STORAGE DEVELOPER CONFERENCE
SDC 22
Fremont, CA
September 12-15, 2022
BY Developers FOR Developers

A SNIA Event

Implementing HDFS ACLs in OneFS

Challenges and Solutions

Subin Govind, Senior Principal Engineer, Dell Technologies

Intro

- OneFS is the Operating system that runs on Dell Powerscale systems
- Powerscale was formerly known as Isilon systems
- HDFS ACLs is an implementation of POSIX ACLs
- OneFS ACLs is a superset of NTFS ACLs

Agenda

- Brief Intro of permissions
  - Mode bits
  - OneFS ACLs (a superset of NTFS ACLs)
- POSIX ACLs
- Design approach
- Related issues
  - Concurrency
  - Replication

Permissions in OneFS

A brief intro

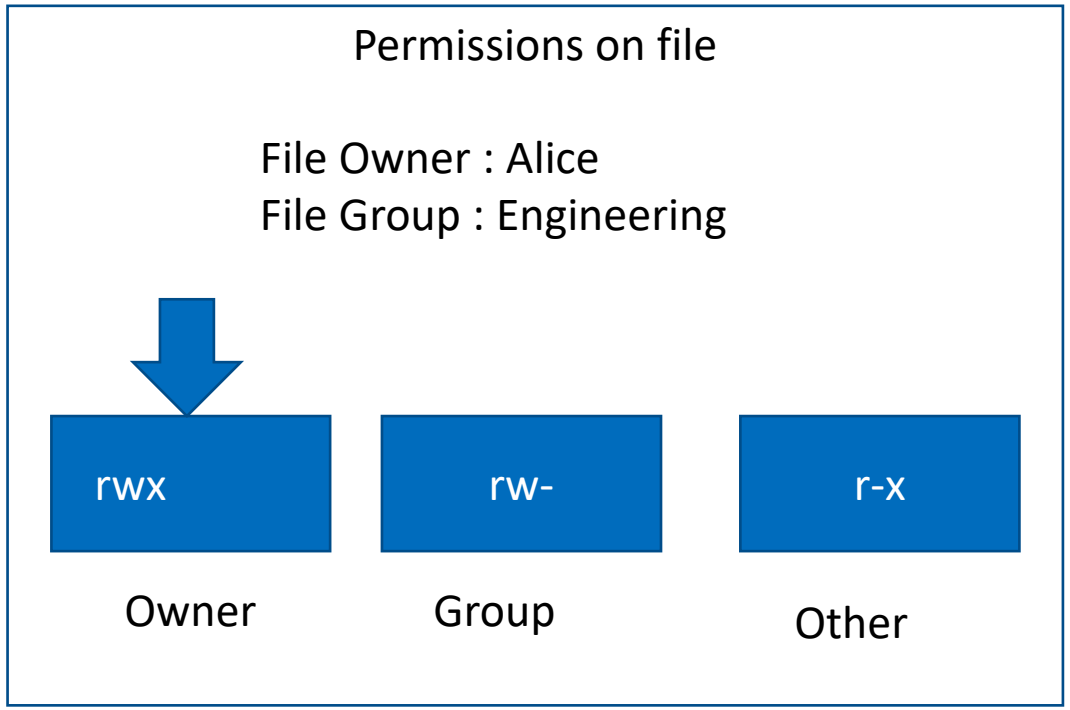
Permissions in OneFS

- POSIX mode bits
- NTFS style ACLs
- ACL policies
  - A way to tweak the access checking behavior

Mode bits

- Permission structure is 9 bits
  - Every file has a UID and GID
  - 3 bits each for file owner, file group and everyone else
- Evaluation algorithm follows first match semantics
- Order
  - file owner
  - file group (not file owner)
  - “other” (not owner or in group)

