EOS workshop 2022

Share ACLs & E-Group Ownership in EOS - a work in progress

IT-ST-PDS

Andreas-Joachim Peters
CERN IT-ST for the EOS team

Introduction [1] Shares

- **contradiction and shortcomings of POSIX and sharing**
  - When we **share a directory** with a given permission set, we want the permission set to be valid for all members referenced in a share. Today act of **sharing manifests in modification of EOS ACLs.** After rewriting an ACL taking into account a new share, the information what part of an ACL originates from a share is lost inside the ACL. Only visible in **OC** database.
    - effectively it is not possible to have permissions managed from **CERNBOX** and the **end-user** or **administrator** at the same time. It would be much better to not rely on **OC** databases to persist sharing information
  - **POSIX** mode/ACLs **cannot express** a permission like **canShare**
  - **POSIX** is **very inefficient** in **managing tree permission** sets
  - many more complications like **overlapping shares**, **moving of shares** and many more …

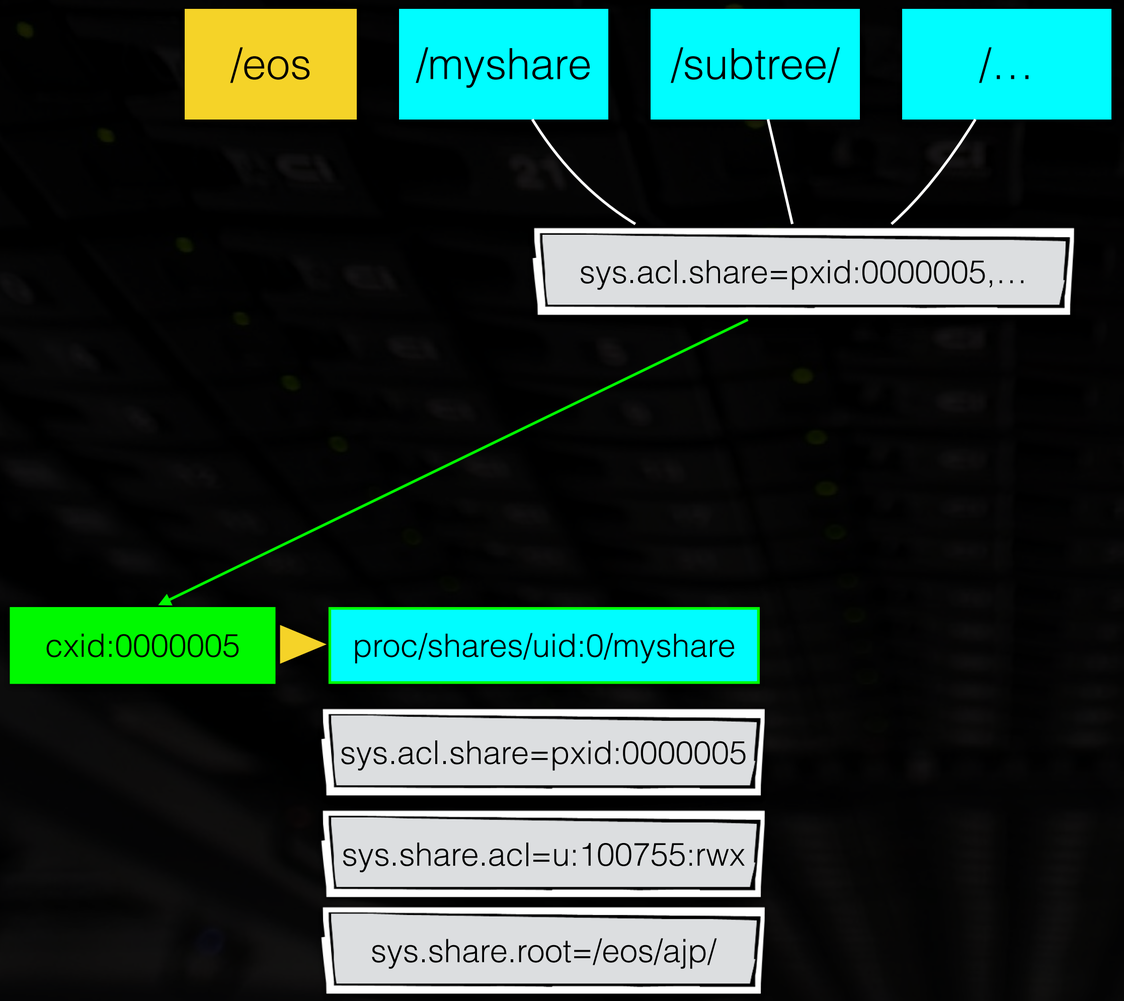
Question: How can we represent sharing in EOS?

