DE LA RECHERCHE À L'INDUSTRIE

cea

www.cea.fr

Lecture on Parallel Filesystems

Lustre (4/5)

Jacques-Charles Lafoucriere

ENSIIE| 2018

Unix Right

Unix User

Unix user identification

- User ID, associated to a User name
- Primary Group ID, associated to a Group name
- Additional group IDs, associated to Group names

```
$ id
uid=1007(lafoucrierejc) gid=1000(teachers) groups=1000(teachers),2000(guests),2001(hpcvm)
```


Unix Standard File access

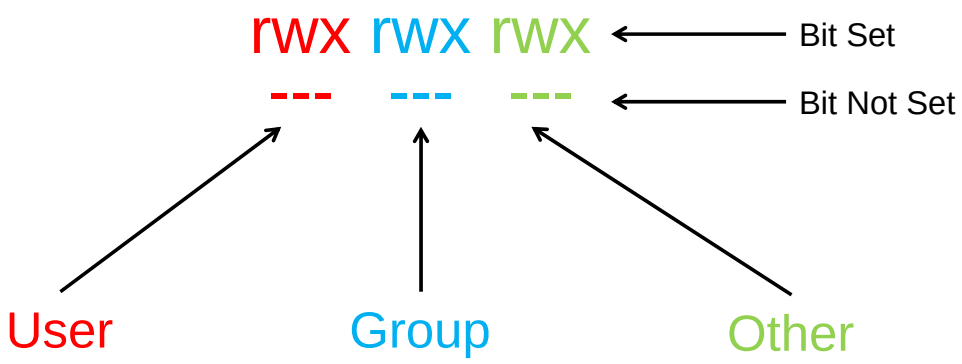
3 components are associated to a file/directory

- User
  - Match a UID
  - File/Directory owner
- Group
  - Match a GID
  - File/Directory group
- Other
  - All others User = all users except owner

3 modes are associated to each component

- Read
- Write
- Execute

Usual Representation of Unix Rights



Right Usage

File

- r access file content
- w modify file content
- x execute file
  - To run a shell script : need rx

Directory

- r list content
- w create, remove, rename an entry
- x cross the directory for searching
  - A file can be used even if the path cannot be listed

Special Bits

Set UID

- On file: at execution change the effective user id by file owner id
- On directory: when an entry is created, use directory user in place of user id (not everywhere and dangerous)
- Display as s if x is on or S if x is off

Set GID

- On file: at execution change effective group id by file group id
- On directory: when an entry is created, use directory group in place of user group
- Display as s if x is on or S if x is off

Sticky bit

- On directory forbid remove of user not owned entries
- On file “stick” the execution code in swap device (no more used)

How to Set/Change Unix Rights

chmod command line

Right rules

[ugoa]*([-+=]([rwxXst]*|[ugo]))+|[-+=][0-7]+

X Chmod Flag Use

- X: give x right if another x bit already exist

```
$ chmod 644 file ; ls -l file
-rw-r--r-- 1 LAFOUCRIEREJC None 0 May  6 15:13 file
$ chmod o+X file ; ls -l file
-rw-r--r-- 1 LAFOUCRIEREJC None 0 May  6 15:13 file
$ chmod u+x file ; ls -l file
-rwxr--r-- 1 LAFOUCRIEREJC None 0 May  6 15:13 file
$ chmod o+X file ; ls -l file
-rwxr--r-x 1 LAFOUCRIEREJC None 0 May  6 15:13 file
```


Access Control List

Context

Origin

- Standard Unix (POSIX) access right are too restricted
- Need to be able to give/remove
  - Rights to a specific user/group
  - Independently of file/directory owner/group

Solution: ACL

- A set of data that informs an operating system about permissions or access rights that each user or group has to specific system objects such as directories or files
- A unique security attribute
- Defined as a POSIX standard

ACL Definition

3 classes

- Owner
- Group
- Other

- Each ACL entry is
  - A class
  - A member of the class (no member == all members)
  - A set of rights rwx

ACL Display

Example of ACL

```
# file: spoo
# owner: root
# group: root
user::rw-
user:bin:rwx
group::r--
mask::rw-
other::r--
```


ACL Display (Cont.)

A more complex of ACL

```
# file: .
# owner: root
# group: root
user::rwx
group::r-x
mask::rwx
other::r-x
default:user::rwx
default:group::rw-
default:group:bin:r--
default:mask::rw-
default:other::r-x
```

ACL Rules

Minimal ACL

- user::--- group::--- other::--- : match std Unix rights

Extended ACL

- Named entries
  - user:NAME:---
  - group:NAME:---
- Mask entry
  - mask::---
  - Mask named user and group entries

ACL Rules (Cont.)

