NFSv4 ACLS: Where are we going?

Sam Falkner
Lisa Week

Sun Microsystems, Inc.

Agenda

- ACLs in Solaris
  - ZFS and UFS
  - Lessons learned
- Plans for the NFSv4 ACL Internet Draft
- Remaining Issues
- Q&A

ACLs in Solaris

- ZFS
  - Implemented pure NFSv4 ACLs
  - NFSv4 ACL I-D based off of the problems we encountered during the design

    http://www.ietf.org/internet-drafts/draft-ietf-nfsv4-acls-00.txt
- UFS
  - POSIX-draft ACLs native
  - Translate from POSIX-draft to NFSv4 and back per the I-D from Bruce and Marius
  - Lots of “rules” for clients to comply with
    - ACLs must be in certain order, must have certain access mask bits set, etc.

Design Goals of ZFS ACLs

- NFSv4 compliance
- POSIX (not POSIX-draft ACL) compliance
- Sane interaction between mode and ACL file attributes
- Don't delete the ACL upon setting the mode
  - Security reasons...
- Don't ignore the mode passed to CREATE/OPEN

ZFS ACL Implementation Details

- NFSv4 ACLs native
- Pure ACL model
  - Every file has an ACL when it is created
  - Mode is computed from the ACL

Solaris ACL Interfaces

- ls and chmod have been modified to manipulate NFSv4 style and POSIX-draft ACLs
  - setfacl and getfacl to go away
- acl() and facl() syscalls modified to accept both flavors of ACLs
- pathconf() syscall allows you to query the file system for the flavor of ACL it supports

Interaction between ACL and Mode

- What happens to an ACL upon setting a mode?
  - Much more detail in the Internet Draft...
  - If ACE is inheritable, separate into two ACEs
  - If ACE applies to OWNER@, GROUP@ or EVERYONE@ ACE gets READ_DATA, WRITE_DATA, APPEND_DATA and EXECUTE cleared
  - If DENY ACE applies to supplemental users and groups it is left alone
  - If ALLOW ACE applies to supplemental users and groups, prepend a DENY ACE to restrict to owning group permissions
  - Append (or modify them if they are there) 6 ACEs to represent the mode

Interaction between ACL and Mode (cont.)

- In the presence of inheritable ACLs don't ignore the mode passed to to CREATE/OPEN
  - Start with the inherited ACEs making up the ACL of the new file
  - Apply algorithm as previously described

Ambiguities in Access Mask Bits

- NFSv4 ACL I-D attempts to define what each of the access mask bits control
  - ACE4_WRITE_ATTRIBUTES
  - ACE4_READ_ATTRIBUTES
  - ACE4_WRITE_OWNER
  - ACE4_WRITE_NAMED_ATTRS
  - ACE4_READ_NAMED_ATTRS
  - ACE4_WRITE_ACL
  - ACE4_DELETE / DELETE_CHILD

What does “EVERYONE@” mean?

- EVERYONE@ is literally everyone, including the owner and owning group of the file
- Not equivalent to UNIX “other” entity
  - That by definition does not include owner and owning group

Plans for the NFSv4 ACL I-D

- Major portions of the I-D will be incorporated into the NFSv4.1 I-D
  - Further review will happen in the context of that draft
- Remaining text move forward in the hopes of becoming an Informational RFC

Remaining Issues

- Clients can't tell what the server supports natively (POSIX-draft / pure NFSv4)
- Should the umask be applied when there are inheritable ACLs on a directory?
  - Ignoring it is against POSIX rules
    - ZFS doesn't ignore the umask
  - Not ignoring it is annoying
    - May cause users to just set a less restrictive umask or even worse, give up on ACLs

Remaining Issues (cont.)

- Exclusive create
- ACCESS doesn't allow you to check things such as ACE4_READ_ATTRIBUTES
- Append-only files
- User interface standardization
  - Use the same abbreviations to signify different access mask bits, etc.
- Unified GUI

Questions?

sam.falkner@sun.com
lisa.week@sun.com