NFSv4 Extension Mechanisms

Looking at Minor Versioning in Context

David Noveck

Nfsv4 Working Group meeting at IETF88

November 5, 2013

Overview

- Follow-up to
  - IETF87 talk about minor versioning
  - And to draft-dnoveck-nfs-extension-00.
- Need to:
  - Clarify relationship between extension and minor versioning
  - Use clarified relationship to start solving problems seen with the current model

Review of IETF87 Talk

- Discussed problems with minor versioning
  - Feature-batching issues
    - Process now takes too long.
    - Requiring implementations, while desirable, would make the existing process even longer.
    - Something's got to give
  - Difficulties fixing protocol (i.e. XDR) mistakes
- Had no time to discuss solutions

Lessons from new draft

1. Minor Versioning was a good replacement for major versioning
   - Worked very well for NFSv4.1
2. Minor Versioning doesn't fit well with optional extensions
   - But it's underlying extension mechanism does
3. The two (extension and versioning) can and should be separated
4. MV number changes still have a role
   - And the working group has to decide what that is

Minor Versioning and Protocol Extension

- They are not the same thing
  - Treating the two as a single thing has been a big part of our problem
  - Group has to choose a better relationship

Problems to Address

- Developing Protocol Extensions
  - Problems resulting from “feature batching”
  - See VS and Feature Addition
- Fixing protocol bugs
  - Problems derive from
    - Feature batching
    - Prohibition of (even *compatible*) XDR changes
    - Version number ordering requirements
  - See VS and Fixing Protocol Bugs

Minor Versioning

What has it been good for?

- Excellent Replacement for major versioning:
  - Enabled us to make large protocol changes such as those in v4.1
  - Changes from v4.0 to v4.1 are bigger than those from v2 to v3
  - Doing those same sorts of changes in an NFSv5 would have been much more disruptive.

Minor Versioning

What was it supposed to be good for?

- But NFSv4.1 wasn't the original intention.
- Intention was to do small incremental features
  - There the record is more mixed
  - Can do it, but the issue is with speed/flexibility.
    - Tried to do this (with NFSv4.2) by making minor versions small.
    - Still wound up with a feature latency near five years.

Minor Versioning for Optional Features

- Optional features don't fit a versioning model
  - Since they're optional, later ones can't build upon previous ones
    - Since the previous one may not be present
    - Poor fit for minor versioning ☹
  - But it's certainly better than major versioning ☺
- Minor versioning has some useful elements
  - XDR extension model
  - Concept of (and infrastructure for) optional features

Taking Minor Versioning Apart

So we can put the pieces back together

- A protocol extension mechanism
  - Tastes great, less filling ☺
- Concepts of features, feature statuses and rules to change them
  - Basically sound but needs some further work.
- The minorversion *field* in COMPOUND
  - Not clear when it is useful. See The minorversion field.
- Some rules that derive from versioning concept
  - Version isolation of stateids, fh's,
  - Requirements to support earlier versions

Versioning Straitjacket and Feature Addition

- Problems with protocol extension work flow
  - Deciding on a set of features in advance
    - A “feature batch”
  - Documenting the batch in a single document
  - IESG approval process takes longer
    - As do lots of other things
  - Very hard to change contents

Versioning Straitjacket and Fixing Protocol Bugs

- A number of issues for fixing protocol bugs
- Can’t change XDR in bis or errata-fixing documents
  - Even to make an otherwise-valid extension.
  - Such extensions only done in minor versions
  - Should disallow incompatible but allow compatible extensions
- Minor version numbers add a further difficulty

Features and Feature Status

- Feature definition not very clear
  - Could treat every (non-mandatory) operation, attribute, flag bit, etc. as a feature
  - Most assume that features are coarser-grained
    - But there hasn't been a clear definition of exact rules
    - draft-ietf-nfsv4-minorversion2 makes a start on it
- Feature status anomalies
  - Operations have never been "recommended"
  - Attributes have never been "optional"

Features and Feature Status (continued)

- Original model never realized
  - Features have never been upgraded/downgraded
  - Have to decide whether:
    - To try to make the original model work
    - To change the model to match reality
  - Some other things to decide:
    - Addition of experimental status
    - Do we need a status between optional and mandatory?
      - If not, what about the whole issue with recommended attributes?

Features and Feature Status

Better feature discovery

- There is a need for better feature discovery
  - Trying lots of operations, options can be onerous
  - May need to communicate client characteristics, if only as far as callback support

The minorversion field

When is it clearly useful?

- Useful for transition from v4.0 to v4.1
  - You are picking one of two different protocols
  - These are more different than v2 and v3
- Other cases pose interesting issues
  - See next slide for details

The minorversion field

Is it useful when …

- Only optional features are added?
  - No.
    - What matters is the set of optional features present.
- When a feature becomes mandatory?
  - Possibly but that has never happened.
    - What really matters is if clients insist on having it.
- When a feature becomes recommended?
  - Probably not.
    - What really matters is if other features are built on top of it, and the if the set of features clients want to use depend on those.

Going forward

Motivation

- We need to decide if change is needed
- If so, way forward depends on what is most important to group:
  - Adding extensions
  - Cleaning up problems in existing functionality
  - Establishing a clean foundation for future extensions

Going forward

Paths to consider

- Two major potential foci for an effort:
  - Define new extension model in a working group standards-track document (see RFC Path)
  - Try to adapt our practices without trying to change the underlying minor-versioning-based model (see Change-of-practices Path)
- Might first address the big issue blocking protocol fixes (see A Possible First Step)
  - Might follow that with one of the two foci above.

A Possible First Step

- Decide to make extension-based fixes in an RFC updating a minor version.
  - Essentially, micro-versioning without the extra dot
  - Could have done this for v4.0 migration (adding a SETCLIENTID_PLUS), but decided not to
    - Were able to treat this as a specification problem (and avoid changing XDR)
    - Next time, we might not be so lucky.
  - Unclear if we can just decide this (by WG consensus), but we can try it, when there is need.

Alternatives to “Possible first Step”

- If there is a need for this and it doesn’t work, we would follow RFC Path first
- If the first step works as a WG-only initiative, or you don’t need it, could then stress either
  - RFC Path
  - Change-of-practices Path

RFC Path

- New standards-track RFC about NFSv4 extension model
  - Would apply to all minor versions, existing and future
  - Would separate extension and versioning.
  - Would update 3530[bis], 5661, NFSv4.2
- Possible contents discussed below

RFC Path (possible document contents)

- Rules for extension updates in existing minor versions:
  - To fix protocol bugs
  - To backport working features, when that makes sense
- New feature discovery mechanism.
  - Should include feature names and numeric codes
  - Could be backported to existing versions as an optional feature

RFC Path (possible document contents, continued)

- Publication plans for features/minor versions
  - Ability to publish individual extensions as separate documents.
  - Requirements for feature prototyping before publication
  - What needs to be in minor version documents
- Rework of feature statuses
  - Role of implementation experience
- Discussion of when it makes sense to change minor version number

Change-of-practices Path

Things that could be done within the old framework.

- Avoid premature consensus on minor version contents.
  - Could and should insist on WG documents defining any new features.
  - Might insist on some degree of implementation
- Let feature documents go through IESG review
  - Then the minor version document can be tiny and just reference the feature documents.
- Still leaves the bug-fix/backport issue
  - That requires A Possible First Step or RFC Path

Summary

- Every part of existing model is good for *something*
  - The problem has been trying to use the same model for everything
- The working group has a number of ways to address the problems we've been having
- We have to decide on and focus on our goals
  - New feature development
  - Protocol fixes, since we do make mistakes ☹