Mode bits - Example



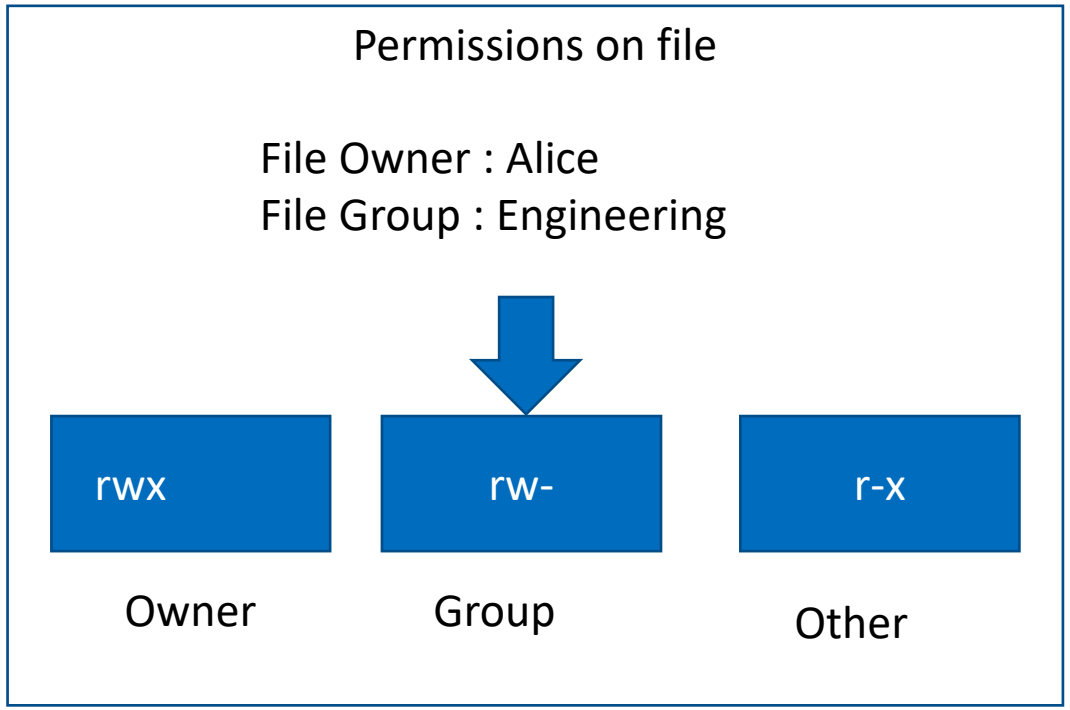
Creds

User Alice
MemberOf
(Engineering)

Requested Perms

rwx

Mode bits - Example



Creds

User Bob
MemberOf
(Engineering)

Requested Perms

r-x

NTFS ACLs

- Consists of a set of Access Control Entries (ACEs)
- Order matters
- Each has
  - Type – ALLOW/DENY
  - Trustee – SID for user/group
  - Access Mask – More fine-grained bit mask of permissions
  - Flags – Govern how inheritance of permissions will happen

NTFS ACLs Access Checking

- An ACE matches a user if :
  - ACE SID is the user's SID
  - ACE SID is a group and user in in that group
- For every matched ACE in ACL :
  - If ACE is ALLOW_ACE
    - add bits to allowed_mask
    - If allowed_mask satisfies requested permissions
      - bail out with success
  - If ACE is a DENY ACE
    - If it denies something in requested_permissions not in allowed_mask
      - bail out with failure

NTFS ACLs - Example

Permissions on file

ALLOW ALICE READ, WRITE, EXECUTE
ALLOW ENGINEERING READ, WRITE
ALLOW EVERYONE READ, EXECUTE

Creds

User Alice
MemberOf (Engineering)

