New Security Document
Options for Addressing the Issue of UNIX Acls

David Noveck

Nfsv4 Working Group Interim Meeting

October 27, 2021

Talk Overview

- Situation so far:
  - Existing Specs
  - Security-02
- Linux server implementation
- Choices going forward:
  - More explicit support within existing framework
  - MAY for each of two semantic models
  - Prepare for uacl attribute in v4.2
- Questions for later discussion

Situation so far
In Existing Specs

- Many attempts to accommodate UNIX Acls
  - Multiple methods of mapping Acl to modes
  - Making support for each mask bit its own OPTIONAL feature
    - Without a way for client to find out which are supported ☹
- Major sources of Specs’ interoperability troubles:
  - Overly broad choices
  - Uncoordinated choices
  - Client left in the dark about sever choices

Situation so far
In Security-02

- Recommended dropping multiple methods of mapping Acls to modes
  - Felt going to one method was pretty much essential
  - Now Consensus Item #27.
- Restricting optionality of mask bits to those required to support UNIX Acls.
  - Now Consensus Item #11.
  - May be possible to restrict further
- Would like to discuss these on list and close on them by -04.

Linux Server Implementation
Supports a UNIX Acl API

- Based on a documented mapping from UNIX to NFSv4 Acls.
  - Available in
    - Expired working group document ☹
- Noteworthy facts:
  - Has no need for alternate mapping of ACLs to modes.
  - Some UNIX Acls map to NFSv4 Acls including DENY Aces

Options Going Forward
More explicit support within existing framework

- Add “need to provide support for UNIX Acls” as a valid reason to bypass SHOULDs.
  - Would be based on Linux implementation and any others we find.
  - Could provide opportunity to further reduce SHOULDs with clear motivation
- This option is probably doable by -03 ☺

Options Going Forward
MAY for each of two semantic models

- Would define two “semantic profiles” for the acl attribute in each fs (full-v4, UNIX acls)
  - Server could choose to implement either one.
  - Dacl attribute would always use the full-v4 one
  - Would allow both models to be supported on a single fs
- Need a way for client to know which model was chosen
  - Might use a special who such as OTHERS@
- Would be doable in -04

Options Going Forward
Prepare for uacl attribute in v4.2

- Near term:
  - Would probably need the work in Slide 6.
  - The work in Slide 7 might be done but would not be worth it.
- Later, define uacl attribute as a v4.2 extension
  - Realistically would have to wait until security doc was an RFC.
  - Would be easy to do using the mapping in the expired I-D.
  - Existing UNIX implementations could be easily adapted to use this.
  - Some choices to make:
    - Possible support for MASK@
    - EVERYONE@ vs. OTHERS@

Questions for Later Discussion
Status of Existing Implementations

- Unix-based server Acl implementations:
  - Which ones exist other than Linux?
  - Do any use the alternate method of computing modes?
  - Do any have an ace mask outside what is allowed in security-02?
  - How do they deal with numeric who values?
- Unix client-side APIs
  - Any other than those based on withdrawn POSIX draft?
  - Do they give rise to interoperability issues that need to be addressed in the security document?

Questions for Later Discussion
What are our needs going forward?

1. Clarified Spec.
   - I’m assuming so.
   - Any disagreement on that?
2. Eventual first-class support for Unix acls.
   - Is it needed and, if so, how soon?
3. Unix client support for full v4 acls
   - History is not encouraging
   - Don’t see how spec can help, other than by narrowing server choices to simplify the client’s work.