Default ACL

- default:user::---
- default:group::---
- default:user:NAME:---
- default:group:NAME:---
- default::other::---
- default::mask:---

- No role in access check
- Used to set ACL inherited at object creation

- Umask is ignored if an default ACL is present

ACL Access Check Algorithm

ACL entries are looked at in the following order

- Owner
- Named users
- Owning or named groups
- Others

- Only a single entry determines access

ACL Usage

How to know if an ACL exist?

```
$ ls -l file
-rw-rw-r--+ 1 root root 0 May  6 16:45 file
```

How create/change/read ACL?

```
$ getfacl
$ setfacl
```

Read ACL

```
$ getfacl object
# file: object
# owner: root
# group: root
user::rw-
group::r--
other::r--
$ getfacl -c object
user::rw-
group::r--
other::r--
$
```


Modify ACL

With commands option

```
$ setfacl -m rule1,rule2,… object
$ setfacl -b object # remove all ACL entries
```

From an ACL file

```
$ getfacl -c object > template.acl # get a template
$ vi template.acl # edit ACL file
$ setfacl -M template.acl object
```

How to Setup ACL on Lustre

Activate ACL support on MDT

```
# mkfs -t lustre --mdt […] --mountfsoption=user_xattr,acl /dev/MDT-Device
```

To check after MDT start

```
# lctl get_param -n mdc.test-MDT0000-mdc-*.connect_flags | grep acl
acl
```


Extended Attributes

Extended Attribute

Definition

- name-value pairs associated with a file or directory on a filesystem

Applications

- POSIX ACLs
- Lustre metadata
  - Layout
  - FID
  - Parent FID
  - …

Lustre Internal Extended Attribute

- trusted.som: size on mdt (not used)
- trusted.lov: file layout
- trusted.lma: obj FID (on MDT obj)
- trusted.lmv: directory layout
- trusted.dmv: defined to use xattr interface
- trusted.link: hard links vector (parent FID + others)
- trusted.fid: obj FID (on OST obj)
- trusted.version: defined to use xattr interface
- trusted.hsm: HSM EA
- trusted.lfsck_bitmap: lfsck namespace
- trusted.dummy: artefact

Lustre Internal Extended Attribute (Cont.)

```
$ touch file
$ lfs path2fid file .
file: [0x200000401:0x8:0x0]
.: [0x200000007:0x1:0x0]
$  getfattr --dump --match=- -e hex file
# file: file
trusted.lov=0xd00bd10b01000000080000000000000001040000020000000000100001000000040
000000000000000000000000000000000000000000000000
trusted.link=0xdff1ea11010000002e0000000000000000000000000000000000160000000200000
0070000000100000000666696c65
trusted.lma=0x00000000000000000104000002000000080000000000000
```


Root Squash

Root Squash

Objective

- Security feature
- Restrict super-user access rights from clients
  - Some client root is not root in Lustre namespace
- Useful if all clients does not share the same admin population

Implementation

- Root UID/GID is mapped to a defined UID/GID on the MGS

Root Squash Configuration

Parameters

- mdt.root_squash=UID:GID
- mdt.nosquash_nids="nids, nids, …"
  - Support use of jokers like *@tcp
  - List cleared with use of "clear" for the list

Root Squash Configuration (Cont.)

Set root squash on MGS

```
# lctl conf_param test.mdt.root_squash=501:501
# lctl set_param mdt.test-MDT0000.root_squash=501:501
```

Test it

```
# touch file
# ls -l file
-rw-r--r-- 1 501 501 0 May  6 23:11 file
#
```


Thank you for your attention

Commissariat à l'énergie atomique et aux énergies alternatives
Centre DAM-Ile de France | 91297 Bruyères-le-Châtel Cedex
T. +33 (0)1 69 26 40 00 | F. +33 (0)1 69 26 70 86
Direction des applications militaires
Département sciences de la simulation et de l'information
Service informatique scientifique et réseaux
Etablissement public à caractère industriel et commercial | RCS Paris B 775 685 019

Lustre Glossary

DNE - Distributed Namespace Environment - feature to aggregate multiple MDTs (possibly on many MDS's) into a single filesystem namespace

IDIF - OST object ID In FID - specific FID range reserved for compatibility with pre-DNE OST objects

IGIF - Inode and Generation In FID - specific FID range reserved for compatibility from Lustre 1.x MDT inode objects

FID - File IDentifier - unique 128-bit identifier for every object within a single filesystem.

LMV - Logical Metadata Volume - client software layer that handles client (llite) access to multiple MDTs

LOD - Logical Object Device - MDS software layer that handles access to multiple MDTs and multiple OSTs

LOV - Logical Object Volume - client software layer that handles client (llite) access to multiple OSTs

MDC - MetaData Client - client software layer that interfaces to the MDS

MDD - Metadata Device Driver - MDS software layer that understands POSIX semantics for file access

MDS - MetaData Server - software service that manages access to filesystem namespace (inodes, paths, permission) requests from the client.

MDT - MetaData Target - storage device that holds the filesystem metadata (attributes, inodes, directories, xattrs, etc)

MGS - Management Server - service that helps clients and servers with configuration

MGT - Management Target - storage device that holds the configuration logs

OFD - Object Filter Device - OSS software layer that handles file IO

OSC - Object Storage Client - client software layer that interfaces to the OST

OSD - Object Storage Device - server software layer that abstracts MDD and OFD access to underlying disk filesystems like ldiskfs and ZFS

OSP - Object Storage Proxy - server software layer that interfaces from one MDS to the OSD on another MDS or another OSS

OSS - Object Storage Server - software service that manages access to filesystem data (read, write, truncate, etc)

OST - Object Storage Target - storage device that holds the filesystem data (regular data files, not directories, xattrs, or other metadata)