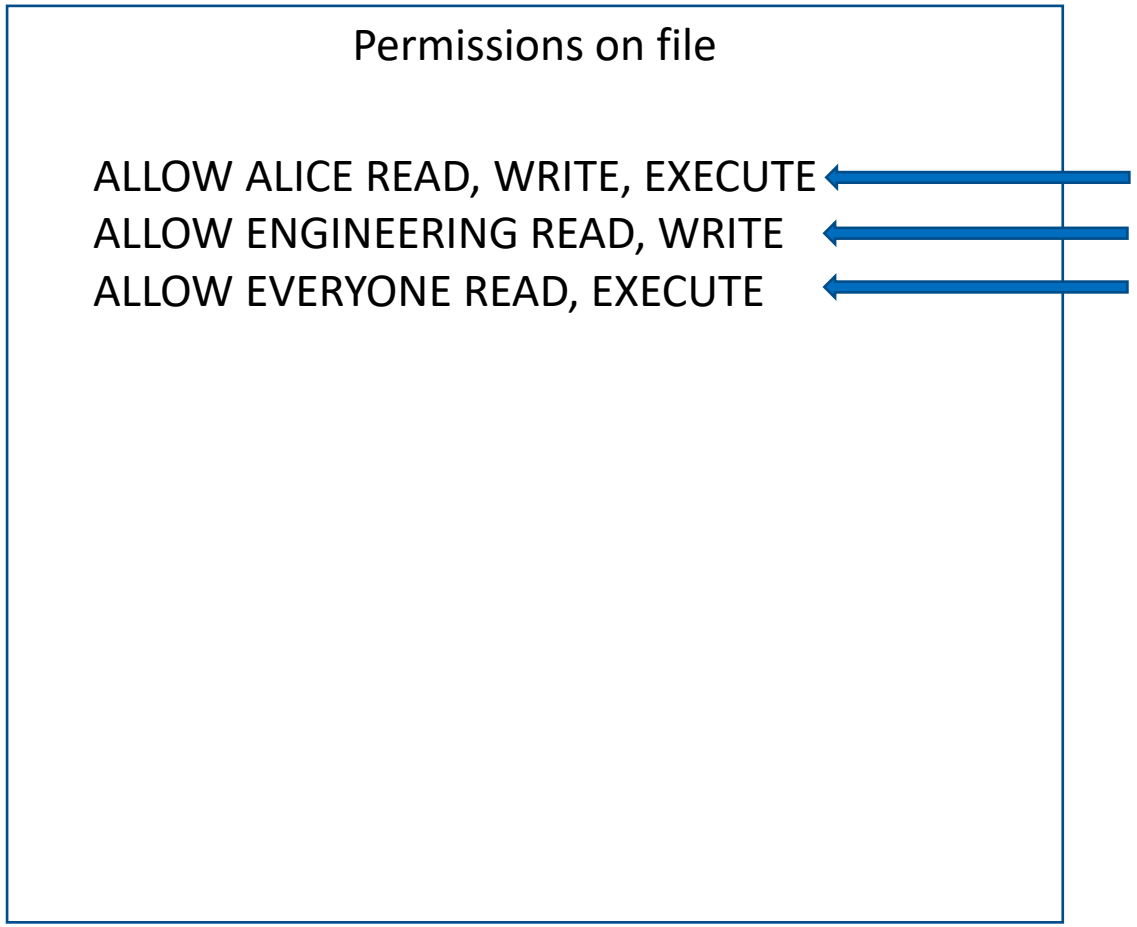
Requested Perms

READ, WRITE, EXECUTE

allowed_mask

READ, WRITE, EXECUTE

NTFS ACLs - Example



Creds

User Bob
MemberOf (Engineering)