Introduction [2] Shared Ownership

- regularly **people** departing from CERN **leaving data orphaned**
- many areas are used as **workgroup** or **project** spaces
- in **POSIX ownership** is limited to a **single person**, which is **by uid** (translated to a name)
  - we can't have a UNIX group 'on the fly'
- EOS recently added an **ACL** syntax, which defines the permissions for an **unnamed owner** - this makes ACLs very uniform and simple over the whole namespace
  `sys.acl=u:owner:rwxc` instead of directory specific `sys.acl=u:foo:rwxc`

Question: how can we express multiple owners in EOS?
With an EGROUP as owner!

Share ACLs



- a shared directory carries an attribute pointing to the definition of a share

- shares are organised by uid as virtual proc directories
- each share contains the defining ACL entry and the logical subtree root where a share starts

Share ACL Evaluation

```
foo:bar r-x---
```

sys.acl=
`u:foo=rwx,u:foo:!r!w`

sys.share.acl=
`u:foo:!rwx`

==========

u:foo:-wx

POSIX Mode

EOS ACL

Share ACL

- we could avoid to use them in the future use it to show POSIX bits for the person looking
- ordered evaluation - last denial wins
- all shares are applied and additive
- denials work only within a single share
- overwrites all EOS ACL denials

Share ACL Interface

```
eos share -h
Usage: share ls|access|create|modify|remove|share|unshare
  share access <name> <username>|<uid> <gid>
    dump all ACL permission when <username> or <uid>/<gid> access the share <name>

  share create <name> <acl> <path>
    create a share with name <name>, acl <acl> under path <path>

  share ls
    list my shares

  share modify <name> <acl>
    modify the acl of the existing share <name>

  share remove <name>
    remove share with name <name>

  share share <name> <acl> <path>
    share the existing share with name <name> using <acl> under <path>

  share unshare <name>
    unshare the existing share with name <name>

Examples:
          eos share ls [-m]
                                              : list all my shares [-m monitoring format]
                                              : list all shares with 'root' role
```

Share ACL Examples

```
EOS Console [root://localhost] |/eos/> share create myshare u:100755:rwx /eos/share/
success: share 'myshare' has been created

EOS Console [root://localhost] |/eos/> share ls
```

uid	name	rule	root	shared
0	**"myshare"**	**u:100755:rwx**	**"/eos/share/"**	**1**

```
EOS Console [root://localhost] |/eos/> attr ls /eos/share/
sys.acl.share="pxid:0006045b"
sys.eos.btime="1646382696.959139969"
```

Share with **EGROUP** Ownership

```
EOS Console [root://localhost] |/eos/> attr set sys.owner.egroup=cms-higgs-analysis /eos/higgs/

EOS Console [root://localhost] |/eos/> share ls
```

uid	name	rule	root	shared
0	**"higgs"**	**u:owner:rwx**	**"/eos/higgs/"**	**1**

-> everybody in the cms-higgs-analysis EGroup acts as an owner with rex permissions

Share ACL Examples

```
EOS Console [localhost] |/eos/> share access myshare root
```

op	perm
read	no
not-read	no
write	no
not-write	no
write-once	no
update	no
not-update	no
browse	no
not-browse	no
chmod	no
not-chmod	no
chown	no
delete	no
not-delete	no
set-quota	no
archive	no
prepare	no
share	no
set-acl	no
egroup	no
eowner	no
mutable	**yes**

```
EOS Console [localhost] |/eos/> share access myshare
100755
```

op	perm
read	**yes**
not-read	no
write	**yes**
not-write	no
write-once	no
update	**yes**
not-update	no
browse	**yes**
not-browse	no
chmod	no
not-chmod	no
chown	no
delete	no
not-delete	no
set-quota	no
archive	no
prepare	no
share	no
set-acl	no
egroup	no
eowner	no
mutable	**yes**

To Do

- **only subtree sharing supported**
  - add single container sharing
  - add single file sharing
- **only sharing by path**
  - add optional sharing by ID to keep shares stable with *mv* operations
- **support ‘shared with me’ command - reverse lookup by user**
  - expensive command because it requires knowledge of all **EGROUP** memberships - with extensive caching
- implement shares ownership by an **EGROUP - currently only by uid**


CERN storage technology
used at the Large Hadron Collider (LHC)
EOS Open Storage
Thank you!
Question or Comments?
eos.web.cern.ch