NFSv4 ACL Problem

History and Possible Resolutions

David Noveck
For NFSv4 Interim (Revised Version)
December 5, 2023
Contents

• How the problem arose (Slides 3-5)
• Why the problem needs to be fixed (Slides 6-8).
• Dealing with the problem
  • Some commonly proposed non-resolutions (Slides 9-13)
  • Choices in security -07 (Slides 14-15)
  • Working group needs to make some decisions (Slide 16)
How the Problem Arose
Precipitating Issue (Slide One of Two)

• Two ACL models to support
  • NFS4 ACLs proper, based on Windows ACLs
  • UNIX ACLs based on withdrawn POSIX ACL draft

• Many differences between these two models:
  • Substantial, extensible mask vs. three bits (rwx) forever.
  • Support for DENY ACE’s in NFSv4 ACLs
  • Support for AUDIT and ALARM ACEs in NFSv4 ACLs
  • Inconsistent approaches in each model to mapping ACLs to Modes
How the Problem Arose
Precipitating Issue (Slide Two of Two)

• Inability to make pick one of these
  • UNIX ACL’s insufficient for those oriented toward and capable of providing Windows-like ACL Semantics.
  • NFSA4 ACLs had no implementations on UNIX –based servers or APIs to deal with them on UNIX clients.

• Inability to compromise.
  • No support to allow of multiple model via multiple attributes.
  • No attempt to allow either and inform client of the choice
How the Problem Arose

Background Factors

• NFS’s history as implementation-focused
  • Lack of specificity about things “everybody knows”
    • Sorta kinda worked for mode, but not for ACLs
  • Reluctance to mandate new features that required implementation within the filesystem.
    • Didn’t take proper account of the fact that ACL support was not REQUIRED

• General feeling that security was not all that important.
  • Which might have been true for many but problematic for users /implementors of ACLs
Issues in RFCs 7530 and 8881
Full Server Freedom ...

NFSv4 model was made official/canonical but extensive freedom was granted:

• Each ACE mask but was made independently OPTIONAL
• Further license granted as to details of authorization for each one
• Each ACE type is independently OPTIONAL

• Essentially UNIX ACL model was allowed by not rclearly specified
  • So were hybrids and other variants, essentially without limits
• “SHOULD “ used extensively with the meaning of MAY
Issues in RFCs 7530 and 8881 … With Clients Left to Cope

• NFSv4 model was made official /canonical but could not be relied on.

• Extensive freedom was granted:
  • Each ACE mask but was made independently OPTIONAL
  • Further license granted as to how implementations were to behave by promiscuous use of “SHOULD”
    • No discussion of reasons to bypass or consequences
    • Acted like MAY
  • Further choice allowed as to mapping from ACLs to mode.
    • With consequent security/authorization confusion.
Issues in RFCs 7530 and 8881

The Problems this Leaves us

• No way for clients and users to determine the effect of an ACL
  • So users adapt to the server they use.
  • Interoperability is unavailable.

• No way to decide on ACL to meet specific security requirements.
  • Makes it difficult/impossible to create a threat analysis.
Ways to avoid dealing with the problem
Some Possible/Proposed Non-resolutions

• Suggestions like the following are sometimes made:
  • Avoid specification of authorization semantics, as done, more-or-less, in RFC8881 (Slide 10)
  • Make this the job of the filesystem (Slide 11)
  • Decide it is “impossible” (Slide 12)
  • Provide one or more canonical maps between ACL models (Slide 13)

• None of the above are compatible with rfc5661bis effort.
Avoiding Dealing with the Problem
Eschew Authorization Semantics

• Functionally equivalent to what is in RFC881:
  • Describes them but then says they be ignored:
• Not clear how clients could adapt to the consequent uncertainty in the NFSv4 context.
  • Worked for NFSv3 where the modes were a well-understood part of POSIX and exceptions were generally well—known.
  • Once ACLs were added, this approach became untenable.
Avoiding Dealing with the Problem
Make this the Job of the Filesystem

• Although, practically speaking, these decisions are made by file systems., the specification problem remains unresolved
  • Would raise the number of entities that had to work together from two to three, and that might not be doable.
  • Would need us to essentially specify the contents of the vnode interface (outside our charter for good reason)
  • Client still needs semantics it can rely on.
Avoiding Dealing with the Problem
Abandon as Putatively Impossible

• Although this term has been used, it is not clear why this impossible since:
  • The description of the NFSV4 ACL model present in RFC7530, is implementable by servers, although many were unwilling to do so
  • The UNIX ACL model is simpler and there is no reason to assume it is impossible to specify/document.

• One cannot make the specification requirements go way without *proving* they are impossible, and maybe not even then.
Avoiding Dealing with the Problem
Provide Canonical Mappings between ACL Models

• Not clear how this helps since:
  • There already is a canonical map from POSIX to OTW NFsSvr4 ACLs, which could be more prominent in the spec.
  • The reverse canonical mapping is impossible because of the wider semantic range of NFSv4 ACLs.
  • It doesn’t address of allowable ACL that don’t match either model and THAT a basic problem that remains.
Ways to deal with the problem
Some Possibilities in security-07 (Slide One of Two)

• As presented as Consensus Item #61:
  • Deduce ACL type from the ACE types supported.
  • Require that specific set of semantics

• Potential problems
  • Possible hybrids, if they exist, and we need to support them

• Only known approach that will address all minor versions.
Ways to deal with the problem
Some Possibilities in security-07 (Slide Two of Two)

• Provide information about ACL implementation via new OPTIONAL attributes, as a v4.2 extension.
  • Could support hybrids such as an implementation that can store NFSv4 ACLs in general but only enforce UNIX ACLs
• Provide separate attributes for NFSv4 and UNIX ACLs
  • Would work as an extension
  • Should have been done that way at first.
  • Might not make sense now.
Decisions that Need to be Made

1) Will this problem be addressed as part of rfc5661bis effort?
   • If not, how will rfc5661bis paper over the problem?

2) How will rfc5661bis address this problem?
   • One of the ways described in security-07
   • Some other way that somebody comes up with???

• Of these, 1) needs to be addressed as part security-07 adoption call, while 2) can be done later.