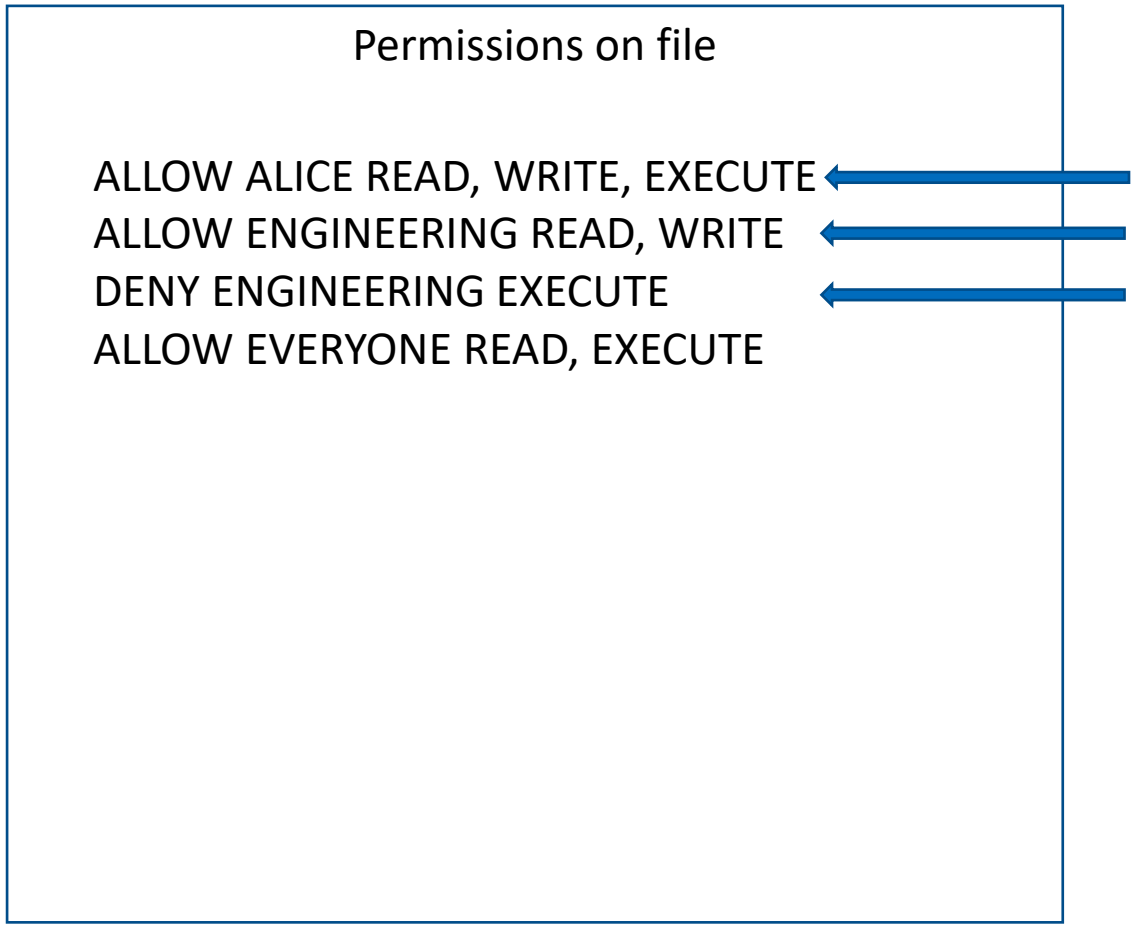
Requested Perms

READ, EXECUTE

allowed_mask

READ, WRITE EXECUTE

NTFS ACLs - Example



Creds

User Bob
MemberOf (Engineering)

Requested Perms

READ EXECUTE

allowed_mask

READ, WRITE

POSIX ACLs

A brief intro

POSIX1.e ACLs

- Defined in the POSIX standard
- Extension of mode bits
- Each file has an owner and group
- Designed to work with mode bits

POSIX ACE format

- Format : type:name:perms
  - Type : user, group, other, mask
  - Name is a string name of a user or group. Can be null
  - Perms are rwx perms
  - Eg user:John:r-x, group:Administrators:rw-
- Three “mandatory” ACEs are defined
  - File owner : user::rwx
  - File group : group::rw-
  - Others : other::r-x
  - Equivalent to user,group,other of mode bits

POSIX ACE format

- ACEs for arbitrary users
  - user:bob:r-x
- ACEs for arbitrary groups
  - group:engineering:r--

POSIX ACLs : Access Checking

- ACEs are checked in a pre-defined order
  - File owner ACE
  - User ACEs
  - Group ACEs
  - “other ACE”

POSIX ACLs : Access Checking

- For every matching ACE in ACL:
  - If its a user ACEs:
    - If it allows the requested perms
      - allow access and bail out
    - else
      - Deny access and bail out
  - If it's a group ACEs:
    - If it allows the requested perms
      - Allow access and bail out
- If no group ACE matched
  - Allow access if allowed by "other" ACE

POSIX ACLs - Example

Permissions on file

File Owner : Alice
File Group : Engineering

user::rwx
user:Bob:rwx
user:Clark:r--
group:Sales:r--
group:Engineering:rw-
other::r-x

Creds

User Bob
MemberOf (Engineering)

Requested Perms

READ, WRITE, EXECUTE

Granted

READ, WRITE, EXECUTE

POSIX ACLs - Example

Permissions on file

File Owner : Alice
File Group : Engineering

user::rwx
user:Bob:rwx
user:Clark:r--
group:Sales:r--
group:Engineering:rw-
other::r-x

Creds

User Clark
MemberOf (Engineering)

Requested Perms

READ, WRITE

Granted

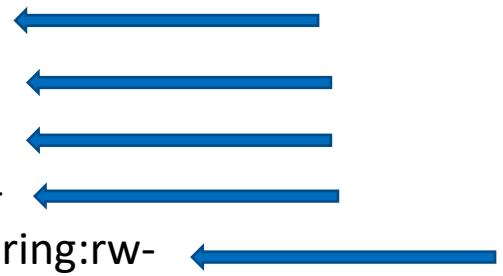
READ

POSIX ACLs - Example

Permissions on file

File Owner : Alice
File Group : Engineering

user::rwx
user:Bob:rwx
user:Clark:r--
group:Sales:r--
group:Engineering:rw-
other::r-x



Creds

User David
MemberOf
(Engineering, Sales)

Requested Perms

READ, WRITE

Granted

READ, WRITE

POSIX ACLs : Mask ACE

- Upper bound of group and named user access
- Used to support chmod
- chmod 0654 file
  - User ACE set to 6
  - Mask ACE set to 5
  - Other ACE set to 4

POSIX ACLs - Example

Permissions on file

File Owner : Alice
File Group : Engineering

user::rwx
user:Bob:rwx
user:Clark:r--
group:Sales:r--
group:Engineering:rw-
other::r-x
mask::r--

