec context

```
-rw-rw----. 1 userA userA 102400000000 Apr 21 15:52 test.100GB
```

User EA present

```
-rw-rw----@ 1 userA userA      41943040 Aug 10 10:04 nova1.h5
```

Application Support

Application	EA Support	Planned ?	Note
ls	NO	Unknown	Unofficial Patches
find	NO	Unknown	Unofficial Patches
cp	YES	-	Needs -a flag
mv	YES	-	
tar	YES	-	Needs -xattrs -xattrs-include='*' flags
cpio	NO	Unknown	
NFS 4.x	NO	Unknown	Stated as optional in NFS4.2+ proposal

Application Support

Application	EA Support	Planned ?	Note
scp/sftp	NO	NO	Author - not planned affects (sshfs/fish)
proftpd	YES	-	
rsync	YES	-	-X flag for EA and -A flag for ACLS but no ssh support
bbcp	YES	-	Not valid if you use scp as a transport
bbftp	YES	-	
shiftc	YES	-	

Issues

- User EAs
 - Inode space is finite
 - Could be easily overwritten/removed by user
 - Possible namespace collisions
- Limited/inconsistent GNU support
 - Use setfattr/getfattr (attr is for XFS)
 - OS and desktop support is lacking (file managers/editors)
- No NFSv4 support
 - Major restriction for end user
 - Files edited on NFS client (desktop) lose EAs
- No SSH support
- Be careful of transferring files across boundaries
 - EAs could be silently dropped
 - AFM support not tested

Benefits

- System EAs only visible by root
- ILM support
 - Hard to identify data in inode (RFE 120535)
 - Could use output as a EA backup mechanism but might miss changes
- TSM support
- Users can use EAs for a data aware workflow
 - Read EA metadata without opening file
 - Query/search EAs without recalling migrated file
 - Identify data for data transfer - hopefully will help to prevent “cp *”
- Encourage the use of tagged data
 - Hopefully will allow the expiry/deletion of transient/unwanted data
- Watch folders (LWE) with EAs ?

Questions ?

My gratitude to Nils Haustein (IBM)
Stephen Rollinson (AWE)
Paul Tomlinson (AWE)