Creds

User David
MemberOf
(Engineering, Sales)

Requested Perms

READ, WRITE

Granted

READ

Implementation Approaches

Implementation Approaches

- Translate POSIX ACLs to OneFS ACLs (NTFS-like ACLs)
- Support POSIX ACLs on disk
  - Complete support at all levels of the FS
- Both require translation algorithms for multiprotocol workflows
  - NTFS<->POSIX
  - POSIX <-> mode-bits (defined in the POSIX spec)

Translation POSIX to OneFS ACLs

Advantages	Disadvantages
Low complexity & less code	No support for Mask ACE
Most of the code is in user space	Inaccurate access control

POSIX ACLs on disk

Advantages	Disadvantages
High accuracy access control	Extensive changes to the file system
Non-destructive chmod	

Design Approach

- Translation to NTFS style ACLs on disk
- Added a new ACE type called a mask ACE
- Mask ACE is not visible to SMB/NFS
  - Mask ACE bits are removed from relevant ACEs
- Only visible to HDFS protocol

Design Approach Analysis

Advantages	Disadvantages
Low Complexity	Lower access control accuracy as compared to on-disk
More accurate access control compared to pure translation	
Good first implementation	

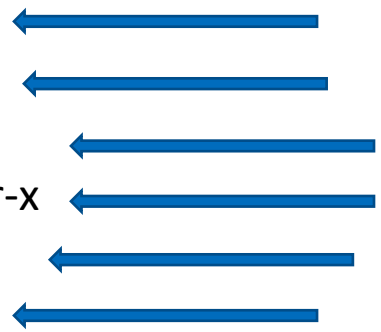
Translation Algorithm

- Loosely follows draft-ietf-nfsv4-acl-mapping-04
- Adds ALLOW ACEs for each POSIX ACE entry
  - Access mask corresponds to POSIX ACE mode bits
- Adds DENY ACE with inverse mask of POSIX ACE
  - Eg if allow is rw-, we add DENY ACE with --x

Translation POSIX to NTFS Example

POSIX ACL

user::rwx
user:Bob:rw-
group::rw-
group:Sales:r-x
other::r-x
mask::r--



OneFS ACL

ALLOW Alice READ WRITE EXECUTE

ALLOW Bob READ WRITE
DENY Bob EXECUTE

ALLOW Engineering READ WRITE
DENY Engineering EXECUTE

ALLOW Sales READ EXECUTE
DENY Sales WRITE

ALLOW Everyone READ EXEC
DENY Everyone WRITE

MASK READ

Translation Algorithm Properties

- HDFS server can expect to get the same ACL it wrote when it reads it back
- Arbitrary NTFS ACLs will translate to inaccurate HDFS ACLs
- Access control inaccuracies
  - ACL has multiple group ACEs with different set of bits eg: rw-, r-x
  - A user is part of multiple groups

Related Issues

Concurrency Issues - Intro

- HDFS Design
  - Single “name node” (metadata node)
  - Many data nodes
  - Designed for high data throughput and lower number of metadata updates
- HDFS on OneFS
  - Just another protocol in the system
  - Scale out cluster where all nodes are identical

Concurrency Issues - Problem

- Some operations need atomic read-update-write cycles
- Example : modify, chmod
- Read-update-cycles should not block/deny file reads and writes
- Needs a new kind of locking in the kernel
  - or maybe not…
- Read-write-cycles are very rare

Concurrency Issues – Solution

- ACL reads return a hash
  - Created from all permission info
- ACL writes pass in the hash
  - If it doesn't match, return retriable error
- This is a form of optimistic concurrency control

Replication Considerations

- OneFS support inter-cluster replication across versions
- Replication to prior versions should maintain security
- POSIX Mask is lost during replication to prior versions

Replication Approach

- We “apply” mask to all relevant ACEs

Before Replication	After Replication
User::rwx Group::rw- Group:admins:rwx User:alice:rwx Other::r— Mask::r--	User::rwx **Group::r—** **Group:admins:r—** **User:alice:r—** Other::r—

Questions?

Please take a moment to rate this session.

Your feedback is important to us.

Translation Inaccuracy Example

POSIX ACL	OneFS ACL
Group:admins:rw- Group:users:r-x	ALLOW admins READ WRITE DENY admins EXEC ALLOW users READ EXEC DENY users WRITE

- If a user is a member of admins and users and user asks for r-x, he will be denied
- Not an issue if admins has rwx
- We order group ACEs in the order rwx, rw-, r--

